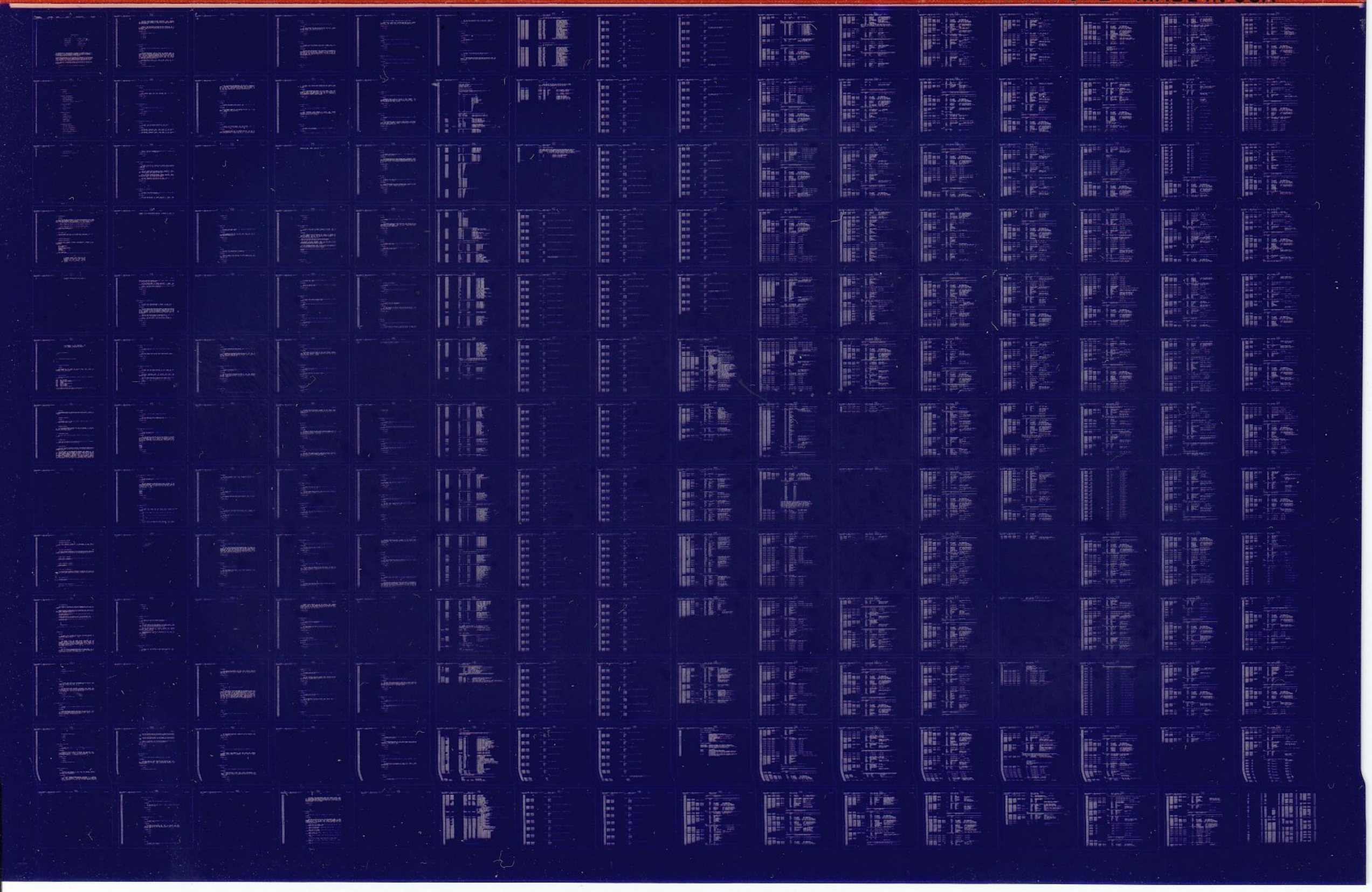
RM03

DISKLESS DIAGNOSTIC
MD-11-DZRMJ-A

EP-DZRMJ-A-DL-A

FICHE 1 OF 2

OCT 1977
digital
MADE IN USA



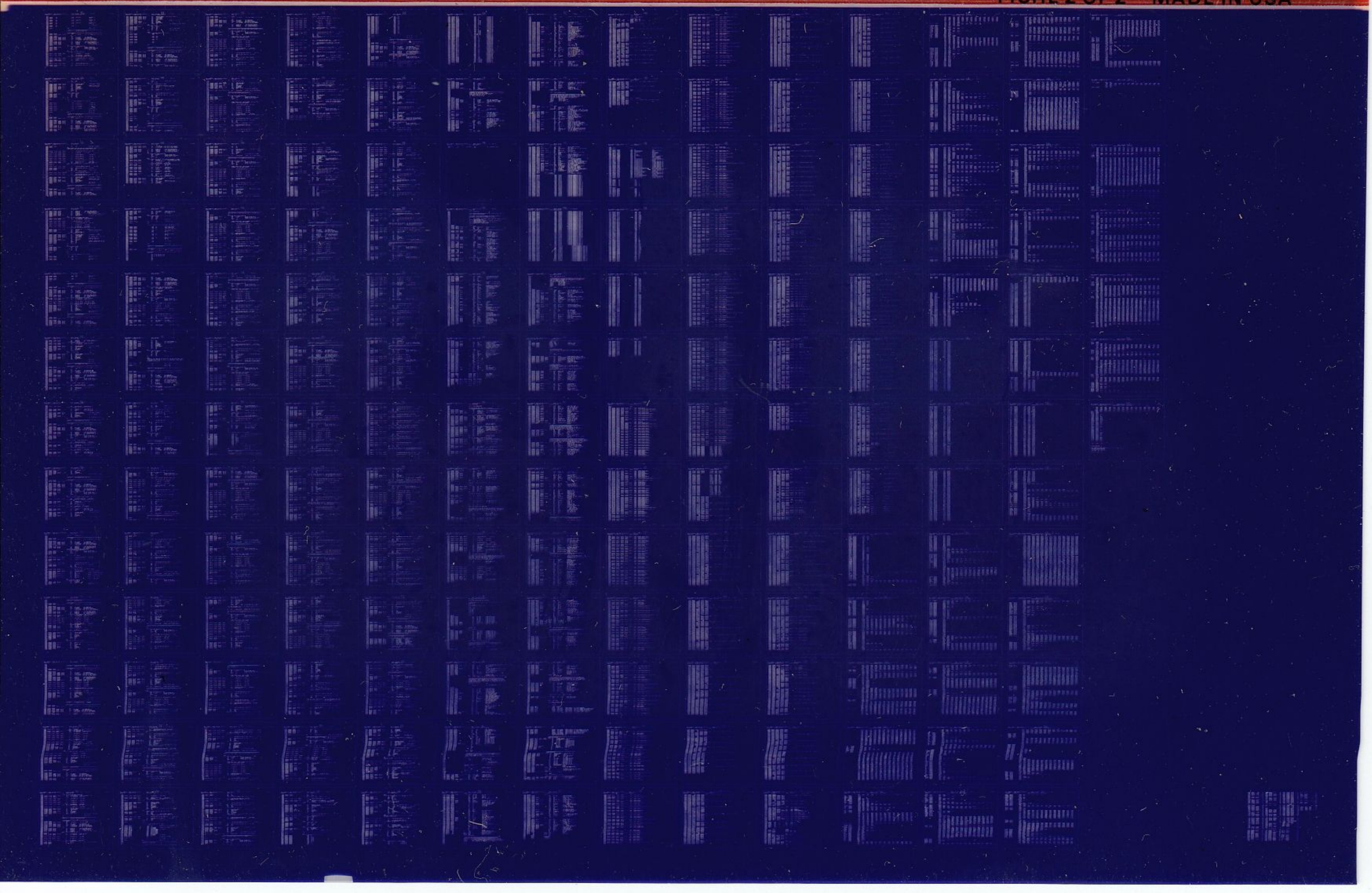
RM03

DISKLESS DIAGNOSTIC
MD-11-DZRMJ-A

EP-DZRMJ-A-DL-A

FICHE 2 OF 2

OCT 1977
digital
MADE IN USA



```
                                .REM    \
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRMJ-A-D
PRODUCT NAME:	RM03 DISKLESS DIAGNOSTIC
DATE CREATED:	1 AUGUST 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

CONTENTS

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM03 DISKLESS DIAGNOSTIC IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL AND DIAGNOSTIC MEANS TO VERIFY THE OPERABILITY OF THE RM03 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 RH70 MASSBUS CONTROLLER;

TO DETECT ERRORS AND FAULTS IN THE RM03 MASSBUS ADAPTER;

TO RESOLVE HARDWARE FAILURES IN THE RH70 AND RM03 TO A FIELD REPLACEABLE MODULE OR MODULES.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM03 DISK SUBSYSTEM, EXCLUDING THE STORAGE MODULE DISK DRIVE AND THE 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 RM03 DISKLESS DIAGNOSTIC:

PDP-11 PROCESSOR
24K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH CONTROLLER

UNIT UNDER TEST,
WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RM03 ADAPTERS.

NOTE

A STORAGE MODULE DISK DRIVE IS NOT REQUIRED FOR EXECUTING THE RM03 DISKLESS DIAGNOSTIC AS LONG AS THE "DEVICE ADDRESS" AND "PLUG ENABLE" LINES ARE PROVIDED BY SOME OTHER MEANS SUCH AS JUMERS.

```
                        THE DEVICE ADDRESS LINES  MAY  FLOAT  TO
                        ADDRESS  7  IF  THERE IS NO OTHER DEVICE
```

WITH ADDRESS 7 ON THE SUBSYSTEM. THE "PLUG ENABLE" LINE MUST BE PROVIDED TO ALLOW MASSBUS DIALOGUE BY THE RM03.

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 STARTS AT LOCATION 200 WHICH USES DEFAULT VALUES OF PARAMETERS AND PROVIDES MAXIMUM TESTING WITH THE SWITCH REGISTER EQUAL TO ZERO.

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 PROGRESS REPORTS

AN END OF PASS REPORT OCCURRS EACH TIME THE PROGRAM IS EXECUTED FOR ALL ADAPTERS IN THE TEST QUE. THE END OF PASS REPORT INCLUDES A MESSAGE AND AN ERROR SUMMARY.

4.3 PERFORMANCE REPORT

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

4.4 PROGRAM HALTS

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

4.5 ERROR REPORTS

THE RM03 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 BEING TESTED, 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

```
   286                                  RESULTS.
```

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM03 DISKLESS DIAGNOSTIC IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET.

5.2 DUAL PORT CONFIGURATIONS

THE RM03 DISKLESS DIAGNOSTIC IS NOT EXECUTABLE ON RM03 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 RM03 DISKLESS DIAGNGOSTIC.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE WILL NOT BE USED DURING THE RM03 DISKLESS DIAGNOSTIC.

5.5 ACT11, APT11 COMPATIBILITY

THE RM03 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 RM03 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 "RM03 DISKLES 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.

TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE RM03 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

CTOD TEST

PURPOSE:

TO VERIFY THAT DATA CAN BE TRANSFERRED TO AND FROM THE RM0, 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

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

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

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.

```
   509
```

PAGE 10

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

REGISTER SELECT TEST

PURPOSE:

TO VERIFY THAT THE REGISTER SELECT LINES ARE 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

PAGE 11

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 RM03 DRIVE.

PROBABLE FAULT:

1. IF MODULE

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

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

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

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.

677
678 "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

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

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

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

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

CONTROL BUS PARITY DETECTION TEST

PURPOSE:

TO TEST THE RM03'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

CONTROL BUS PARITY GENERATION TEST

PURPOSE:

TO TEST THE RM03'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
003000	0
056747	0
135672	0
163135	0

PROBABLE FAULT:

1. TF MODULE
2. ASYNCHRONOUS MASSBUS MODULE

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 L" 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

```
   896                                  WITH ONES.
```

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

3. CS MODULE

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

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.

953 PROCEDURE:

PAGE 18

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 TRANSFERRED TO AND FROM RMDC AND THE PROGRAM CHECKS FOR BIT INTERFERENCE.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

ILLEGAL REGISTER TEST

PURPOSE:

TO TEST ILLEGAL REGISTER ERROR DETECTION IN THE RM03.

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

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

DIAGNOSTIC MODE TEST

PURPOSE:

TO VERIFY THAT "DIAGNOSTIC MODE", BIT 0 OF PMMR1, IS NOT STUCK AT ONE OR ZERO.

PROCEDURE:

THE RM03 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

MOL TEST

PURPOSE:

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

```
1059
1060                                    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

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

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

```
1117                                    AND DVC AND UNS SHOULD BE RESET.
```

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

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

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

```
1174
```

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

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 "LS/T" STATUS FOR EACH PATTERN. THE PROCEDURE IS DONE ONCE FOR 16 BIT FORMAT AND ONCE FOR 18 BIT FORMAT.

PROBABLE FAULT:

1. DS MODULE

2. CS MODULE

RMDA COUNT TEST

PURPOSE

TO VERIFY THAT THE DISK ADDRESS REGISTER (RMDA) INCREMENTS

```
1231                                    PROPERLY.
```

PROCEDURE:

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

PROBABLE FAULT:

1. DS MODULE

RMDC COUNT TEST

PURPOSE:

TO VERIFY THAT THE DESIRED CYLINDER REGISTER, RMDC, INCREMENTS PROPERLY.

PROCEDURE:

THE PROGRAM INCREMENTS RMDC USING DIAGNOSTIC END OF BLOCK, "DEBL", VERIFYING THAT RMDC INCREMENTS THROUGH A COMPLETE CYCLE.

PROBABLE FAULT:

1. DS MODULE

LBT TEST

PURPOSE:

TO INSURE THAT LAST BLOCK TAKEN STATUS, "LBT", CLEARS WHEN RMDA IS WRITTEN, AND SETS WHEN THE LAST SECTOR IS TRANSFERRED.

PROCEDURE:

THE TEST USES DIAGNOSTIC EBL TO SET LBT, AND TRANSFERS TO RMDA TO RESET LBT.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

COMPOSITE ERROR TEST

PURPOSE:

TO TEST "COMPOSITE ERROR", BIT 14 OF RMDS.

PROCEDURE:

THE TEST USES INITIALIZE AND DIAGNOSTIC MODE TO FORCE ALL ERRORS TO ZERO THEN VERIFIES THAT "ERR" IS ZERO. EACH ERROR IS INDIVIDUALLY SET AND "ERR" SHOULD BE ONE FOR EVERY ERROR TESTED. ADDRESSES #2 AND #17 OF THE COMPOSITE ERROR PLA ARE NOT TESTED. "ABORT" AND "EXCEPTION" OUTPUTS OF THE PLA ARE NOT TESTED. THE TEST FAILS IF ERR IS NOT ZERO WITH ALL SET ARGUMENTS ZERO OR IF ERR IS NOT ONE WITH ANY SET ARGUMENT ONE.

PROBABLE FAULT:

1. IF MODULE

WRITE GO TEST

PURPOSE:

TO VERIFY THAT GO CAN BE SET.

PROCEDURE:

THE TEST ENABLES THE DEBUG CLOCK, THEN TRANSFERS A NOP FUNCTION CODE AND GO BIT TO RMCS1, VERIFYING THAT GO SETS. ALL FUNCTION CODES ARE TESTED.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

BRANCH MULTIPLEXOR TEST

PURPOSE:

TO VERIFY THAT THE OUTPUT OF THE COMMAND SEQUENCER BRANCH

```
 1344                                   MULTIPLEXOR DOES NOT HAVE A FAULT.
```

PROCEDURE:

WITH DEBUG CLOCK ENABLED, THE TEST USES VARIOUS FUNCTION CODES AND REGISTER CONDITIONS TO ADDRESS THE TEST BIT MULTIPLEXOR SUCH THAT THE TEST BIT, BIT12 OF RMMR2, CAN BE CHECKED FOR A STUCK FAULT.

PROBABLE FAULT:

1. CS MODULE

SET/RESET GO TEST

PURPOSE:

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

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

SET PULSE TEST

PURPOSE:

TO VERIFY THAT THE COMMAND SEQUENCER CAN GENERATE SET PULSE.

PROCEDURE:

WHICH 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 GENERATED.

PROBABLE FAULT:

1. CS MODULE

SET/RESET IVC TEST

PURPOSE:

TO TEST "INVALID COMMAND" STATUS FOR EACH FUNCTION CODE.

PROCEDURE:

THE PROGRAM RESETS VOLUME VALID USING "MAINTENANCE UNIT READY", BIT09 OF RMMR1, THEN LOADS THE FUNCTION CODE AND GO IN RMCS1. EACH FUNCTION CODE IS TESTED AND "IVC", BIT 12 OF RMER2 IS CHECKED.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

```
1457
```

SET LSC TEST

PURPOSE:

TO VERIFY THAT "LOSS OF SYSTEM CLOCK" CAN SET AND RESET.

PROCEDURE:

THE TEST ENABLES THE DEBUG CLOCK AND SETS THE GO BIT. AFTER WAITING ENOUGH TIME FOR THE ONE SHOT TO SET, THE TEST DISABLES THE DEBUG CLOCK AND VERIFIES THAT LSC SETS.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

DECODE TEST

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

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

ILLEGAL FUNCTION TEST

PURPOSE:

TO TEST ILLEGAL FUNCTION ERROR IN THE RM03.

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

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

```
1569                    SHOULD  BE  ONE.  DEBUG  CLOCK  IS  ENABLED  TO  PREVENT  GO FROM
1570                    RESETTING BEFORE STATUS IS SAMPLED.
```

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

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.

. BITS 10-12 OF RMOF ARE SET BY A MOVE INSTRUCTION AND THE TEST PASSES IF THESE BITS ARE ZERO AFTER THE RIP COMMAND.

. THE DESIRED CYLINDER REGISTER, RMDC, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-03, OR BITS 04-07, OR BITS 08-09 ARE ZERO AFTER THE RIP COMMAND.

. THE DISK ADDRESS REGISTER, RMDA, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-03, OR BITS 04-07, OR BITS 08-11, OR BITS 12-15 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

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

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

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

RETURN TO CENTER TEST

PURPOSE:

TO VERIFY 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

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

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

RUN AND GO TEST

PURPOSE:

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

PROCEDURE:

THE RM03 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

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

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

INVALID SECTOR/TRACK TEST

PUPOSE:

TO VERIFY THAT INVALID SECTOR AND TRACK 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

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

SET AOE TEST

PRUPOSE:

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.

PROBABLE FAULT:

1. DS MODULE

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:

PAGE 35

"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 RMER1, SHOULD BE ONE.

PROBABLE FAULT:

1. IF MODULE

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

DVA/DPR STATUS CHECK PURPOSE:

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

PROCEDURE:

DPR AND DVA ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

1. IF MODULE

PAGE 36

PORT REQUEST TEST, PART 1/3

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

PORT REQUEST TEST, PART 2/3

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

DATA COMMAND TESTS

PURPOSE:

TO VERIFY THE COMMAND SEQUENCER DURING DATA COMMANDS.

2011 PROCEDURE:

THIS TEST, LIKE RECALIBRATE, SEEK, AND SEARCH TESTS, USES THE MAINTENANCE REGISTER TO SIMULATE DRIVE CONDITIONS AND FORCE THE COMMAND SEQUENCER THROUGH EACH BRANCH PATH. ADDITIONAL ITEMS WHICH ARE TESTED INCLUDE OFFSET PLUS AND MINUS ON THE TAG BUS AND "ENABLE SEARCH", BIT 11 OF RMMR1.

PROBABLE FAULT:

1. CS MODULE

DATA SEQUENCER READ TEST

PURPOSE:

TO TEST THE CS AND DS MODULES FOR EXECUTION OF A READ COMMAND.

PROCEDURE:

THE RM03 IS INITIALIZED AND, USING DIAGNOSTIC MODE, A READ COMMAND IS STEPPED TO THE POINT WHERE THE COMMAND SEQUENCER ENABLES SEARCH. THE PROGRAM THEN STARTS STEPPING THE DATA SEQUENCER ON THE CS MODULE AND MONITORS HARDWARE OPERATION USING THE FOLLOWING STATUS:

.PLFS, BIT 10 OF RMMR1, SHOWS THE PERIOD WHEN THE DATA PROM HAS ENABLED SYNC BYTE DETECTION;

.PDA, BIT 08 OF RMMR1, SHOWS WHEN THE DATA PROM IS IN THE DATA AREA OF THE SECTOR;

.PHA, BIT 07 OF RMMR1, SHOWS WHEN THE DATA PROM IS IN THE HEADER AREA OF THE SECTOR;

.BB00, BIT00 OF RMMR2, SHOWS THE STATE OF WRITE GATE PROVIDING TAG IS HIGH;

.BB01, BIT 01 OF RMMR2, SHOWS THE STATE OF READ GATE PROVING TAG IS HIGH;

.WC, BIT 05 OF RMMR1, SHOWS THE STATE OF WORD CLOCK;

.EECC, BIT 04 OF RMMR1, SHOWS WHEN ECC CHARACTER IS ENABLED DURING WRITE;

.ECRC, BIT 09 OF RMMR1, SHOWS WHEN CRC CHARACTER IS ENABLED DURING WRITE;

PAGE 38

.WD, BIT 03 OF RMMR1, SHOWS THE STATE OF WRITE DATA.

ADDITIONALLY, STATUS BITS IN RMER1 ARE USED TO TEST COMMAND EXECUTION. THE WRITE ONLY BITS OF RMMR1 ARE USED TO SIMULATE DRIVE STATUS AND CONTROL SIGNALS, AND TO SPECIFY READ DATA.

PROBABLE FAULT:

1. CS MODULE

2. DS MODULE

DATA SEQUENCER WRITE TEST

PURPOSE:

TO TEST THE CS AND DS MODULES FOR EXECUTION OF A WRITE COMMAND.

PROCEDURE:

THE PROCEDURE IS THE SAME AS THE DATA SEQUENCER READ TEST.

PROBABLE FAULT:

1. CS MODULE

2. DS MODULE

DATA SEQUENCER FORMAT TEST

PURPOSE:

TO TEST CS AND DS MODULES FOR EXECUTION OF A FORMAT COMMAND.

PROCEDURE:

THE PROCEDURE IS THE SAME AS THE DATA SEQUENCER READ TEST.

PROBABLE FAULT:

1. CS MODULE

2. DS MODULE

18 BIT WORD CLOCK TEST

PURPOSE:

TO TEST THE GENERATION OF WORD CLOCK IN 18 BIT FORMAT.

PROCEDURE:

DIAGNOSTIC MODE IS USED TO EXECUTE A READ COMMAND. THE PROGRAM VERIFIES THAT WORD CLOCK SWITCHES TO 18 BIT FORMAT AT THE DATA AREA AND SWITCHES BACK TO 16 BIT FORMAT DURING ECC. THE TEST IS REPEATED FOR A WRITE COMMAND.

PROBABLE FAULT:

1. CS MODULE

2. DS MODULE

HCRC SET TEST

PURPOSE:

TO VERIFY THAT "HCRC" STATUS SETS WHEN THE HEADER CRC WORD IS IN ERROR.

PROCEDURE:

USING DIAGNOSTIC MODE TO EXECUTE A READ COMMAND, THE PROGRAM STEPS THE DATA SEQUENCER THROUGH THE HEADER AND PROVIDES READ DATA WHICH SHOULD CAUSE A CRC ERROR. THE TEST FAILS IF "HCRC", BIT 08 OF RMER1 DOES NOT SET.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

HCE SET TEST

PURPOSE:

TO VERIFY THAT "HCE" STATUS SETS IF THE HEADER DOES NOT COMPARE WITH RMDC AND RMDA.

PROCEDURE:

THE PROGRAM EXECUTES A READ COMMAND USING DIAGNOSTIC MODE. READ DATA IS SUPPLIED WHICH CAUSES A HEADER COMPARE ERROR IN BITS 00-03 OF HEADER WORD ONE AND THE PROGRAM VERIFIES THAT "HCE", BIT 07 OF RMER1 IS ONE. THIS PROCESS IS REPEATED FOR BITS 04-07 AND 08-11.

PROBABLE FAULT:

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

FER SET TEST

PURPOSE:

TO VERIFY THAT "FER" STATUS SETS WHEN THE FORMAT BIT DOES NOT COMPARE WITH THE HEADER.

PROCEDURE:

THIS PROGRAM EXECUTES A READ COMMAND USING DIAGNOSTIC MODE AND FORCES A FORMAT ERROR IN THE FIRST HEADER WORD. THE TEST FAILS IF "FER", BIT 04 OF RMER1 DOES NOT SET.

PROBABLE FAULT:

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

DCK SET TEST

PURPOSE:

TO VERIFY THAT "DCK" STATUS SETS WHEN A CORRECTABLE ECC ERROR OCCURRS.

PROCEDURE:

THE PROGRAM SIMULATES A READ OPERATION AND FORCES AN ECC ERROR WHICH IS CORRECTABLE. "DCK", BIT 15 OF RMER1 SHOULD BE SET AND "ECH", BIT 06 OF RMER1 SHOULD BE RESET. RMEC1 AND RMEC2 SHOULD PERMIT DATA CORRECTION.

PROBABLE FAULT:

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

ECH SET TEST

PURPOSE:

TO VERIFY THAT "ECH" STATUS SETS WHEN A NONCORRECTABLE ECC ERROR OCCURRS DURING READ.

PROCEDURE:

THIS TEST IS LIKE THE DCK SET TEST EXCEPT THE PROGRAM USES AN UNCORRECTABLE ERROR BURST.

PROBABLE FAULT:

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

DVA/DPR

STATUS CHECK PURPOSE:

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

PROCEDURE:

DPR AND DVA ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

1. IF MODULE

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

SET AOE TEST

PRUPOSE:

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.

```
2334                    END OF BLOCK IS FORCED TO INCREMENT THE SECTOR ADDRESS,  AND  THE
2335                    TEST VERIFIES THAT "AOE" IS SET.
```

PROBABLE FAULT:

1. DS MODULE

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

INVALID SECTOR/TRACK TEST

PUPOSE:

TO VERIFY THAT INVALID SECTOR AND TRACK 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

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

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

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

SET LSC TEST

PURPOSE:

TO VERIFY THAT "LOSS OF SYSTEM CLOCK" CAN SET AND RESET.

PROCEDURE:

THE TEST ENABLES THE DEBUG CLOCK AND SETS THE GO BIT. AFTER WAITING ENOUGH TIME FOR THE ONE SHOT TO SET, THE TEST DISABLES THE DEBUG CLOCK AND VERIFIES THAT LSC SETS.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

DECODE TEST

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.

```
2500
2501                            -       PROBABLE FAULT:
```

1. IF MODULE

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

PORT REQUEST TEST, PART 3/3

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

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

RESET ATA BY GO TEST

PURPOSE:

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

PROCEDURE:

THE PROGRAM SETS MAINTENANCE UNIT READY WHICH SHOULD CAUSE AN ATTENTION. THEN, WITH DEBUG CLOCK ENABLED, GO IS SET, AND ATA SHOULD BE ZERO.

PROBABLE FAULT:

1. IF MODULE

UNIT READY ATA TEST

PURPOSE:

TO VERIFY THAT ONE-ZERO AND ZERO-ONE TRANSITIONS OF UNIT READY SET ATTENTION.

2610 PROCEDURE:

THE TEST USES DIAGNOSTIC MODE TO FORCE BOTH TRANSITIONS OF UNIT READY AND VERIFIES THAT ATA SETS WITH EACH TRANSITION.

PROBABLE FAULT:

1. IF MODULE

ERROR ATA TEST

PURPOSE:

TO VERIFY THAT ATTENTION SETS WHEN COMPOSITE ERROR OCCURS WHILE GO IS OFF.

PROCEDURE:

THE PROGRAM CLEARS THE DEVICE AND SETS AN ERROR, THEN VERIFIES ATA IS ON.

PROBABLE FAULT:

1. IF MODULE

REGISTER TRANSFER ATA TEST

TO VERIFY THAT ATTENTION SETS WHEN ANY REGISTER, EXCEPT FOR RMAS AND RMCS, IS WRITTEN WHILE COMP ERROR IS SET.

PROCEDURE:

THE PROGRAM FORCES AN ERROR THEN RESETS ATTENTION FROM THE ERROR. THE PROGRAM THEN WRITES RMAS AND RMCS AND VERIFIES THAT NO ATTENTION OCCURS, AND WRITES RMDC AND VERIFIES THAT ATTENTION DOES OCCUR.

PROBABLE FAULT:

1. IF MODULE

\

```
                                ;PROGRAM REVISION #001
                                
                                .TITLE  MD-11-DZRMJA-A, RM03 DISKLESS DIAGNOSTIC
                                ;*COPYRIGHT (C) 1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DOUG RIIKONEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
                                .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>
                                ;*         7            TN128
                                ;*         6            TN64
                                ;*         5            TN32
                                ;*         4            TN16
                                ;*         3            TN8
                                ;*         2            TN4
                                ;*         1            TN2
                                ;*         0            TN1
                                .SBTTL  BASIC DEFINITIONS
                                
                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
                                
                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                                
                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
```

```
2721            000004          R4=     %4              ;;GENERAL REGISTER
2722            000005          R5=     %5              ;;GENERAL REGISTER
2723            000006          R6=     %6              ;;GENERAL REGISTER
2724            000007          R7=     %7              ;;GENERAL REGISTER
2725            000006          SP=     %6              ;;STACK POINTER
2726            000007          PC=     %7              ;;PROGRAM COUNTER
2727
2728                            ;*PRIORITY LEVEL DEFINITIONS
2729            000000          PR0=    0               ;;PRIORITY LEVEL 0
2730            000040          PR1=    40              ;;PRIORITY LEVEL 1
2731            000100          PR2=    100             ;;PRIORITY LEVEL 2
2732            000140          PR3=    140             ;;PRIORITY LEVEL 3
2733            000200          PR4=    200             ;;PRIORITY LEVEL 4
2734            000240          PR5=    240             ;;PRIORITY LEVEL 5
2735            000300          PR6=    300             ;;PRIORITY LEVEL 6
2736            000340          PR7=    340             ;;PRIORITY LEVEL 7
2737
2738                            ;*"SWITCH REGISTER" SWITCH DEFINITIONS
2739            100000          SW15=   100000
2740            040000          SW14=   40000
2741            020000          SW13=   20000
2742            010000          SW12=   10000
2743            004000          SW11=   4000
2744            002000          SW10=   2000
2745            001000          SW09=   1000
2746            000400          SW08=   400
2747            000200          SW07=   200
2748            000100          SW06=   100
2749            000040          SW05=   40
2750            000020          SW04=   20
2751            000010          SW03=   10
2752            000004          SW02=   4
2753            000002          SW01=   2
2754            000001          SW00=   1
2755                            .EQUIV  SW09,SW9
2756                            .EQUIV  SW08,SW8
2757                            .EQUIV  SW07,SW7
2758                            .EQUIV  SW06,SW6
2759                            .EQUIV  SW05,SW5
2760                            .EQUIV  SW04,SW4
2761                            .EQUIV  SW03,SW3
2762                            .EQUIV  SW02,SW2
2763                            .EQUIV  SW01,SW1
2764                            .EQUIV  SW00,SW0
2765
2766                            ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
2767            100000          BIT15=  100000
2768            040000          BIT14=  40000
2769            020000          BIT13=  20000
2770            010000          BIT12=  10000
2771            004000          BIT11=  4000
2772            002000          BIT10=  2000
2773            001000          BIT09=  1000
2774            000400          BIT08=  400
2775            000200          BIT07=  200
2776            000100          BIT06=  100
```

```
                        BITO5=  40
                        BITO4=  20
                        BITO3=  10
                        BITO2=  4
                        BITO1=  2
                        BITOO=  1
                        .EQUIV  BITO9,BIT9
                        .EQUIV  BITO8,BIT8
                        .EQUIV  BITO7,BIT7
                        .EQUIV  BITO6,BIT6
                        .EQUIV  BITO5,BIT5
                        .EQUIV  BITO4,BIT4
                        .EQUIV  BITO3,BIT3
                        .EQUIV  BITO2,BIT2
                        .EQUIV  BITO1,BIT1
                        .EQUIV  BITOO,BITO

                        ;*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

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

004000                  DVA     =       BIT11           ;DEVICE AVAILABLE-READ ONLY
000040                  F4      =       BITO5           ;FUNCTION CODE
000020                  F3      =       BITO4           ;FUNCTION CODE
000010                  F2      =       BITO3           ;FUNCTION CODE
000004                  F1      =       BITO2           ;FUNCTION CODE
000002                  F0      =       BITO1           ;FUNCTION CODE
000001                  GO      =       BITOO           ;GO BIT
000077                  FNCMSK  =       000077          ;FUNCTION CODE MASK

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

000000                  NOP     =       000000          ;NOP COMMAND
000002                  ILF02   =       000002          ;ILLEGAL COMMAND
000004                  SEEK    =       000004          ;SEEK COMMAND
000006                  RECAL   =       000006          ;RECALIBRATE COMMAND
000010                  DRVCLR  =       000010          ;DRIVE CLEAR COMMAND
000012                  RLEASE  =       000012          ;RELEASE COMMAND
000014                  OFFSET  =       000014          ;OFFSET COMMAND
000016                  RTC     =       000016          ;RETURN TO CENTERLINE COMMAND
000020                  RIP     =       000020          ;READ IN PRESET COMMAND
000022                  PAKACK  =       000022          ;PACK ACKNOWLEDGE COMMAND
```

```
2833            000022          PACACK  =       PAKACK
2834            000024          ILF24   =       000024          ;ILLEGAL COMMAND
2835            000026          ILF26   =       000026          ;ILLEGAL COMMAND
2836            000030          SEARCH  =       000030          ;SEARCH COMMAND
2837            000030          ILF30   =       000030          ;ILLEGAL COMMAND
2838            000032          ILF32   =       000032          ;ILLEGAL COMMAND
2839            000034          ILF34   =       000034          ;ILLEGAL COMMAND
2840            000036          ILF36   =       000036          ;ILLEGAL COMMAND
2841            000040          ILF40   =       000040          ;ILLEGAL COMMAND
2842            000042          ILF42   =       000042          ;ILLEGAL COMMAND
2843            000044          ILF44   =       000044          ;ILLEGAL COMMAND
2844            000046          ILF46   =       000046          ;ILLEGAL COMMAND
2845            000050          WCD     =       000050          ;WRITE CHECK DATA COMMAND
2846            000052          WCH     =       000052          ;WRITE CHECK HEADER AND DATA
2847            000054          ILF54   =       000054          ;ILLEGAL COMMAND
2848            000056          ILF56   =       000056          ;ILLEGAL COMMAND
2849            000060          WD      =       000060          ;WRITE DATA COMMAND
2850            000062          WH      =       000062          ;WRITE HEADER AND DATA COMMAND
2851            000064          ILF64   =       000064          ;ILLEGAL COMMAND
2852            000066          ILF66   =       000066          ;ILLEGAL COMMAND
2853            000070          RD      =       000070          ;READ DATA COMMAND
2854            000072          RH      =       000072          ;READ HEADER AND DATA COMMAND
2855            000074          ILF74   =       000074          ;ILLEGAL COMMAND
2856            000076          ILF76   =       000076          ;ILLEGAL COMMAND
2857
2858                            ;RMDA   DISK ADDRESS REGISTER
2859
2860            002000          TA4     =       BIT10           ;TRACK ADDRESS 4
2861            001000          TA2     =       BIT09           ;TRACK ADDRESS 2
2862            000400          TA1     =       BIT08           ;TRACK ADDRESS 1
2863            000020          SA16    =       BIT04           ;SECTOR ADDRESS 16
2864            000010          SA8     =       BIT03           ;SECTOR ADDRESS 8
2865            000004          SA4     =       BIT02           ;SECTOR ADDRESS 4
2866            000002          SA2     =       BIT01           ;SECTOR ADDRESS 2
2867            000001          SA1     =       BIT00           ;SECTOR ADDRESS 1
2868
2869                            ;TRACK,SECTOR MASKS
2870
2871            003400          TADMSK  =       003400          ;TRACK ADDRESS MASK
2872            000037          SADMSK  =       000037          ;SECTOR ADDRESS MASK
2873
2874                            ;RMDS   DRIVE STATUS REGISTER
2875
2876            100000          ATA     =       BIT15           ;ATTENTION ACTIVE
2877            040000          ERR     =       BIT14           ;COMPOSITE ERROR
2878            020000          PIP     =       BIT13           ;POSITIONING IN PROGRESS
2879            010000          MOL     =       BIT12           ;MEDIUM ON LINE
2880            004000          WRL     =       BIT11           ;WRITE LOCK
2881            002000          LBT     =       BIT10           ;LAST BLOCK TRANSFERRED
2882            001000          PGM     =       BIT09           ;PROGRAMMABLE
2883            000400          DPR     =       BIT08           ;DRIVE PRESENT
2884            000200          DRY     =       BIT07           ;DRIVE READY
2885            000100          VV      =       BIT06           ;VOLUME VALID
2886            000001          OM      =       BIT00           ;OFFSET MODE ACTIVE
2887
2888                            ;RMER1  ERROR REGISTER #1
```

```
2889
2890    100000          DCK     =       BIT15           ;DATA CHECK ERROR
2891    040000          UNS     =       BIT14           ;DRIVE UNSAFE
2892    020000          OPI     =       BIT13           ;OPERATION INCOMPLETE
2893    010000          DTE     =       BIT12           ;DRIVE TIMING ERROR
2894    004000          WLE     =       BIT11           ;WRITE LOCK ERROR
2895    002000          IAE     =       BIT10           ;INVALID ADDRESS ERROR
2896    001000          AOE     =       BIT09           ;ADDRESS OVERFLOW ERROR
2897    000400          HCRC    =       BIT08           ;HEADER CRC ERROR
2898    000200          HCE     =       BIT07           ;HEADER COMPARE ERROR
2899    000100          ECH     =       BIT06           ;ECC "HARD" ERROR
2900    000040          WCF     =       BIT05           ;WRITE CLOCK FAILURE
2901    000020          FER     =       BIT04           ;FORMAT ERROR
2902    000010          PAR     =       BIT03           ;PARITY ERROR
2903    000004          RMR     =       BIT02           ;REGISTER MODIFICATION REFUSED
2904    000002          ILR     =       BIT01           ;ILLEGAL REGISTER
2905    000001          ILF     =       BIT00           ;ILLEGAL FUNCTION
2906
2907    115760          NDTMSK  =       DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
2908                    ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
2909                    ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
2910
2911                    ;RMAS   ATTENTION SUMMARY REGISTER
2912
2913    000377          ATNMSK  =       377             ;MASK FOR ATTENTION BITS
2914
2915                    ;RMLA   LOOK AHEAD REGISTER
2916
2917    002000          SC4     =       BIT10           ;SECTOR COUNT = 16
2918    001000          SC3     =       BIT09           ;SECTOR COUNT = 8
2919    000400          SC2     =       BIT08           ;SECTOR COUNT = 4
2920    000200          SC1     =       BIT07           ;SECTOR COUNT = 2
2921    000100          SC0     =       BIT06           ;SECTOR COUNT = 1
2922
2923    003700          SCTMSK  =       003700          ;SECTOR COUNT MASK
2924
2925                    ;RMMR   MAINTENANCE REGISTER
2926
2927                    ;       WRITE ONLY BITS
2928
2929    100000          DBCK    =       BIT15           ;DEBUG CLOCK
2930    040000          DBEN    =       BIT14           ;DEBUG CLOCK ENABLE
```

```
2931            020000          DEBL    =       BIT13           ;DIAGNOSTIC END OF BLOCK
2932            010000          DTO     =       BIT12           ;DIAGNOSTIC TIMEOUT
2933            004000          MCLK    =       BIT11           ;MAINTENANCE CLOCK
2934            002000          MRD     =       BIT10           ;READ DATA
2935            001000          MUR     =       BIT09           ;UNIT READY
2936            000400          MOC     =       BIT08           ;ON CYLINDER
2937            000200          MSER    =       BIT07           ;SEEK ERROR
2938            000100          MDF     =       BIT06           ;DRIVE FAULT
2939            000040          MS      =       BIT05           ;SECTOR PULSE
2940            000010          MWP     =       BIT03           ;WRITE PROTECT
2941            000004          MI      =       BIT02           ;INDEX PULSE
2942            000002          MSC     =       BIT01           ;SECTOR COMPARE
2943            000001          DMD     =       BIT00           ;DIAGNOSTIC MODE
2944
2945                            ;       READ ONLY BITS
2946
2947            100000          OCC     =       BIT15           ;OCCUPIED
2948            040000          RG      =       BIT14           ;RUN AND GO
2949            020000          EBL     =       BIT13           ;END OF BLOCK
2950            010000          REX     =       BIT12           ;EXCEPTION
2951            004000          ESRC    =       BIT11           ;ENABLE SEARCH
2952            002000          PLFS    =       BIT10           ;LOOKING FOR SYNC
2953            001000          ECRC    =       BIT09           ;ENABLE CRC OUT
2954            000400          PDA     =       BIT08           ;DATA AREA
2955            000200          PHA     =       BIT07           ;HEADER AREA
2956            000100          CONT    =       BIT06           ;CONTINUE
2957            000040          WC      =       BIT05           ;WORD CLOCK
2958            000020          EECC    =       BIT04           ;ENABLE ECC OUT
2959            000010          MWD     =       BIT03           ;WRITE DATA BIT
2960            000004          LS      =       BIT02           ;LAST SECTOR
2961            000002          LST     =       BIT01           ;LAST SECTOR AND TRACK
2962            000001          DMD     =       BIT00           ;DIAGNOSTIC MODE
2963
2964                            ;RMDT   DRIVE TYPE REGISTER
2965
2966            100000          NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
2967            040000          TAP     =       BIT14           ;TAPE DRIVE = 0
2968            020000          MOH     =       BIT13           ;MOVING HEAD = 1
2969            004000          DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
2970
2971            020024          SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
2972            024024          DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
2973
2974                            ;RMOF   OFFSET REGISTER
2975
2976            010000          FMT16   =       BIT12           ;16 BIT WORD FORMAT
2977            004000          ECI     =       BIT11           ;ECC INHIBIT
2978            002000          HCI     =       BIT10           ;HEADER COMPARE INHIBIT
2979            000200          OFD     =       BIT07           ;OFFSET FORWARD
2980            161577          XNUOF   =       161577          ;UNUSED BITS OF RMOF
2981
2982
2983                            ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
2984            001777          CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
2985            176000          XNUDC   =       176000          ;UNUSED BITS OF RMDC
2986
```

```
                                ;RMMR2  MAINTENANCE REGISTER #2
                                ;
                                ;       READ ONLY BITS
100000                          RQA     =       BIT15           ;PORT A REQUEST
040000                          RQB     =       BIT14           ;PORT B REQUEST
020000                          TAG     =       BIT13           ;TAG CONTROL
010000                          TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
004000                          CC      =       BIT11           ;CONTROL OR CYLINDER TAG
002000                          CH      =       BIT10           ;CONTROL OR HEAD TAG
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

                                ;RMER2  ERROR REGISTER 2
100000                          BSE     =       BIT15           ;BAD SECTOR ERROR
040000                          SKI     =       BIT14           ;SEEK INCOMPLETE
020000                          OPE     =       BIT13           ;OPERATOR PLUG ERROR
010000                          IVC     =       BIT12           ;INVALID COMMAND ERROR
004000                          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
002000                          LBC     =       BIT10           ;LOSS OF BIT CLOCK
000200                          DVC     =       BIT07           ;DEVICE CHECK
000010                          DPE     =       BIT03           ;DATA PARITY ERROR
001567                          XNUER2  =       001567          ;UNUSED BITS OF RMER2

                                .SBTTL  PROGRAM MNEMONICS

100000                          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
040000                          USE     =       BIT14           ;USER DETECTED SECTOR ERROR

                                .SBTTL  RM03 REGISTER INDEX VALUES

000000                          RMCS1   =       00              ;CONTROL STATUS REGISTER
000006                          RMDA    =       06              ;DISK ADDRESS REGISTER
000012                          RMDS    =       12              ;DRIVE STATUS REGISTER
000014                          RMER1   =       14              ;ERROR REGISTER 1
000016                          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
000020                          RMLA    =       20              ;LOOK AHEAD REGISTER
000024                          RMMR1   =       24              ;MAINTENANCE REGISTER
000026                          RMDT    =       26              ;DRIVE TYPE REGISTER
000030                          RMSN    =       30              ;SERIAL NUMBER REGISTER
000032                          RMOF    =       32              ;OFFSET REGISTER
000034                          RMDC    =       34              ;DESIRED CYLINDER REGISTER
000036                          RMHR    =       36              ;HOLDING REGISTER
000040                          RMMR2   =       40              ;MAINTENANCE REGISTER 2
000042                          RMER2   =       42              ;ERROR REGISTER 2
000044                          RMEC1   =       44              ;ECC POSITION REGISTER
000046                          RMEC2   =       46              ;ECC PATTERN REGISTER
```

```
3043            000050                  ILRG50  =       50              ;ILLEGAL REGISTER 50
3044            000052                  ILRG52  =       52              ;ILLEGAL REGISTER 52
3045            000054                  ILRG54  =       54              ;ILLEGAL REGISTER 54
3046            000056                  ILRG56  =       56              ;ILLEGAL REGISTER 56
3047            000060                  ILRG60  =       60              ;ILLEGAL REGISTER 60
3048            000062                  ILRG62  =       62              ;ILLEGAL REGISTER 62
3049            000064                  ILRG64  =       64              ;ILLEGAL REGISTER 64
3050            000066                  ILRG66  =       66              ;ILLEGAL REGISTER 66
3051            000070                  ILRG70  =       70              ;ILLEGAL REGISTER 70
3052            000072                  ILRG72  =       72              ;ILLEGAL REGISTER 72
3053            000074                  ILRG74  =       74              ;ILLEGAL REGISTER 74
3054            000076                  ILRG76  =       76              ;ILLEGAL REGISTER 76
3055
3056
3057            000077                  IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
3058
3059                                    .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
3060
3061                                    ;RMCS1  CONTROL STATUS REGISTER #1
3062
3063            100000                  SC      =       BIT15           ;SPECIAL CONDITION-READ ONLY
3064            040000                  TRE     =       BIT14           ;TRANSFER ERROR
3065            020000                  MCPE    =       BIT13           ;MASSBUS CONTROL BUS PARITY
3066                                                                    ;ERROR-READ ONLY
3067            002000                  PSEL    =       BIT10           ;PORT B SELECT
3068            001000                  A17     =       BIT09           ;ADDRESS EXTENSION
3069            000400                  A16     =       BIT08           ;ADDRESS EXTENSION
3070            000200                  RDY     =       BIT07           ;READY-READ ONLY
3071            000100                  IE      =       BIT06           ;INTERRUPT ENABLE
3072
3073                                    ;RMCS2  RH CONTROL STATUS REGISTER #2
3074
3075            100000                  DLT     =       BIT15           ;DATA LATE-READ ONLY
3076            040000                  WCE     =       BIT14           ;WRITE CHECK ERROR-READ ONLY
3077            020000                  UPE     =       BIT13           ;UNIBUS PARITY ERROR
3078            010000                  NED     =       BIT12           ;NONEXISTANT DRIVE-READ ONLY
3079            004000                  NEM     =       BIT11           ;NONEXISTANT MEMORY-READ ONLY
3080            002000                  PGE     =       BIT10           ;PROGRAM ERROR-READ ONLY
3081            001000                  MXF     =       BIT09           ;MISSED TRANSFER
3082            000400                  MDPE    =       BIT08           ;MASSBUS DATA BUS PARITY
3083                                                                    ;ERROR-READ ONLY
3084            000200                  OR      =       BIT07           ;OUTPUT READY-READ ONLY
3085            000100                  IR      =       BIT06           ;INPUT READY-READ ONLY
3086            000040                  CLR     =       BIT05           ;CONTROLLER CLEAR
3087            000020                  PAT     =       BIT04           ;PARITY TEST
3088            000010                  BAI     =       BIT03           ;UNIBUS ADDRESS INCREMENT
3089                                                                    ;INHIBIT
3090            000004                  U2      =       BIT02           ;UNIT SELECT
3091            000002                  U1      =       BIT01           ;UNIT SELECT
3092            000001                  U0      =       BIT00           ;UNIT SELECT
3093
3094                                    ;UNIT SELECT MASK
3095            000007                  UNTMSK  =       7               ;UNIT SELECT MASK
3096
3097                                    ;RMCS3  RH70 CONTROL STATUS REGISTER #3
3098            100000                  APE     =       BIT15           ;ADDRESS PARITY ERROR
```

```
3099            040000                  DPEHI   =       BIT14           ;DATA PARITY ERROR HIGH WORD
3100            020000                  DPELO   =       BIT13           ;DATA PARITY ERROR LOW WORD
3101            010000                  WCEHI   =       BIT12           ;WRITE CHECK ERROR HIGH WORD
3102            004000                  WCELO   =       BIT11           ;WRITE CHECK ERROR LOW WORD
3103            002000                  DBL     =       BIT10           ;DOUBLE WORD TRANSFER
3104            000100                  IE      =       BIT06           ;INTERRUPT ENABLE
3105            000010                  IPCK3   =       BIT03           ;INVERT PARITY CHECK
3106            000004                  IPCK2   =       BIT02           ;INVERT PARITY CHECK
3107            000002                  IPCK1   =       BIT01           ;INVERT PARITY CHECK
3108            000001                  IPCK0   =       BIT00           ;INVERT PARITY CHECK
3109                                    .SBTTL  RH CONTROLLER REGISTER INDEX VALUES
3110
3111            000000                  RMCS1   =       00              ;CONTROL, STATUS REGISTER
3112            000002                  RMWC    =       02              ;WORD COUNT REGISTER
3113            000004                  RMBA    =       04              ;BUS ADDRESS REGISTER
3114            000010                  RMCS2   =       10              ;CONTROLLER STATUS REGISTER
3115            000022                  RMDB    =       22              ;DATA BUFFER
3116            000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
3117            000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3
3118
3119            176700                  ABASE   =       176700          ;UNIBUS ADDRESS
3120            120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY
3121
3122
3123                                    .SBTTL  TRAP CATCHER
3124
3125            000000                          .=0
3126                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
3127                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
3128                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
3129            000174                          .=174
3130    000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
3131    000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
3132
3133
3134                                    .SBTTL  ACT11 HOOKS
3135
3136                                    ;;*****************************************************************
3137                                    ;HOOKS REQUIRED BY ACT11
3138            000200                          $SVPC=.                 ;SAVE PC
3139            000046                          .=46
3140    000046  057274                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
3141            000052                          .=52
3142    000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
3143            000200                          .=$SVPC                 ;; RESTORE PC
3144
3145                                    .SBTTL  STARTING ADDRESS
3146
3147                                    ;THE PROGRAM STARTS AT LOCATION 200
3148            000200                          .       =       200
3149    000200  000137  004542                  JMP     START           ;JUMP TO START OF PROGRAM
3150
3151            001100                  .=1100
3152                                    .SBTTL  APT PARAMETER BLOCK
3153
3154                                    ;;*****************************************************************
```

```
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;****************************************************************
001100                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001100                          $APTHDR ;;POINT TO APT HEADER BLOCK
001100                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;****************************************************· *******
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-POP11 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  000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
001106  000002                  $PASTM: .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001110  000002                  $UNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001112  000042                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
001114                          TAGADR  =       .
```

```
                                .SBTTL  COMMON TAGS

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

                                        .=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  177607  000377           $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001216  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001217  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001220  000012                  $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
```

```
3231  001226  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
3232  001230  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
3233  001232  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
3234  001234  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
3235  001236  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
3236  001240  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
3237  001242                  $ETABLE:                ;;APT ENVIRONMENT TABLE
3238  001242     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
3239  001243     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
3240  001244  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
3241  001246  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
3242  001250  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
3243                          ;*                      BITS 15-11=CPU TYPE
3244                          ;*                             11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
3245                          ;*                             11/70=06,PDQ=07,Q=10
3246                          ;*                      BIT 10=REAL TIME CLOCK
3247                          ;*                      BIT  9=FLOATING POINT PROCESSOR
3248                          ;*                      BIT  8=MEMORY MANAGEMENT
3249  001252     000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
3250  001253     000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
3251                          ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
3252                          ;*                             900 NSEC CORE=001
3253                          ;*                             300 NSEC BIPOLAR=002
3254                          ;*                             500 NSEC MOS=003
3255  001254  000000          $MADR1:  WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
3256                          ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
3257  001256     000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
3258  001257     000          $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
3259  001260  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
3260  001262     000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
3261  001263     000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
3262  001264  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
3263  001266     000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
3264  001267     000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
3265  001270  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
3266  001272  120254          $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
3267  001274  000000          $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
3268  001276  176700          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
3269  001300  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
3270  001302  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
3271  001304  000000          $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
3272  001306  000000          $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
3273  001310  000000          $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
3274  001312  000000          $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
3275  001314  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
3276  001316  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
3277  001320  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
3278  001322  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
3279  001324  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
3280  001326                  $ETEND:
3281                          .MEXIT
3282
3283
3284
3285                          .SBTTL  REGISTER INPUT BUFFER
3286
```

```
3287                                    ;       THE REGISTER INPUT BUFFER IS USED FOR STORING REGISTER
3288                                    ;CONTENTS AS THEY ARE READ FROM THE DEVICE.
3289
3290  001326  000000                    RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER #1
3291  001330  000000                    RMWCI:  .WORD                   ;WORD COUNT REGISTER
3292  001332  000000                    RMBAI:  .WORD                   ;BUS ADDRESS REGISTER
3293  001334  000000                    RMDAI:  .WORD                   ;DISK ADDRESS REGISTER
3294  001336  000000                    RMCS2I: .WORD                   ;CONTROL, STATUS REGISTER #2
3295  001340  000000                    RMDSI:  .WORD                   ;DRIVE STATUS REGISTER
3296  001342  000000                    RMER1I: .WORD                   ;ERROR REGISTER 1
3297  001344  000000                    RMASI:  .WORD                   ;ATTENTION SUMMARY REGISTER
3298  001346  000000                    RMLAI:  .WORD                   ;LOOK AHEAD REGISTER
3299  001350  000000                    RMDBI:  .WORD                   ;DATA BUFFER
3300  001352  000000                    RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
3301  001354  000000                    RMDTI:  .WORD                   ;DRIVE TYPE REGISTER
3302  001356  000000                    RMSNI:  .WORD                   ;SERIAL NUMBER REGISTER
3303  001360  000000                    RMOFI:  .WORD                   ;OFFSET REGISTER
3304  001362  000000                    RMDCI:  .WORD                   ;DESIRED CYLINDER REGISTER
3305  001364  000000                    RMHRI:  .WORD                   ;HOLDING REGISTER
3306  001366  000000                    RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
3307  001370  000000                    RMER2I: .WORD                   ;ERROR REGISTER 2
3308  001372  000000                    RMEC1I: .WORD                   ;ECC POSITION REGISTER
3309  001374  000000                    RMEC2I: .WORD                   ;ECC PATTERN REGISTER
3310  001376  000000                    RMBAEI: .WORD           ;BUS ADDRESS EXTENSION REGISTER
3311  001400  000000                    RMCS3I: .WORD           ;CONTROL, STATUS REGISTER #3
3312
3313
3314
3315                                    .SBTTL  REGISTER OUTPUT BUFFER
3316
3317                                    ;       THE REGISTER OUTPUT BUFFER IS USED FOR ASSEMBLING DATA TO
3318                                    ;BE WRITTEN TO THE DEVICE.
3319
3320  001402  000000                    RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER #1
3321  001404  000000                    RMWCO:  .WORD                   ;WORD COUNT REGISTER
3322  001406  000000                    RMBAO:  .WORD                   ;BUS ADDRESS REGISTER
3323  001410  000000                    RMDAO:  .WORD                   ;DISK ADDRESS REGISTER
3324  001412  000000                    RMCS2O: .WORD                   ;CONTROL, STATUS REGISTER #2
3325  001414  000000                    RMDSO:  .WORD                   ;DRIVE STATUS REGISTER
3326  001416  000000                    RMER1O: .WORD                   ;ERROR REGISTER 1
3327  001420  000000                    RMASO:  .WORD                   ;ATTENTION SUMMARY REGISTER
3328  001422  000000                    RMLAO:  .WORD                   ;LOOK AHEAD REGISTER
3329  001424  000000                    RMDBO:  .WORD                   ;DATA BUFFER
3330  001426  000000                    RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
3331  001430  000000                    RMDTO:  .WORD                   ;DRIVE TYPE REGISTER
3332  001432  000000                    RMSNO:  .WORD                   ;SERIAL NUMBER REGISTER
3333  001434  000000                    RMOFO:  .WORD                   ;OFFSET REGISTER
3334  001436  000000                    RMDCO:  .WORD                   ;DESIRED CYLINDER REGISTER
3335  001440  000000                    RMHRO:  .WORD                   ;HOLDING REGISTER
3336  001442  000000                    RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
3337  001444  000000                    RMER2O: .WORD                   ;ERROR REGISTER 2
3338  001446  000000                    RMEC1O: .WORD                   ;ECC POSITION REGISTER
3339  001450  000000                    RMEC2O: .WORD                   ;ECC PATTERN REGISTER
3340  001452  000000                    RMBAEO: .WORD           ;BUS ADDRESS EXTENSION REGISTER
3341  001454  000000                    RMCS3O: .WORD           ;CONTROL, STATUS REGISTER #3
3342
```

```
                                .SBTTL  TEST QUE

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

001456  000012                TSTQUE: .BLKW   10.      ;TEST QUE

001502  172540                $LPCSR: .WORD   172540          ;KW11-P CONTROL + STATUS REGISTER
001504  172542                $LPCSB: .WORD   172542          ;KW11-P COUNT SET BUFFER
001506  000104                $LPVEC: .WORD   104             ;KW11-P INTERRUPT VECTOR
001510  000106                        .WORD   106
001512  177546                $LLCSR: .WORD   177546          ;KW11-L CONTROL + STATUS REGISTER
001514  000100                $LLVEC: .WORD   100             ;KW11-L INTERRUPT VECTOR
001516  000102                        .WORD   102
001520  000000                $PSW:   .WORD                   ;STORAGE FOR PRIORITY
001522  000000                TIME:   .WORD                   ;STORAGE FOR ELAPSED TIME
001524  000000                WATCH:  .WORD                   ;STORAGE FOR REMAINING TIME
001526  000000                CLOCK:  .WORD                   ;ADDRESS OF START CLOCK SUB
001530  000000                STOP:   .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

001532                          $ERRTB:
```

```
3383
3384                                    ;ERROR  1       CANNOT CLEAR NED STATUS
3385
3386  001532  066522                            EMT1
3387  001534  074400                            EHT1
3388  001536  074474                            EDT1
3389  001540  074522                            EFT1
3390
3391
3392                                    ;ERROR  2       CANNOT READ OR WRITE ANY DEVICE REG WITHOUT NED
3393
3394  001542  066530                            EMT2
3395  001544  074404                            EHT2
3396  001546  074476                            EDT2
3397  001550  074524                            EFT2
3398
3399
3400                                    ;ERROR  3       CANNOT WRITE/READ ONES TO ANY DEVICE REGISTER
3401
3402  001552  066556                            EMT3
3403  001554  000000                            0
3404  001556  000000                            0
3405  001560  000000                            0
3406
3407
3408                                    ;ERROR  4       CANNOT CLEAR ANY DEVICE REGISTER BITS W/MASSBUS INIT
3409
3410  001562  066576                            EMT4
3411  001564  000000                            0
3412  001566  000000                            0
3413  001570  000000                            0
3414
3415
3416                                    ;ERROR  5       CANNOT WRITE/READ ZEROS TO ALL BIT POSITIONS
3417
3418  001572  066620                            EMT5
3419  001574  074410                            EHT5
3420  001576  074500                            EDT5
3421  001600  074526                            EFT5
3422
3423
3424                                    ;ERROR  6       CANNOT WRITE/READ ONES TO ALL BIT POSITIONS
3425
3426  001602  066644                            EMT6
3427  001604  074410                            EHT5
3428  001606  074500                            EDT5
3429  001610  074526                            EFT5
3430
3431
3432                                    ;ERROR  7       CANNOT WRITE/READ SHIFTING ONE BIT TO ALL BIT POSITIONS
3433                                    ;               OF DEVICE REGISTERS
3434
3435  001612  066666                            EMT7
3436  001614  074414                            EHT7
3437  001616  074500                            EDT5
3438  001620  074526                            EFT5
```

```
3439
3440
3441                                    ;ERROR  10      REGISTER SELECT 1 APPEARS S-A-0
3442
3443  001622  066710                            EMT10
3444  001624  000000                            0
3445  001626  000000                            0
3446  001630  000000                            0
3447
3448
3449                                    ;ERROR  11      REGISTER SELECT 1 APPEARS S-A-1
3450
3451  001632  066726                            EMT11
3452  001634  000000                            0
3453  001636  000000                            0
3454  001640  000000                            0
3455
3456
3457                                    ;ERROR  12      REGISTER SELECT 2 APPEARS S-A-0
3458
3459  001642  066744                            EMT12
3460  001644  000000                            0
3461  001646  000000                            0
3462  001650  000000                            0
3463
3464
3465                                    ;ERROR  13      REGISTER SELECT 2 APPEARS S-A-1
3466
3467  001652  066762                            EMT13
3468  001654  000000                            0
3469  001656  000000                            0
3470  001660  000000                            0
3471
3472
3473                                    ;ERROR  14      REGISTER SELECT 4 APPEARS S-A-0
3474
3475  001662  067000                            EMT14
3476  001664  000000                            0
3477  001666  000000                            0
3478  001670  000000                            0
3479
3480
3481                                    ;ERROR  15      REGISTER SELECT 4 APPEARS S-A-1
3482
3483  001672  067016                            EMT15
3484  001674  000000                            0
3485  001676  000000                            0
3486  001700  000000                            0
3487
3488
3489                                    ;ERROR  16      REGISTER SELECT 8 APPEARS S-A-0
3490
3491  001702  067034                            EMT16
3492  001704  000000                            0
3493  001706  000000                            0
3494  001710  000000                            0
```

```
3495
3496
3497                                    ;ERROR  17      REGISTER SELECT 8 APPEARS S-A-1
3498
3499  001712  067052                              EMT17
3500  001714  000000                              0
3501  001716  000000                              0
3502  001720  000000                              0
3503
3504
3505                                    ;ERROR  20      CANT WRITE ZEROS RMDA
3506
3507  001722  067070                              EMT20
3508  001724  074400                              EHT1
3509  001726  074474                              EDT1
3510  001730  074522                              EFT1
3511
3512
3513                                    ;ERROR  21      CANT WRITE ONES RMDA
3514
3515  001732  067110                              EMT21
3516  001734  074400                              EHT1
3517  001736  074474                              EDT1
3518  001740  074522                              EFT1
3519
3520
3521                                    ;ERROR  22      BIT INTERFERENCE IN WRITING/READING RMDA
3522
3523  001742  067130                              EMT22
3524  001744  074400                              EHT1
3525  001746  074474                              EDT1
3526  001750  074522                              EFT1
3527
3528
3529                                    ;ERROR  23      CANT WRITE ZEROS RMCS1
3530
3531  001752  067144                              EMT23
3532  001754  074400                              EHT1
3533  001756  074474                              EDT1
3534  001760  074522                              EFT1
3535
3536
3537                                    ;ERROR  24      CANT WRITE ONES RMCS1
3538
3539  001762  067164                              EMT24
3540  001764  074400                              EHT1
3541  001766  074474                              EDT1
3542  001770  074522                              EFT1
3543
3544
3545                                    ;ERROR  25      BIT INTERFERENCE IN WRITING/READING RMCS1
3546
3547  001772  067204                              EMT25
3548  001774  074400                              EHT1
3549  001776  074474                              EDT1
3550  002000  074522                              EFT1
```

```
3551
3552
3553                                    ;ERROR  26      MBA CLR L IS STUCK ACTIVE
3554
3555   002002  067220                           EMT26
3556   002004  000000                           0
3557   002006  000000                           0
3558   002010  000000                           0
3559
3560
3561                                    ;ERROR  27      CANNOT CLEAR RMER1-PAR,RMR,ILF,ILR
3562
3563   002012  067252                           EMT27
3564   002014  074400                           EHT1
3565   002016  074474                           EDT1
3566   002020  074522                           EFT1
3567
3568
3569                                    ;ERROR  30      CANNOT CLEAR RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
3570
3571   002022  067264                           EMT30
3572   002024  074400                           EHT1
3573   002026  074474                           EDT1
3574   002030  074522                           EFT1
3575
3576
3577                                    ;ERROR  31      CANNOT CLEAR RMER1-OPI,DTE
3578
3579   002032  067300                           EMT31
3580   002034  074400                           EHT1
3581   002036  074474                           EDT1
3582   002040  074522                           EFT1
3583
3584
3585                                    ;ERROR  32      CANNOT WRITE 0 IN RMER1-PAR,RMR,ILF,ILR
3586
3587   002042  067314                           EMT32
3588   002044  074400                           EHT1
3589   002046  074474                           EDT1
3590   002050  074522                           EFT1
3591
3592
3593                                    ;ERROR  33      CANNOT WRITE 0 IN RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
3594
3595   002052  067330                           EMT33
3596   002054  074400                           EHT1
3597   002056  074474                           EDT1
3598   002060  074522                           EFT1
3599
3600
3601                                    ;ERROR  34      CANNOT WRITE 0 IN RMER1-OPI,DTE
3602
3603   002062  067346                           EMT34
3604   002064  074400                           EHT1
3605   002066  074474                           EDT1
3606   002070  074522                           EFT1
```

```
3607
3608
3609                                    ;ERROR  35      CANNOT WRITE 1 IN RMER1
3610
3611  002072  067364                            EMT35
3612  002074  074400                            EHT1
3613  002076  074474                            EDT1
3614  002100  074522                            EFT1
3615
3616
3617                                    ;ERROR  36      CANNOT WRITE SHIFTING 1 IN RMER1
3618
3619  002102  067400                            EMT36
3620  002104  074400                            EHT1
3621  002106  074474                            EDT1
3622  002110  074522                            EFT1
3623
3624
3625                                    ;ERROR  37      CANNOT WRITE ZEROS IN RMDC
3626
3627  002112  067414                            EMT37
3628  002114  074400                            EHT1
3629  002116  074474                            EDT1
3630  002120  074522                            EFT1
3631
3632
3633                                    ;ERROR  40      CANNOT WRITE ONES IN RMDC
3634
3635  002122  067430                            EMT40
3636  002124  074400                            EHT1
3637  002126  074474                            EDT1
3638  002130  074522                            EFT1
3639
3640
3641                                    ;ERROR  41      BIT INTERFERENCE IN WRITING/READING RMDC
3642
3643  002132  067450                            EMT41
3644  002134  074400                            EHT1
3645  002136  074474                            EDT1
3646  002140  074522                            EFT1
3647
3648
3649                                    ;ERROR  42      CANNOT WRITE 1'S IN RMDC OR RMDA
3650
3651  002142  067464                            EMT42
3652  002144  000000                            0
3653  002146  000000                            0
3654  002150  000000                            0
3655
3656
3657                                    ;ERROR  43      CANNOT CLEAR RMCS1-FUNCTION CODE
3658
3659  002152  067510                            EMT43
3660  002154  074400                            EHT1
3661  002156  074474                            EDT1
3662  002160  074522                            EFT1
```

```
3663
3664
3665                                    ;ERROR  44      UNUSED BITS OF RMER2 NOT ZERO
3666
3667  002162  067522                            EMT44
3668  002164  074400                            EHT1
3669  002166  074474                            EDT1
3670  002170  074522                            EFT1
3671
3672
3673                                    ;ERROR  45      CANNOT CLEAR RMER2-OPE,IVC,LSC
3674
3675  002172  067536                            EMT45
3676  002174  074400                            EHT1
3677  002176  074474                            EDT1
3678  002200  074522                            EFT1
3679
3680
3681                                    ;ERROR  46      CANNOT CLEAR RMER2-LBC,DPE
3682
3683  002202  067552                            EMT46
3684  002204  074400                            EHT1
3685  002206  074474                            EDT1
3686  002210  074522                            EFT1
3687
3688
3689                                    ;ERROR  47      CANNOT WRITE ZEROS RMER2-OPE,IVC,LSC
3690
3691  002212  067566                            EMT47
3692  002214  074400                            EHT1
3693  002216  074474                            EDT1
3694  002220  074522                            EFT1
3695
3696
3697                                    ;ERROR  50      CANNOT WRITE ZEROS RMER2-LBC,DPE
3698
3699  002222  067610                            EMT50
3700  002224  074400                            EHT1
3701  002226  074474                            EDT1
3702  002230  074522                            EFT1
3703
3704
3705                                    ;ERROR  51      CANNOT WRITE ONES RMER2
3706
3707  002232  067632                            EMT51
3708  002234  074400                            EHT1
3709  002236  074474                            EDT1
3710  002240  074522                            EFT1
3711
3712
3713                                    ;ERROR  52      CANNOT WRITE SHIFTING ONES RMER2
3714
3715  002242  067652                            EMT52
3716  002244  074400                            EHT1
3717  002246  074474                            EDT1
3718  002250  074522                            EFT1
```

```
3719
3720
3721                                    ;ERROR  53      UNUSED BITS OF RMOF ARE NOT ZERO
3722
3723  002252  067666                              EMT53
3724  002254  074400                              EHT1
3725  002256  074474                              EDT1
3726  002260  074522                              EFT1
3727
3728
3729                                    ;ERROR  54      CANNOT WRITE ZEROS RMOF-FMT,ECI,HCI,OFD
3730
3731  002262  067702                              EMT54
3732  002264  074400                              EHT1
3733  002266  074474                              EDT1
3734  002270  074522                              EFT1
3735
3736
3737                                    ;ERROR  55      CANNOT WRITE ONES RMOF-FMT,ECI,HCI,OFD
3738
3739  002272  067722                              EMT55
3740  002274  074400                              EHT1
3741  002276  074474                              EDT1
3742  002300  074522                              EFT1
3743
3744
3745                                    ;ERROR  56      CANNOT WRITE SHIFTING ONES RMOF
3746
3747  002302  067742                              EMT56
3748  002304  074400                              EHT1
3749  002306  074474                              EDT1
3750  002310  074522                              EFT1
3751
3752
3753                                    ;ERROR  57      DEVICE IS NOT AN RM03
3754
3755  002312  067756                              EMT57
3756  002314  074420                              EHT57
3757  002316  074502                              EDT57
3758  002320  074530                              EFT57
3759
3760
3761                                    ;ERROR  60      DEVICE AVAILABLE IS NOT SET
3762
3763  002322  067772                              EMT60
3764  002324  074400                              EHT1
3765  002326  074474                              EDT1
3766  002330  074522                              EFT1
3767
3768
3769                                    ;ERROR  61      CANNOT WRITE ZEROS RMHR
3770
3771  002332  070006                              EMT61
3772  002334  074400                              EHT1
3773  002336  074474                              EDT1
3774  002340  074522                              EFT1
```

```
3775
3776
3777                                    ;ERROR  62      CANNOT WRITE ONES RMHR
3778
3779  002342  070026                            EMT62
3780  002344  074400                            EHT1
3781  002346  074474                            EDT1
3782  002350  074522                            EFT1
3783
3784
3785                                    ;ERROR  63      CANNOT WRITE SHIFTING ONES RMHR
3786
3787  002352  070046                            EMT63
3788  002354  074400                            EHT1
3789  002356  074474                            EDT1
3790  002360  074522                            EFT1
3791
3792
3793                                    ;ERROR  64      CANNOT CLEAR ILR STATUS
3794
3795  002362  070062                            EMT64
3796  002364  074400                            EHT1
3797  002366  074474                            EDT1
3798  002370  074522                            EFT1
3799
3800
3801                                    ;ERROR  65      ILR ERROR SHOULD NOT BE SET
3802
3803  002372  070074                            EMT65
3804  002374  074424                            EHT65
3805  002376  074504                            EDT65
3806  002400  074532                            EFT65
3807
3808
3809                                    ;ERROR  66      ILR ERROR SHOULD BE SET
3810
3811  002402  070110                            EMT66
3812  002404  074424                            EHT65
3813  002406  074504                            EDT65
3814  002410  074532                            EFT65
3815
3816
3817                                    ;ERROR  67      CANNOT CLEAR PAR STATUS-DPE IS RESET
3818
3819  002412  070124                            EMT67
3820  002414  074400                            EHT1
3821  002416  074474                            EDT1
3822  002420  074522                            EFT1
3823
3824
3825                                    ;ERROR  70      CANNOT CLEAR PAR AND DPE STATUS
3826
3827  002422  070142                            EMT70
3828  002424  074400                            EHT1
3829  002426  074474                            EDT1
3830  002430  074522                            EFT1
```

```
3831
3832
3833                                    ;ERROR  71      "PAR" ERROR SHOULD NOT BE SET-"PAT" IS OFF
3834
3835  002432  070162                            EMT71
3836  002434  074430                            EHT71
3837  002436  074506                            EDT71
3838  002440  074534                            EFT71
3839
3840
3841                                    ;ERROR  72      "PAR" ERROR SHOULD BE SET-"PAT" IS ON
3842
3843  002442  070206                            EMT72
3844  002444  074430                            EHT71
3845  002446  074506                            EDT71
3846  002450  074534                            EFT71
3847
3848
3849                                    ;ERROR  73      "MCPE" ERROR SHOULD NOT BE SET
3850
3851  002452  070232                            EMT73
3852  002454  074430                            EHT71
3853  002456  074506                            EDT71
3854  002460  074534                            EFT71
3855
3856
3857                                    ;ERROR  74      UNEXPECTED BUS TIMEOUT
3858
3859  002462  070252                            EMT74
3860  002464  074434                            EHT74
3861  002466  074510                            EDT74
3862  002470  074536                            EFT74
3863
3864
3865                                    ;ERROR  75      CANT CLEAR "DMD"
3866
3867  002472  070262                            EMT75
3868  002474  074400                            EHT1
3869  002476  074474                            EDT1
3870  002500  074522                            EFT1
3871
3872
3873                                    ;ERROR  76      CANT WRITE ZERO "DMD"
3874
3875  002502  070274                            EMT76
3876  002504  074400                            EHT1
3877  002506  074474                            EDT1
3878  002510  074522                            EFT1
3879
3880
3881                                    ;ERROR  77      CANT WRITE ONE "DMD"
3882
3883  002512  070310                            EMT77
3884  002514  074400                            EHT1
3885  002516  074474                            EDT1
3886  002520  074522                            EFT1
```

```
3887
3888
3889                                    ;ERROR  100     DMD SET BY WRONG BIT
3890
3891  002522  070324                            EMT100
3892  002524  074430                            EHT71
3893  002526  074504                            EDT65
3894  002530  074532                            EFT65
3895
3896
3897                                    ;ERROR  101     CANT CLEAR "MOL" IN DIAGNOSTIC MODE
3898
3899  002532  070344                            EMT101
3900  002534  074400                            EHT1
3901  002536  074474                            EDT1
3902  002540  074522                            EFT1
3903
3904
3905                                    ;ERROR  102     CANT SET "MOL" IN DIAGNOSTIC MODE
3906
3907  002542  070364                            EMT102
3908  002544  074400                            EHT1
3909  002546  074474                            EDT1
3910  002550  074522                            EFT1
3911
3912
3913                                    ;ERROR  103     "MUR" SET BY WRONG BIT
3914
3915  002552  070404                            EMT103
3916  002554  074430                            EHT71
3917  002556  074504                            EDT65
3918  002560  074532                            EFT65
3919
3920
3921                                    ;ERROR  104     CANT RESET "WRL" IN DIAGNOSTIC MODE
3922
3923  002562  070430                            EMT104
3924  002564  074400                            EHT1
3925  002566  074474                            EDT1
3926  002570  074522                            EFT1
3927
3928
3929                                    ;ERROR  105     CANT SET "WRL" IN DIAGNOSTIC MODE
3930
3931  002572  070450                            EMT105
3932  002574  074400                            EHT1
3933  002576  074474                            EDT1
3934  002600  074522                            EFT1
3935
3936
3937                                    ;ERROR  106     "MWP" SET BY WRONG BIT
3938
3939  002602  070470                            EMT106
3940  002604  074430                            EHT71
3941  002606  074504                            EDT65
3942  002610  074532                            EFT65
```

```
3943
3944
3945                                    ;ERROR  107     CANT RESET "DVC" USING "MDVC"
3946
3947  002612  070514                              EMT107
3948  002614  074400                              EHT1
3949  002616  074474                              EDT1
3950  002620  074522                              EFT1
3951
3952
3953                                    ;ERROR  110     "DVC" IS RESET BUT "UNS" IS SET
3954
3955  002622  070534                              EMT110
3956  002624  074400                              EHT1
3957  002626  074474                              EDT1
3958  002630  074522                              EFT1
3959
3960
3961                                    ;ERROR  111     "DVC" IS SET BUT "UNS" IS NOT SET
3962
3963  002632  070556                              EMT111
3964  002634  074400                              EHT1
3965  002636  074474                              EDT1
3966  002640  074522                              EFT1
3967
3968
3969                                    ;ERROR  112     CANT SET "DVC" USING MDVC"
3970
3971  002642  070576                              EMT112
3972  002644  074400                              EHT1
3973  002646  074474                              EDT1
3974  002650  074522                              EFT1
3975
3976
3977                                    ;ERROR  113     "DVC" IS RESET BUT "UNS" IS SET
3978
3979  002652  070616                              EMT113
3980  002654  074400                              EHT1
3981  002656  074474                              EDT1
3982  002660  074522                              EFT1
3983
3984
3985                                    ;ERROR  114     "DVC" IS SET BUT "UNS" IS NOT SET
3986
3987  002662  070642                              EMT114
3988  002664  074400                              EHT1
3989  002666  074474                              EDT1
3990  002670  074522                              EFT1
3991
3992
3993                                    ;ERROR  115     "MDF" IS SET BY WRONG BIT
3994
3995  002672  070664                              EMT115
3996  002674  074440                              EHT115
3997  002676  074512                              EDT115
3998  002700  074540                              EFT115
```

```
3999
4000
4001                                    ;ERROR  116     CANT RESET "SKI" USING "MSER"
4002
4003  002702  070710                            EMT116
4004  002704  074400                            EHT1
4005  002706  074474                            EDT1
4006  002710  074522                            EFT1
4007
4008
4009                                    ;ERROR  117     CANT SET "SKI" USING "MSER"
4010
4011  002712  070730                            EMT117
4012  002714  074400                            EHT1
4013  002716  074474                            EDT1
4014  002720  074522                            EFT1
4015
4016
4017                                    ;ERROR  120     "SKI" SET BY WRONG BIT
4018
4019  002722  070750                            EMT120
4020  002724  074440                            EHT115
4021  002726  074512                            EDT115
4022  002730  074540                            EFT115
4023
4024
4025                                    ;ERROR  121     CANT RESET "PIP" USING "MOC"
4026
4027  002732  070774                            EMT121
4028  002734  074400                            EHT1
4029  002736  074474                            EDT1
4030  002740  074522                            EFT1
4031
4032
4033                                    ;ERROR  122     CANT SET "PIP" USING "MOC"
4034
4035  002742  071014                            EMT122
4036  002744  074400                            EHT1
4037  002746  074474                            EDT1
4038  002750  074522                            EFT1
4039
4040
4041                                    ;ERROR  123     "MOC" SET BY WRONG BIT
4042
4043  002752  071034                            EMT123
4044  002754  074440                            EHT115
4045  002756  074512                            EDT115
4046  002760  074540                            EFT115
4047
4048
4049                                    ;ERROR  124     CANT CLEAR "EBL"
4050
4051  002762  071060                            EMT124
4052  002764  074400                            EHT1
4053  002766  074474                            EDT1
4054  002770  074522                            EFT1
```

```
4055
4056
4057                                    ;ERROR  125     "EBL" NOT ZERO IN DIAGNOSTIC MODE
4058
4059  002772  071076                            EMT125
4060  002774  074400                            EHT1
4061  002776  074474                            EDT1
4062  003000  074522                            EFT1
4063
4064
4065                                    ;ERROR  126     CANT SET "EBL" USING "DEBL"
4066
4067  003002  071116                            EMT126
4068  003004  074400                            EHT1
4069  003006  074474                            EDT1
4070  003010  074522                            EFT1
4071
4072
4073                                    ;ERROR  127     "DEBL" SET BY WRONG BIT
4074
4075  003012  071134                            EMT127
4076  003014  074440                            EHT115
4077  003016  074512                            EDT115
4078  003020  074540                            EFT115
4079
4080
4081                                    ;ERROR  130     "LS" NOT CORRECT ACCORDING TO RMDA
4082
4083  003022  071160                            EMT130
4084  003024  074444                            EHT130
4085  003026  074514                            EDT130
4086  003030  074542                            EFT130
4087
4088
4089                                    ;ERROP  131     "LST" NOT CORRECT ACCORDING TO RMDA
4090
4091  003032  071176                            EMT131
4092  003034  074444                            EHT130
4093  003036  074514                            EDT130
4094  003040  074542                            EFT130
4095
4096
4097                                    ;ERROR  132     CANNOT INCREMENT SECTOR ADDRESS USING "DEBL"
4098
4099  003042  071214                            EMT132
4100  003044  074450                            EHT132
4101  003046  074516                            EDT132
4102  003050  074544                            EFT132
4103
4104
4105                                    ;ERROR  133     CANNOT INCREMENT TRACK ADDRESS USING "DEBL"
4106
4107  003052  071234                            EMT133
4108  003054  074450                            EHT132
4109  003056  074516                            EDT132
4110  003060  074544                            EFT132
```

```
4111
4112
4113                                    ;ERROR  134     UNUSED BITS OF RMDC NOT ZERO
4114
4115  003062  071254                              EMT134
4116  003064  074400                              EHT1
4117  003066  074474                              EDT1
4118  003070  074522                              EFT1
4119
4120
4121                                    ;ERROR  135     "VV" NOT RESET BY UNIT READY
4122
4123  003072  071270                              EMT135
4124  003074  074400                              EHT1
4125  003076  074474                              EDT1
4126  003100  074522                              EFT1
4127
4128
4129                                    ;ERROR  136     SERIAL NUMBER IS INCONSISTENT
4130
4131  003102  071306                              EMT136
4132  003104  074400                              EHT1
4133  003106  074474                              EDT1
4134  003110  074522                              EFT1
4135
4136
4137                                    ;ERROR  137     CANT CLEAR "GO" BIT
4138
4139  003112  071320                              EMT137
4140  003114  074400                              EHT1
4141  003116  074474                              EDT1
4142  003120  074522                              EFT1
4143
4144
4145                                    ;ERROR  140     CANT INCREMENT CYLINDER USING "DEBL"
4146
4147  003122  071336                              EMT140
4148  003124  074450                              EHT132
4149  003126  074516                              EDT132
4150  003130  074544                              EFT132
4151
4152
4153                                    ;ERROR  141     CANT RESET "LBT" BY WRITING RMDC
4154
4155  003132  071356                              EMT141
4156  003134  074400                              EHT1
4157  003136  074474                              EDT1
4158  003140  074522                              EFT1
4159
4160
4161                                    ;ERROR  142     CANT SET "LBT" USING "DEBL"
4162
4163  003142  071372                              EMT142
4164  003144  074400                              EHT1
4165  003146  074474                              EDT1
4166  003150  074522                              EFT1
```

```
                                ;ERROR  143     CANT READ ZERO FROM COMP ERROR

003152  071410                          EMT143
003154  074400                          EHT1
003156  074474                          EDT1
003160  074522                          EFT1


                                ;ERROR  144     CANT SET COMP ERROR WITH RMER1 OR RMER2

003162  071424                          EMT144
003164  074400                          EHT1
003166  074474                          EDT1
003170  074522                          EFT1


                                ;ERROR  145     COMP ERROR DID NOT SET

003172  071446                          EMT145
003174  074454                          EHT145
003176  074514                          EDT130
003200  074542                          EFT130


                                ;ERROR  146     CANT SET "GO" BIT

003202  071470                          EMT146
003204  074400                          EHT1
003206  074474                          EDT1
003210  074522                          EFT1


                                ;ERROR  147     CANT READ A ONE FROM "TST"

003212  071506                          EMT147
003214  074400                          EHT1
003216  074474                          EDT1
003220  074522                          EFT1


                                ;ERROR  150     "TST" IS INCORRECT FOR THE FUNCTION CODE

003222  071520                          EMT150
003224  074460                          EHT150
003226  074512                          EDT115
003230  074540                          EFT115


                                ;ERROR  151     CANT SET THE "GO" BIT

003232  071542                          EMT151
003234  074400                          EHT1
003236  074474                          EDT1
003240  074522                          EFT1
```

```
4223
4224
4225                                    ;ERROR  152     "DRY" NOT THE COMPLEMENT OF "GO"
4226
4227  003242  071554                            EMT152
4228  003244  074400                            EHT1
4229  003246  074474                            EDT1
4230  003250  074522                            EFT1
4231
4232
4233                                    ;ERROR  153     "GO" RESET EARLY
4234
4235  003252  071570                            EMT153
4236  003254  074400                            EHT1
4237  003256  074474                            EDT1
4238  003260  074522                            EFT1
4239
4240
4241                                    ;ERROR  154     "GO" DIDNT RESET ON TIME
4242
4243  003262  071610                            EMT154
4244  003264  074400                            EHT1
4245  003266  074474                            EDT1
4246  003270  074522                            EFT1
4247
4248
4249                                    ;ERROR  155     CANT CLEAR CONTINUE
4250
4251  003272  071630                            EMT155
4252  003274  074400                            EHT1
4253  003276  074474                            EDT1
4254  003300  074522                            EFT1
4255
4256
4257                                    ;ERROR  156     CONTINUE IS INCORRECT FOR THE FUNCTION CODE
4258
4259  003302  071646                            EMT156
4260  003304  074460                            EHT150
4261  003306  074512                            EDT115
4262  003310  074540                            EFT115
4263
4264
4265                                    ;ERROR  157     CANT CLEAR IVC
4266
4267  003312  071670                            EMT157
4268  003314  074400                            EHT1
4269  003316  074474                            EDT1
4270  003320  074522                            EFT1
4271
4272
4273                                    ;ERROR  160     IVC IS INCORRECT FOR THE FUNCTION CODE
4274
4275  003322  071706                            EMT160
4276  003324  074460                            EHT150
4277  003326  074512                            EDT115
4278  003330  074540                            EFT115
```

```
4279
4280
4281                                    ;ERROR  161     CANT CLEAR LSC
4282
4283  003332  071736                            EMT161
4284  003334  074400                            EHT1
4285  003336  074474                            EDT1
4286  003340  074522                            EFT1
4287
4288
4289                                    ;ERROR  162     CANT SET LSC
4290
4291  003342  071754                            EMT162
4292  003344  074400                            EHT1
4293  003346  074474                            EDT1
4294  003350  074522                            EFT1
4295
4296
4297                                    ;ERROR  163     COMMAND DECODE WAS ENABLED WITH COMP ERROR SET
4298
4299  003352  071772                            EMT163
4300  003354  074400                            EHT1
4301  003356  074474                            EDT1
4302  003360  074522                            EFT1
4303
4304
4305                                    ;ERROR  164     COMMAND DECODE WAS ENABLED WITH COMP ERROR SET
4306
4307  003362  072014                            EMT164
4308  003364  074400                            EHT1
4309  003366  074474                            EDT1
4310  003370  074522                            EFT1
4311
4312
4313                                    ;ERROR  165     DECODE DOES NOT SET
4314
4315  003372  072036                            EMT165
4316  003374  000000                            0
4317  003376  000000                            0
4318  003400  000000                            0
4319
4320
4321                                    ;ERROR  166     CANT CLEAR OCCUPIED
4322
4323  003402  072062                            EMT166
4324  003404  074400                            EHT1
4325  003406  074474                            EDT1
4326  003410  074522                            EFT1
4327
4328
4329                                    ;ERROR  167     ILF SET WITHOUT GO BIT
4330
4331  003412  072102                            EMT167
4332  003414  074400                            EHT1
4333  003416  074474                            EDT1
4334  003420  074522                            EFT1
```

```
                                ;ERROR  170     CANT SET VOLUME VALID

003422  072124                          EMT170
003424  074460                          EHT150
003426  074512                          EDT115
003430  074540                          EFT115

                                ;ERROR  171     ILF IS INCORRECT

003432  072136                          EMT171
003434  074460                          EHT150
003436  074512                          EDT115
003440  074540                          EFT115

                                ;ERROR  172     CANT SET OFFSET DIRECTION BIT

003442  072152                          EMT172
003444  074400                          EHT1
003446  074474                          EDT1
003450  074522                          EFT1

                                ;ERROR  173     OCCUPIED IS INCORRECT FOR FUNCTION CODE

003452  072170                          EMT173
003454  074460                          EHT150
003456  074512                          EDT115
003460  074540                          EFT115

                                ;ERROR  174     READ IN PRESET DIDNT CLEAR RMDA, RMDC OR RMOF

003462  072212                          EMT174
003464  000000                          0
003466  000000                          0
003470  000000                          0

                                ;ERROR  175     READ IN PRESET DIDNT CLEAR RMOF

003472  072240                          EMT175
003474  074400                          EHT1
003476  074474                          EDT1
003500  074522                          EFT1

                                ;ERROR  176     READ IN PRESET DIDNT CLEAR RMDA

003502  072256                          EMT176
003504  074400                          EHT1
003506  074474                          EDT1
003510  074522                          EFT1
```

```
4391
4392
4393                                    ;ERROR  177     READ IN PRESET DIDNT CLEAR RMDC
4394
4395  003512  072274                            EMT177
4396  003514  074400                            EHT1
4397  003516  074474                            EDT1
4398  003520  074522                            EFT1
4399
4400
4401                                    ;ERROR  200     CANT SET OFFSET MODE BY OFFSET COMMAND
4402
4403  003522  072312                            EMT200
4404  003524  074400                            EHT1
4405  003526  074474                            EDT1
4406  003530  074522                            EFT1
4407
4408
4409                                    ;ERROR  201     CANT RESET OFFSET MODE BY RTC COMMAND
4410
4411  003532  072330                            EMT201
4412  003534  074400                            EHT1
4413  003536  074474                            EDT1
4414  003540  074522                            EFT1
4415
4416
4417                                    ;ERROR  202     CANT RESET OFD BY RTC COMMAND
4418
4419  003542  072346                            EMT202
4420  003544  074400                            EHT1
4421  003546  074474                            EDT1
4422  003550  074522                            EFT1
4423
4424
4425                                    ;ERROR  203     CANT RESET OM BY RMDC
4426
4427  003552  072364                            EMT203
4428  003554  074400                            EHT1
4429  003556  074474                            EDT1
4430  003560  074522                            EFT1
4431
4432
4433                                    ;ERROR  204     CANT RESET OM BY EBL
4434
4435  003562  072406                            EMT204
4436  003564  074400                            EHT1
4437  003566  074474                            EDT1
4438  003570  074522                            EFT1
4439
4440
4441                                    ;ERROR  205     RUN AND GO NOT CORRECT FOR FUNCTION CODE
4442
4443  003572  072426                            EMT205
4444  003574  074460                            EHT150
4445  003576  074512                            EDT115
4446  003600  074540                            EFT115
```

```
                                ;ERROR  206     CANT SET IAE ERROR

003602  072446                          EMT206
003604  000000                          0
003606  000000                          0
003610  000000                          0

                                ;ERROR  207     IAE IS INCORRECT FOR FUNCTION CODE

003612  072476                          EMT207
003614  074460                          EHT150
003616  074512                          EDT115
003620  074540                          EFT115

                                ;ERROR  210     IAE IS INCORRECT FOR RMDA

003622  072512                          EMT210
003624  074440                          EHT115
003626  074512                          EDT115
003630  074540                          EFT115

                                ;ERROR  211     IAE IS INCORRECT FOR RMDC

003632  072530                          EMT211
003634  074440                          EHT115
003636  074512                          EDT115
003640  074540                          EFT115

                                ;ERROR  212     CANT SET AOE

003642  072546                          EMT212
003644  074400                          EHT1
003646  074474                          EDT1
003650  074522                          EFT1

                                ;ERROR  213     RMR SET WHEN WRITING RMAS OR RMCS

003652  072560                          EMT213
003654  074464                          EHT213
003656  074512                          EDT115
003660  074540                          EFT115

                                ;ERROR  214     CANT SET RMR

003662  072610                          EMT214
003664  074464                          EHT213
003666  074512                          EDT115
003670  074540                          EFT115
```

```
4503
4504
4505                                    ;ERROR  215     DRQ IS 0 AND PGM IS 1
4506
4507  003672  072622                            EMT215
4508  003674  074400                            EHT1
4509  003676  074474                            EDT1
4510  003700  074522                            EFT1
4511
4512
4513                                    ;ERROR  216     DVA IS NOT SET
4514
4515  003702  072642                            EMT216
4516  003704  074400                            EHT1
4517  003706  074474                            EDT1
4518  003710  074522                            EFT1
4519
4520
4521                                    ;ERROR  217     DPR IS NOT SET
4522
4523  003712  072656                            EMT217
4524  003714  074400                            EHT1
4525  003716  074474                            EDT1
4526  003720  074522                            EFT1
4527
4528
4529                                    ;ERROR  220     CANT SET PORT REQUEST BY READING RMCS1
4530
4531  003722  072672                            EMT220
4532  003724  074470                            EHT220
4533  003726  074520                            EDT220
4534  003730  074546                            EFT220
4535
4536
4537                                    ;ERROR  221     CANT SET PORT REQUEST BY WRITING RMAS
4538
4539  003732  072710                            EMT221
4540  003734  074470                            EHT220
4541  003736  074520                            EDT220
4542  003740  074546                            EFT220
4543
4544
4545                                    ;ERROR  222     CANT SET PORT REQUEST BY WRITING RMDA
4546
4547  003742  072726                            EMT222
4548  003744  074470                            EHT220
4549  003746  074520                            EDT220
4550  003750  074546                            EFT220
4551
4552
4553                                    ;ERROR  223     CANT RESET PORT REQUEST BY RELEASE COMMAND
4554
4555  003752  072744                            EMT223
4556  003754  074470                            EHT220
4557  003756  074520                            EDT220
4558  003760  074546                            EFT220
```

```
4559
4560
4561                                    ;ERROR  224     CANT CLEAR ATA BY RMAS
4562
4563  003762  072762                            EMT224
4564  003764  074400                            EHT1
4565  003766  074474                            EDT1
4566  003770  074522                            EFT1
4567
4568
4569                                    ;ERROR  225     ATA IS RESET BUT RMAS NOT ZERO
4570
4571  003772  073002                            EMT225
4572  003774  074400                            EHT1
4573  003776  074474                            EDT1
4574  004000  074522                            EFT1
4575
4576
4577                                    ;ERROR  226     CANT RESET ATA BY GO
4578
4579  004002  073024                            EMT226
4580  004004  074400                            EHT1
4581  004006  074474                            EDT1
4582  004010  074522                            EFT1
4583
4584
4585                                    ;ERROR  227     ATA NOT SET BY UNIT READY
4586
4587  004012  073042                            EMT227
4588  004014  074400                            EHT1
4589  004016  074474                            EDT1
4590  004020  074522                            EFT1
4591
4592
4593                                    ;ERROR  230     ATA NOT SET BY UNIT READY
4594
4595  004022  073060                            EMT230
4596  004024  074400                            EHT1
4597  004026  074474                            EDT1
4598  004030  074522                            EFT1
4599
4600
4601                                    ;ERROR  231     ATA NOT SET BY COMP ERROR
4602
4603  004032  073076                            EMT231
4604  004034  074400                            EHT1
4605  004036  074474                            EDT1
4606  004040  074522                            EFT1
4607
4608
4609                                    ;ERROR  232     ATA SET/DID NOT SET WHEN REGISTER WRITTEN
4610                                    ;               WHILE COMP ERROR WAS SET
4611
4612  004042  073112                            EMT232
4613  004044  074464                            EHT213
4614  004046  074512                            EDT115
```

```
4615  004050  074540                        EFT115
4616
4617
4618                                ;ERROR  233     ATA NOT SET BY COMMAND SEQUENCER
4619
4620  004052  073134                        EMT233
4621  004054  074460                        EHT150
4622  004056  074512                        EDT115
4623  004060  074540                        EFT115
4624
4625
4626                                ;ERROR  234     WLE INCORRECT ACCORDING TO FUNCTION CODE
4627
4628  004062  073156                        EMT234
4629  004064  074460                        EHT150
4630  004066  074512                        EDT115
4631  004070  074540                        EFT115
4632
4633
4634                                ;ERROR  235     CANT CLEAR EXCEPTION
4635
4636  004072  073204                        EMT235
4637  004074  074400                        EHT1
4638  004076  074474                        EDT1
4639  004100  074522                        EFT1
4640
4641
4642                                ;ERROR  236     CANT SET EXCEPTION
4643
4644  004102  073224                        EMT236
4645  004104  074440                        EHT115
4646  004106  074512                        EDT115
4647  004110  074540                        EFT115
4648
4649
4650                                ;ERROR  237     CANT CLEAR OPI
4651
4652  004112  073256                        EMT237
4653  004114  074400                        EHT1
4654  004116  074474                        EDT1
4655  004120  074522                        EFT1
4656
4657
4658                                ;ERROR  240     CANT SET OPI
4659
4660  004122  073256                        EMT237
4661  004124  074460                        EHT150
4662  004126  074512                        EDT115
4663  004130  074540                        EFT115
4664
4665
4666                                ;ERROR  241     OPI NOT SET DURING RECALIBRATE
4667
4668  004132  073322                        EMT241
4669  004134  074400                        EHT1
4670  004136  074474                        EDT1
```

```
4671  004140  074522                        EFT1
4672
4673
4674                                ;ERROR  242     RECALIBRATE DID NOT ABORT WHEN DRIVE FAULT SET
4675
4676  004142  073346                        EMT242
4677  004144  074400                        EHT1
4678  004146  074474                        EDT1
4679  004150  074522                        EFT1
4680
4681
4682                                ;ERROR  243     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
4683                                ;               DROPPED DURING RECALIBRATE
4684
4685  004152  073372                        EMT243
4686  004154  074400                        EHT1
4687  004156  074474                        EDT1
4688  004160  074522                        EFT1
4689
4690
4691                                ;ERROR  244     ATA NOT SET DURING RECALIBRATE
4692
4693  004162  073416                        EMT244
4694  004164  074400                        EHT1
4695  004166  074474                        EDT1
4696  004170  074522                        EFT1
4697
4698
4699                                ;ERROR  245     GO RESET EARLY DURING RECALIBRATE
4700
4701  004172  073434                        EMT245
4702  004174  074400                        EHT1
4703  004176  074474                        EDT1
4704  004200  074522                        EFT1
4705
4706
4707                                ;ERROR  246     GO NOT RESET DURING RECALIBRATE
4708
4709  004202  073452                        EMT246
4710  004204  074400                        EHT1
4711  004206  074474                        EDT1
4712  004210  074522                        EFT1
4713
4714
4715                                ;ERROR  247     INCORRECT TAG BUS DURING RECALIBRATE
4716
4717  004212  073476                        EMT247
4718  004214  074400                        EHT1
4719  004216  074474                        EDT1
4720  004220  074522                        EFT1
4721
4722
4723                                ;ERROR  250     OPI SHOULD HAVE SET DURING SEEK BECAUSE UNIT
4724                                ;               READY DROPPED
4725
4726  004222  073514                        EMT250
```

```
 4727  004224  074400                    EHT1
 4728  004226  074474                    EDT1
 4729  004230  074522                    EFT1
 4730
 4731
 4732                            ;ERROR  251     SEEK DID NOT ABORT WHEN DRIVE FAULT SET
 4733
 4734  004232  073540                    EMT251
 4735  004234  074400                    EHT1
 4736  004236  074474                    EDT1
 4737  004240  074522                    EFT1
 4738
 4739
 4740                            ;ERROR  252     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
 4741                            ;               DROPPED DURING SEEK
 4742
 4743  004242  073564                    EMT252
 4744  004244  074400                    EHT1
 4745  004246  074474                    EDT1
 4746  004250  074522                    EFT1
 4747
 4748
 4749                            ;ERROR  253     ATA NOT SET DURING SEEK
 4750
 4751  004252  073610                    EMT253
 4752  004254  074400                    EHT1
 4753  004256  074474                    EDT1
 4754  004260  074522                    EFT1
 4755
 4756
 4757                            ;ERROR  254     GO RESET EARLY DURING SEEK
 4758
 4759  004262  073626                    EMT254
 4760  004264  074400                    EHT1
 4761  004266  074474                    EDT1
 4762  004270  074522                    EFT1
 4763
 4764
 4765                            ;ERROR  255     GO DID NOT RESET DURING SEEK
 4766
 4767  004272  073644                    EMT255
 4768  004274  074400                    EHT1
 4769  004276  074474                    EDT1
 4770  004300  074522                    EFT1
 4771
 4772
 4773                            ;ERROR  256     INCORRECT TAG BUS DURING SEEK
 4774
 4775  004302  073670                    EMT256
 4776  004304  074400                    EHT1
 4777  004306  074474                    EDT1
 4778  004310  074522                    EFT1
 4779
 4780
 4781                            ;ERROR  257     OPI NOT SET DURING SEARCH
 4782
```

```
4783  004312  073706                  EMT257
4784  004314  074400                  EHT1
4785  004316  074474                  EDT1
4786  004320  074522                  EFT1
4787
4788
4789                          ;ERROR  260     SEARCH DID NOT ABORT WHEN DRIVE FAULT SET
4790
4791  004322  073732                  EMT260
4792  004324  074400                  EHT1
4793  004326  074474                  EDT1
4794  004330  074522                  EFT1
4795
4796
4797                          ;ERROR  261     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
4798                          ;               DROPPED DURING SEARCH
4799
4800  004332  073756                  EMT261
4801  004334  074400                  EHT1
4802  004336  074474                  EDT1
4803  004340  074522                  EFT1
4804
4805
4806                          ;ERROR  262     ATA NOT SET DURING SEARCH
4807
4808  004342  074002                  EMT262
4809  004344  074400                  EHT1
4810  004346  074474                  EDT1
4811  004350  074522                  EFT1
4812
4813
4814                          ;ERROR  263     GO RESET EARLY DURING SEARCH
4815
4816  004352  074020                  EMT263
4817  004354  074400                  EHT1
4818  004356  074474                  EDT1
4819  004360  074522                  EFT1
4820
4821
4822                          ;ERROR  264     GO DID NOT RESET DURING SEARCH
4823
4824  004362  074036                  EMT264
4825  004364  074400                  EHT1
4826  004366  074474                  EDT1
4827  004370  074522                  EFT1
4828
4829
4830                          ;ERROR  265     SEARCH ENABLE DIDNT SET DURING SEARCH
4831
4832  004372  074062                  EMT265
4833  004374  074400                  EHT1
4834  004376  074474                  EDT1
4835  004400  074522                  EFT1
4836
4837
4838                          ;ERROR  266     INCORRECT TAG BUS DURING SEARCH
```

```
4839
4840  004402  074100                    EMT266
4841  004404  074400                    EHT1
4842  004406  074474                    EDT1
4843  004410  074522                    EFT1
4844
4845
4846                            ;ERROR  267     OPI NOT SET BY SEARCH TIMEOUT
4847
4848  004412  074116                    EMT267
4849  004414  074400                    EHT1
4850  004416  074474                    EDT1
4851  004420  074522                    EFT1
4852
4853
4854                            ;ERROR  270     OPI NOT SET DURING DATA COMMAND
4855
4856  004422  074132                    EMT270
4857  004424  074400                    EHT1
4858  004426  074474                    EDT1
4859  004430  074522                    EFT1
4860
4861
4862                            ;ERROR  271     DATA COMMAND DID NOT ABORT WHEN DRIVE FAULT SET
4863
4864  004432  074156                    EMT271
4865  004434  074400                    EHT1
4866  004436  074474                    EDT1
4867  004440  074522                    EFT1
4868
4869
4870                            ;ERROR  272     EBL RESET EARLY DURING DATA COMMAND
4871
4872  004442  074202                    EMT272
4873  004444  074400                    EHT1
4874  004446  074474                    EDT1
4875  004450  074522                    EFT1
4876
4877
4878                            ;ERROR  273     EBL DIDNT RESET ON TIME DURING DATA COMMAND
4879
4880  004452  074220                    EMT273
4881  004454  074400                    EHT1
4882  004456  074474                    EDT1
4883  004460  074522                    EFT1
4884
4885
4886                            ;ERROR  274     GO NOT RESET DURING DATA COMMAND
4887
4888  004462  074236                    EMT274
4889  004464  074400                    EHT1
4890  004466  074474                    EDT1
4891  004470  074522                    EFT1
4892
4893
4894                            ;ERROR  275     RUN AND GO NOT SET DURING DATA COMMAND
```

```
4895
4896  004472  074254                    EMT275
4897  004474  074400                    EHT1
4898  004476  074474                    EDT1
4899  004500  074522                    EFT1
4900
4901
4902                              ;ERROR  276     INCORRECT TAG BUS DURING DATA COMMAND
4903
4904  004502  074274                    EMT276
4905  004504  074400                    EHT1
4906  004506  074474                    EDT1
4907  004510  074522                    EFT1
4908
4909
4910                              ;ERROR  277     OPI NOT SET DURING DATA COMMAND WHEN ON
4911                              ;               CYLINDER DIDNT DROP
4912
4913  004512  074312                    EMT277
4914  004514  074400                    EHT1
4915  004516  074474                    EDT1
4916  004520  074522                    EFT1
4917
4918
4919                              ;ERROR  300     DATA COMMAND DID NOT ABORT WHEN SEEK ERROR SET
4920
4921  004522  074336                    EMT300
4922  004524  074400                    EHT1
4923  004526  074474                    EDT1
4924  004530  074522                    EFT1
4925
4926
4927                              ;ERROR  301     SEARCH NOT ENABLED DURING DATA COMMAND
4928
4929  004532  074362                    EMT301
4930  004534  074400                    EHT1
4931  004536  074474                    EDT1
4932  004540  074522                    EFT1
4933
4934
4935                              ;PUT ERROR TABLE HERE
4936
```

```
                                .SBTTL  START OF PROGRAM

004542                          START:

                                ;CLEAR AND SETUP FOR TEST EXECUTION
004542  000240                          NOP
004544  000005                          RESET                   ;INITIALIZE THE SYSTEM

                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004546  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004552  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004554  022706  001154                  CMP     #SWR,R6 ;;DONE?
004560  001374                          BNE     .-6             ;;LOOP BACK IF NO
004562  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
004566  012737  062064  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
004574  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
004602  012737  062640  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004610  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004616  012737  064400  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004624  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
004632  012737  064506  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
004640  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
004646  013737  057140  057132          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
004654  005037  001206                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
004660  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
004664  112737  000001  001131          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
004672  012737  004672  001122          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
004700  012737  004700  001124          MOV     #.,$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.
004706  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
004712  012737  004746  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
004720  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
004726  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004734  022777  177777  174212          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
004742  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
004744  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
004746  012716  004754          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
004752  000002                          RTI
004754  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004762  012737  000174  001156          MOV     #DISPREG,DISPLAY
004770  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

004774  005037  001230                  CLR     $PASS           ;;CLEAR PASS COUNT
005000  132737  000200  001243          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005006  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
005010  012737  001244  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005016                          67$:
005016  012746  000140                  MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
005022  012746  005030                  MOV     #68$,-(SP)      ;;PUT NEW PC ON  STACK
005026  000002                          RTI                     ;;POP NEW PC AND PS
005030                          68$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005030  005227  177777                          INC     #-1             ;;FIRST TIME?
005034  001060                                  BNE     69$             ;;BRANCH IF NO
005036  022737  057274  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
005044  001454                                  BEQ     69$             ;;BRANCH IF YES
005046  104401  005114                          TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005052  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005056  001012                                  BNE     71$             ;;BRANCH IF YES
005060  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005066  001406                                  BEQ     71$             ;;BRANCH IF YES
005070  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005076  001005                                  BNE     72$             ;;BRANCH IF NO
005100  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
005102  000403                                  BR      72$
005104  112737  000001  001150  71$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005112                          72$:
005112  000431                                  BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@MD-11-DZRMJ-A, RM03 DISKLESS DIAGNOSTIC PROGRAM@<CRLF>
005176                          69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005176  005737  000042                          TST     42              ;IS LOC 42 ZERO ??
005202  001003                                  BNE     10$             ;NO - NOT IN STANDALONE
005204  105737  001242                          TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
005210  001411                                  BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
005212                          10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
005212  132737  000200  001243                  BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
005220  001003                                  BNE     20$             ;NO
005222  012737  000377  001300                  MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
005230                          20$:


                                        ;GO TO COMMON START CODE
005230  000137  006052                          JMP     CMNSTART
```

```
5031  005234                                  STANDALONE:
5032
5033  005234  004737  063050                          JSR     PC,$TKINT
5034                                          ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
5035  005240  005327  000001                          DEC     #1              ;FIRST START ??
5036  005244  100024                                  BPL     10$             ;YES !!
5037
5038
5039                                          ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
5040  005246  104401  065272                  5$:     TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
5041  005252  104411                                  RDCHR                   ;GET RESPONSE
5042  005254  012637  001176                          MOV     (SP)+,$TMP1     ;ECHO RESPONSE
5043  005260  104401  001176                          TYPE    ,$TMP1
5044  005264  123727  001176  000131                  CMPB    $TMP1,#'Y       ;YES RESPONSE??
5045  005272  001002                                  BNE     6$              ;NO!!
5046  005274  000137  006212                          JMP     READY           ;KEEP PREVIOUS PARAMETERS
5047  005300  123727  001176  000116          6$:     CMPB    $TMP1,#'N       ;NO RESPONSE??
5048  005306  001420                                  BEQ     20$             ;GET NEW PARAMETERS
5049  005310  104401  065141                          TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
5050  005314  000754                                  BR      5$              ;RETRY
5051  005316                                  10$:
5052
5053                                          ;SEE IF OPERATOR WANTS HELP FILE
5054  005316  104401  065143                          TYPE    ,HELPQST        ;WANT HELP ??
5055  005322  104411                                  RDCHR                   ;GET RESPONSE
5056  005324  012637  001176                          MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
5057  005330  104401  001176                          TYPE    ,$TMP1
5058  005334  123727  001176  000131                  CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
5059  005342  001002                                  BNE     20$             ;NO - DONT TYPE HELP
5060  005344  104401  106652                          TYPE    ,HELP           ;YES - TYPE HELP TEXT
5061  005350                                  20$:
5062
5063                                          ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
5064  005350  104401  065177                          TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
5065  005354  104411                                  RDCHR                   ;GET RESPONSE
5066  005356  012637  001176                          MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
5067  005362  104401  001176                          TYPE    ,$TMP1
5068  005366  123727  001176  000131                  CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
5069  005374  001137                                  BNE     90$             ;NO !!
5070  005376                                  30$:
5071
5072                                          ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
5073  005376  104401  065331                          TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
5074  005402  013746  001276                          MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
5075  005406  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5076  005410  104401  065132                          TYPE    ,CLSPRN
5077  005414  104413                                  RDOCT                   ;GET NEW BUS ADDRESS
5078  005416  012637  001176                          MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
5079  005422  001412                                  BEQ     50$             ;YES-SKIP TO NEXT ENTRY
5080  005424  022737  160000  001176                  CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
5081  005432  101403                                  BLOS    40$             ;YES
5082  005434  104401  065356                          TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
5083  005440  000756                                  BR      30$             ;RETRY
5084  005442  013737  001176  001276          40$:    MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
5085  005450  113737  001272  001176          50$:    MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
5086  005456  105037  001177                          CLRB    $TMP1+1
```

```
 5087  005462  104401  065437                    TYPE    ,CNSL03
 5088  005466  013746  001176                    MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
 5089  005472  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
 5090  005474     003                            .BYTE   3               ;;TYPE 3 DIGIT(S)
 5091  005475     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
 5092  005476  104401  065132                    TYPE    ,CLSPRN
 5093  005502  104413                            RDOCT                   ;GET NEW VECTOR ADDRESS
 5094  005504  012637  001176                    MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
 5095  005510  001412                            BEQ     70$             ;YES-SKIP TO NEXT ENTRY
 5096  005512  022737  001000  001176            CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
 5097  005520  101003                            BHI     60$             ;YES!!
 5098  005522  104401  065467                    TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
 5099  005526  000750                            BR      50$             ;RETRY
 5100  005530  113737  001176  001272  60$:     MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
 5101  005536  113737  001273  001176  70$:     MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
 5102  005544  006237  001176                    ASR     $TMP1
 5103  005550  006237  001176                    ASR     $TMP1
 5104  005554  006237  001176                    ASR     $TMP1
 5105  005560  006237  001176                    ASR     $TMP1
 5106  005564  006237  001176                    ASR     $TMP1
 5107  005570  105237  001177                    CLRB    $TMP1+1
 5108  005574  104401  065543                    TYPE    ,CNSL05
 5109  005600  013746  001176                    MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
 5110  005604  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
 5111  005606     001                            .BYTE   1               ;;TYPE 1 DIGIT(S)
 5112  005607     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
 5113  005610  104401  065132                    TYPE    ,CLSPRN
 5114  005614  104413                            RDOCT                   ;GET NEW PRIORITY
 5115  005616  012637  001176                    MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
 5116  005622  001424                            BEQ     90$             ;YES-SKIP TO NEXT ENTRY
 5117  005624  023727  001176  000007            CMP     $TMP1,#7        ;LEGAL PRIORITY??
 5118  005632  002403                            BLT     80$             ;YES!!
 5119  005634  104401  065577                    TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
 5120  005640  000736                            BR      70$             ;RETRY
 5121  005642                          80$:                              ;STORE NEW PRIORITY
 5122  005642  006337  001176                    ASL     $TMP1
 5123  005646  006337  001176                    ASL     $TMP1
 5124  005652  006337  001176                    ASL     $TMP1
 5125  005656  006337  001176                    ASL     $TMP1
 5126  005662  006337  001176                    ASL     $TMP1
 5127  005666  113737  001176  001273            MOVB    $TMP1,$VECT1+1
 5128  005674                          90$:
 5129
 5130                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
 5131  005674  005037  001300                    CLR     $DEVM           ;CLEAR DEVICE MAP
 5132  005700  104401  065624                    TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
 5133  005704  104401  065135                    TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
 5134  005710  104411                            RDCHR                   ;GET RESPONSE
 5135  005712  012637  001176                    MOV     (SP)+,$TMP1     ;ECHO RESPONSE
 5136  005716  104401  001176                    TYPE    ,$TMP1
 5137  005722  023727  001176  000101            CMP     $TMP1,#'A       ;TEST ALL DRIVES??
 5138  005730  001017                            BNE     110$            ;NO
 5139  005732  012737  000377  001300            MOV     #377,$DEVM      ;TEST ALL DEVICES
 5140  005740  000444                            BR      140$            ;SKIP TO NEXT ENTRY
 5141  005742  104401  065135          100$:     TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
 5142  005746  104411                            RDCHR                   ;GET RESPONSE
```

```
 5143  005750  012637  001176              MOV     (SP)+,$TMP1     ;ECHO RESPONSE
 5144  005754  104401  001176              TYPE    ,$TMP1
 5145  005760  023727  001176  000015      CMP     $TMP1,#CR       ;CARRIAGE RETURN??
 5146  005766  001431                      BEQ     140$
 5147  005770  023727  001176  000060  110$: CMP     $TMP1,#'0       ;NUMBER < 0??
 5148  005776  002404                      BLT     120$            ;YES!!
 5149  006000  023727  001176  000067      CMP     $TMP1,#'7       ;NUMBER > 7??
 5150  006006  003403                      BLE     130$            ;NO!!
 5151  006010  104401  065141          120$: TYPE    ,QSTMRK         ;TYPE "?"
 5152  006014  000752                      BR      100$            ;RETRY
 5153  006016  013701  001176          130$: MOV     $TMP1,R1        ;R1=DRIVE NUMBER
 5154  006022  042701  177770              BIC     #^C7,R1
 5155  006026  116102  066064              MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
 5156  006032  042702  177400              BIC     #^C377,R2               ;CLEAR UNUSED BITS
 5157  006036  050237  001300              BIS     R2,$DEVM                ;SET DEVICE # IN MAP
 5158  006042  122737  000377  001300      CMPB    #377,$DEVM      ;DONE ??
 5159  006050  101334                      BHI     100$            ;NO
 5160  006052                          140$:
 5161
```

```
5162  006052                                  CMNSTART:
5163
5164                                          ;ASSEMBLE TEST QUE FROM DEVICE MAP
5165  006052  013700  001300                  10$:   MOV     $DEVM,R0          ;R0 = DEVICE MAP
5166  006056  012701  001460                         MOV     #TSTQUE+2,R1      ;R1 = ADDRESS OF FIRST ENTRY IN QUE
5167  006062  010137  001456                         MOV     R1,TSTQUE         ;INITIALIZE ENTRY POINTER
5168  006066  012702  000001                         MOV     #1,R2             ;R2 = DEVICE POINTER
5169  006072  005003                               CLR     R3                ;R3 = DEVICE NUMBER
5170  006074  030200                               BIT     R2,R0             ;IS THIS DEVICE IN MAP ??
5171  006076  001406                               BEQ     20$               ;NO !!
5172  006100  010311                               MOV     R3,(R1)           ;YES - ENTER DEVICE NUMBER IN QUE
5173  006102  116361  066064  000001               MOVB    ATNTBL(R3),1(R1)  ;ENTER ATTENTION BIT IN QUE
5174  006110  062701  000002                       ADD     #2,R1             ;ADVANCE ENTRY POINTER
5175  006114  006302                         20$:  ASL     R2                ;ADVANCE DEVICE POINTER
5176  006116  105702                               TSTB    R2                ;DONE ALL DEVICES ??
5177  006120  001402                               BEQ     25$               ;YES
5178  006122  005203                               INC     R3                ;ADVANCE DEVICE NUMBER
5179  006124  000763                               BR      10$               ;ENTER NEXT DEVICE
5180  006126  005011                         25$:  CLR     (R1)              ;TERMINATE TEST QUE
5181
5182                                          ;SIZE FOR CLOCK
5183  006130  004737  060116                       JSR     PC,SIZCLK         ;SEE IF CLOCK PRESENT
5184  006134  000403                               BR      40$               ;YES - CLOCK IS PRESENT
5185  006136  104000                         30$:  ERROR                     ;NO CLOCK
5186  006140  000000                               HALT
5187  006142  000775                               BR      30$
5188  006144                                  40$:
5189                                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
5190  006144  005737  000042                       TST     @#42              ;;ARE WE RUNNING UNDER XXDP/ACT?
5191  006150  001012                               BNE     64$               ;;BRANCH IF YES
5192  006152  123727  001242  000001               CMPB    $ENV,#1           ;;ARE WE RUNNING UNDER APT?
5193  006160  001406                               BEQ     64$               ;;BRANCH IF YES
5194  006162  023727  001154  000176               CMP     SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
5195  006170  001005                               BNE     65$               ;;BRANCH IF NO
5196  006172  104407                               GTSWR                     ;;GET SOFT-SWR SETTINGS
5197  006174  000403                               BR      65$
5198  006176  112737  000001  001150  64$:   MOVB    #1,$AUTOB         ;;SET AUTO-MODE INDICATOR
5199  006204                                  65$:
5200  006204  012737  000140  000032               MOV     #PR3,@#EMTVEC+2   ;DROP PRIORITY DURING ERROR TYPEOUT
5201  006212  000240                         READY: NOP                     ;READY TO START TEST
5202  006214  004737  063050                       JSR     PC,$TKINT         ;INITIALIZE TTY
5203  006220  117737  173232  001234               MOVB    @TSTQUE,$UNIT     ;LOAD UNIT NUMBER
```

```
5204                            .SBTTL  REGISTER AND STORAGE USAGE
5205
5206                            ;REGISTER ASSIGNMENTS
5207                            ;
5208                            ;R0     =       UNIBUS ADDRESS OF RH CONTROLLER
5209                            ;R1     =       ADDRESS OF ENTRY IN TEST QUE CORRESPONDING TO THE
5210                            ;               UNIT UNDER TEST
5211                            ;R2,R3  =       WORKING REGISTERS FOR TEST IN PROGRESS, MUST BE
5212                            ;               SAVED BY SUBROUTINES
5213                            ;R4,R5  =       GENERAL WORKING REGISTERS, ARE NOT SAVED BY
5214                            ;               SUBROUTINES
5215                            ;R6     =       STACK POINTER
5216                            ;R7     =       LINKAGE REGISTER TO SUBROUTINES
5217
5218                            ;STORAGE ASSIGNMENTS
5219                            ;
5220                            ;$TMP0-$TMP4    TEMPORARY STORAGE, NOT SAVED BY SUBROUTINES
5221                            ;$GDDAT,$BDDAT  EXPECTED AND RRECEIVED STATUS FOR ERROR TYPEOUT
5222                            ;$GDADR,$BDADR  ADDRESS OF EXPECTED AND RECEIVED STATUS IF APPLICABLE,
5223                            ;               ALSO THE ADDRESS OF A REGISTER ERROR
5224                            ;
5225                            ;$TSTN  =       TEST NUMBER
5226                            ;$UNIT  =       NUMBER OF DEVICE BEING TESTED
5227                            ;RGINBF =       THE REGISTER INPUT BUFFER HAS A STORAGE LOCATION FOR
5228                            ;               EACH REGISTER, AND IS USED WHEN READING STATUS AND
5229                            ;               CONTROL DATA
5230                            ;RGOTBF =       THE REGISTER OUTPUT BUFFER HAS A STORAGE LOCATION FOR
5231                            ;               EACH REGISTER, AND IS USED FOR ASSEMBLING DATA TO BE
5232                            ;               WRITTEN IN REGISTERS
```

```
                                ;;*****************************************************************
                                ;*TEST 1          TRANSFER TEST
                                ;;*****************************************************************
006226  000004                  $ST1:   SCOPE
006230  012737  000001  001226          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
006236  000240                          NOP
006240  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
006246  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
006254  012737  006270  001122          MOV     #T1,$LPADR  ;LOAD LOOP ON TEST ADDRESS
006262  012737  006270  001124          MOV     #T1,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
006270                          T1:
006270  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
006274  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
006300  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
006304  012702  000000                  MOV     #0,R2           ;R2 = REGISTER INDEX
006310                          10$:
                                ;CLEAR THE MASSBUS AND VERIFY THAT NONEXISTANT DEVICE ERROR IS RESET
006310  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
006316  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

006322  016037  000010  001142          MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
006330  032737  010000  001142          BIT     #NED,$BDDAT
006336  001417                          BEQ     20$
006340  111137  001140                  MOVB    (R1),$GDDAT
006344  042737  177770  001140          BIC     #^CUNTMSK,$GDDAT
006352  052737  000100  001140          BIS     #IR,$GDDAT
006360  010037  001136                  MOV     R0,$BDADR
006364  062737  000010  001136          ADD     #RMCS2,$BDADR
006372  104001                          ERROR   1
006374  000500                          BR      60$
006376                          20$:
                                ;READ THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE READ
                                ;DOES NOT SET "NED" ERROR
006376  010003                          MOV     R0,R3           ;R3 = REGISTER ADDRESS
006400  060203                          ADD     R2,R3
006402  011304                          MOV     (R3), R4                ;READ REGISTER
006404  032760  010000  000010          BIT     #NED, RMCS2(R0)         ;IS "NED" SET??
006412  001473                          BEQ     70$             ;NO!!
006414  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
006422  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

006426  016037  000010  001142          MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
006434  032737  010000  001142          BIT     #NED,$BDDAT
006442  001417                          BEQ     30$
006444  111137  001140                  MOVB    (R1),$GDDAT
006450  042737  177770  001140          BIC     #^CUNTMSK,$GDDAT
006456  052737  000100  001140          BIS     #IR,$GDDAT
006464  010037  001136                  MOV     R0,$BDADR
006470  062737  000010  001136          ADD     #RMCS2,$BDADR
006476  104001                          ERROR   1
006500  000436                          BR      60$
006502                          30$:
                                ;WRITE THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE WRITE
                                ;DOES NOT SET "NED" ERROR
```

```
5289  006502  012713  000000                  MOV     #0,(R3)         ;WRITE REGISTER
5290  006506  032760  010000  000010          BIT     #NED, RMCS2     (R0)    ;IS "NED" SET??
5291  006514  001432                          BEQ     70$             ;NO!!
5292
5293  006516                          40$:
5294                                  ;COULD NOT READ OR WRITE THE REGISTER WITHOUT SETING "NED" ERROR -
5295                                  ;ADVANCE THE REGISTER INDEX AND REPEAT THE TEST FOR THE NEXT
5296                                  ;AVAILABLE DEVICE REGISTER
5297  006516  062702  000002                  ADD     #2,R2           ;ADVANCE TO NEXT REGISTER
5298  006522  022702  000002                  CMP     #RMWC,R2                ;IS THIS RMWC??
5299  006526  001773                          BEQ     40$             ;YES - TRY NEXT REGISTER
5300  006530  022702  000004                  CMP     #RMBA,R2                ;IS THIS RMBA??
5301  006534  001770                          BEQ     40$             ;YES - TRY NEXT REGISTER
5302  006536  022702  000010                  CMP     #RMCS2,R2               ;IS THIS RMCS5??
5303  006542  001765                          BEQ     40$             ;YES - TRY ANOTHER REGISTER
5304  006544  022702  000016                  CMP     #RMAS,R2                ;IS THIS RMAS ??
5305  006550  001762                          BEQ     40$             ;YES - TRY ANOTHER REGISTER
5306  006552  022702  000022                  CMP     #RMDB,R2                ;IS THIS RMDB??
5307  006556  001757                          BEQ     40$             ;YES - TRY ANOTHER REGISTER
5308  006560  022702  000046                  CMP     #RMEC2,R2               ;IS THIS A LEGAL REGISTER
5309  006564  103251                          BHIS    10$             ;YES - TRY THIS REGISTER
5310
5311  006566                          50$:
5312
5313                                  ;GOT 'NONEXISTENT DEVICE' ERROR FOR EVERY REMOTE REGISTER ADDRESS
5314  006566  013737  001276  001136          MOV     $BASE, $BOADR           ;STORE BASE ADDRESS
5315  006574  104002                          ERROR   2               ;DEMAND OR TRANSFER FAILED
5316  006576  000137  057050          60$:    JMP     $EOSP           ;GO SELECT NEXT DEVICE
5317
5318  006602                          70$:
5319
5320                                  ;;*****************************************************************
5321                                  ;*TEST 2        CTOD TEST
5322
5323                                  ;;*****************************************************************
5324  006602  000004                  TST2:   SCOPE
5325  006604  012737  000002  001226          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5326  006612  000240                          NOP
5327  006614  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
5328  006622  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
5329  006630  012737  006644  001122          MOV     #T2,$LPADR  ;LOAD LOOP ON TEST ADDRESS
5330  006636  012737  006644  001124          MOV     #T2,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
5331  006644                          T2:
5332  006644  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
5333  006650  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
5334  006654  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
5335  006660  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5336  006666  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5337                                  ;WRITE ONES IN REMOTE REGISTERS
5338
5339  006672  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
5340
5341  006700  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
5342
5343  006706  012760  001777  000034          MOV     #CYLMSK,RMDC(R0)        ;LOAD RMDC
5344
```

```
5345  006714  012760  016200  000032          MOV     #FMT16!ECI!HCI!OFD,RMOF(R0)     ;LOAD RMOF
5346                                  ;READ REMOTE REGISTERS TWICE
5347  006722  012702  000001                  MOV     #1,R2
5348  006726                          10$:
5349
5350  006726  016037  000000  001326          MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
5351
5352  006734  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5353
5354  006742  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5355
5356  006750  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5357  006756  005302                          DEC     R2
5358  006760  100362                          BPL     10$
5359                                  ;SEE IF ANY ONE BITS CAME BACK
5360  006762  042737  177701  001326          BIC     #^CILF76,RMCS1I ;IS RMCS1 0??
5361  006770  001014                          BNE     20$             ;NO!!
5362  006772  005737  001334                  TST     RMDAI           ;IS RMDA 0??
5363  006776  001011                          BNE     20$             ;NO!!
5364  007000  042737  176000  001362          BIC     #XNUDC,RMDCI    ;IS RMDC 0??
5365  007006  001005                          BNE     20$             ;NO!!
5366  007010  042737  161577  001360          BIC     #XNUOF,RMOFI    ;IS RMOF 0 ??
5367  007016  001001                          BNE     20$             ;NO!!
5368                                  ;CANNOT READ ANY ONE BITS FROM REMOTE REGISTERS
5369  007020  104003                          ERROR   3               ;CTOD MUST BE STUCK
5370  007022                          20$:
5371
5372                                  ;;*****************************************************************
5373                                  ;*TEST 3        MASSBUS INITIALIZE TEST
5374
5375                                  ;;*****************************************************************
5376  007022  000004                  TST3:   SCOPE
5377  007024  012737  000003  001226          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5378
5379  007032  000240                          NOP
5380  007034  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
5381  007042  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
5382  007050  012737  007064  001122          MOV     #T3,$LPADR  ;LOAD LOOP ON TEST ADDRESS
5383  007056  012737  007064  001124          MOV     #T3,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
5384  007064                          T3:
5385  007064  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
5386  007070  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
5387  007074  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
5388  007100  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5389  007106  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5390                                  ;WRITE ONES IN SELECTED REGISTERS
5391
5392  007112  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
5393
5394  007120  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
5395
5396  007126  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
5397                                  ;USING CONTROLLER CLEAR, IE., BIT 5 OF RMCS2, INITIALIZE THE MASSBUS
5398  007134  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5399  007142  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5400                                  ;READ THE REGISTERS THAT WERE WRITTEN
```

```
5401
5402  007146  016037  000000  001326          MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
5403
5404  007154  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5405
5406  007162  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5407                                  ;SEE IF ANY REGISTER BITS WERE CLEARED
5408  007170  052737  177701  001326          BIS     #^C1LF76,RMCS1I         ;SET ANY BIT NOT WRITTEN
5409  007176  052737  001567  001370          BIS     #XNUER2,RMER2I
5410  007204  022737  177777  001326          CMP     #-1,RMCS1I              ;ANY ZEROS IN RMCS1??
5411  007212  001011                          BNE     10$             ;YES!!
5412  007214  022737  177777  001342          CMP     #-1,RMER1I              ;ANY ZEROS IN RMER1??
5413  007222  001005                          BNE     10$             ;YES!!
5414  007224  022737  177777  001370          CMP     #-1,RMER2I              ;ANY ZEROS IN RMER2??
5415  007232  001001                          BNE     10$
5416                                  ;NONE OF THE BITS WERE CLEARED
5417  007234  104004                          ERROR   4               ;MASSBUS INIT FAILED
5418  007236                          10$:
5419
5420                                  ;;*****************************************************************
5421                                  ;*TEST 4        CLEAR STUCK ACTIVE TEST
5422
5423                                  ;;*****************************************************************
5424  007236  000004                  T$T4:   SCOPE
5425  007240  012737  000004  001226          MOV     #4,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
5426
5427  007246  000240                          NOP
5428  007250  012737  000024  001120          MOV     #20.,$ICNT              ;20 ITERATIONS
5429  007256  112737  000001  001131          MOVB    #1,$ERMAX               ;ONE ERROR ALLOWED
5430  007264  012737  007300  001122          MOV     #T4,$LPADR      ;LOAD LOOP ON TEST ADDRESS
5431  007272  012737  007300  001124          MOV     #T4,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
5432  007300                          T4:
5433  007300  012706  001100                  MOV     #STACK,SP               ;LOAD THE STACK POINTER
5434  007304  013700  001276                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS OF UUT
5435  007310  013701  001456                  MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
5436  007314  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5437  007322  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5438                                  ;WRITE ONES IN TEST REGISTERS
5439
5440  007326  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
5441
5442  007334  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
5443
5444  007342  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
5445                                  ;READ TEST REGISTERS AND SEE IF ANY BITS ARE ON
5446
5447  007350  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5448
5449  007356  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5450
5451  007364  016037  000024  001352          MOV     RMMR1(R0),RMMR1I        ;STORE RMMR1 IN INPUT BUFFER
5452  007372  042737  040000  001342          BIC     #UNS,RMER1I     ;DONT ACCEPT UNSAFE
5453  007400  001011                          BNE     10$             ;BRANCH IF ANY OTHER BITS ON
5454  007402  042737  040200  001370          BIC     #SKI!DVC,RMER2I ;DONT ACCEPT SKI OR DVC
5455  007410  001005                          BNE     10$             ;BRANCH IF ANY OTHER BITS ON
5456  007412  032737  000001  001352          BIT     #DMD,RMMR1I     ;BRANCH IF DMD IS ON
```

```
5457  007420  001001                          BNE     10$
5458  007422  104026                          ERROR   26              ;MBA CLR IS STUCK ACTIVE
5459  007424                          10$:
5460
5461
5462                                  ;;**********************************************************************
5463                                  ;*TEST 5         TRISTATE TRANSFER TEST
5464
5465                                  ;;**********************************************************************
5466  007424  000004                  TST5:   SCOPE
5467  007426  012737  000005  001226          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5468  007434  000240                          NOP
5469  007436  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
5470  007444  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
5471  007452  012737  007466  001122          MOV     #T5,$LPADR  ;LOAD LOOP ON TEST ADDRESS
5472  007460  012737  007466  001124          MOV     #T5,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
5473  007466                          T5:
5474  007466  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
5475  007472  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
5476  007476  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
5477  007502  005002                          CLR     R2              ;CLEAR ERROR FLAGS
5478  007504  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5479  007512  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5480                                  ;WRITE ONES IN SELECTED REGISTERS
5481
5482  007516  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
5483
5484  007524  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
5485
5486  007532  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
5487
5488  007540  012760  177777  000032          MOV     #-1,RMOF(R0)    ;LOAD RMOF
5489
5490  007546  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
5491                                  ;WRITE ZEROS IN SELECTED REGISTERS
5492
5493  007554  012760  000000  000000          MOV     #0,RMCS1(R0)    ;LOAD RMCS1
5494
5495  007562  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
5496
5497  007570  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
5498
5499  007576  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
5500
5501  007604  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5502
5503  007612  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
5504                                  ;READ BACK ALL REGISTERS
5505
5506  007620  016037  000000  001326          MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
5507
5508  007626  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5509
5510  007634  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5511
5512  007642  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
```

```
5513
5514  007650  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5515
5516  007656  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5517                                  ;CHECK EACH REGISTER CONTENT FOR ZERO BITS WRITTEN & READ
5518  007664  012702  177777                  MOV     #-1,R2          ;ACCUMULATE ZEROS IN R2
5519  007670  052737  177701  001326          BIS     #^CILF76,RMCS1I ;SET ALL BITS NOT WRITTEN
5520  007676  052737  161577  001360          BIS     #XNUOF,RMOFI
5521  007704  052737  176000  001362          BIS     #XNUDC,RMDCI
5522  007712  052737  001567  001370          BIS     #XNUER2,RMER2I
5523  007720  005137  001326                  COM     RMCS1I          ;COMPLEMENT REGISTER CONTENTS
5524  007724  005137  001334                  COM     RMDAI
5525  007730  005137  001342                  COM     RMER1I
5526  007734  005137  001360                  COM     RMOFI
5527  007740  005137  001362                  COM     RMDCI
5528  007744  005137  001370                  COM     RMER2I
5529  007750  043702  001326                  BIC     RMCS1I,R2               ;ACCUMULATE ALL ZERO BITS
5530  007754  043702  001334                  BIC     RMDAI,R2
5531  007760  043702  001342                  BIC     RMER1I,R2
5532  007764  043702  001360                  BIC     RMOFI,R2
5533  007770  043702  001362                  BIC     RMDCI,R2
5534  007774  043702  001370                  BIC     RMER2I,R2
5535  010000  001407                          BEQ     10$             ;BRANCH IF EACH BIT IS ZERO
5536                                  ;ONE OR MORE BIT POSITIONS ARE NOT ZERO
5537  010002  010237  001142                  MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
5538  010006  005037  001140                  CLR     $GDDAT          ;LOAD EXPECTED RESULT
5539  010012  104005                          ERROR   5               ;TRISTATE BUS IS STUCK AT ONE
5540  010014  052702  000001                  BIS     #BIT0,R2        ;SET ERROR FLAG
5541  010020                          10$:
5542
5543  010020  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5544  010026  111160  000010                          MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5545                                  ;PRESET SELECTED REGISTERS TO ZEROS
5546                                  ;(ASSUME RMCS1, RMER1, RMER2 WERE CLEARED BY INIT)
5547
5548  010032  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
5549
5550  010040  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
5551
5552  010046  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5553                                  ;WRITE ONES IN SELECTED REGISTERS
5554
5555  010054  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
5556
5557  010062  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
5558
5559  010070  012760  016200  000032          MOV     #^CXNUOF,RMOF(R0)       ;LOAD RMOF
5560
5561  010076  012760  001777  000034          MOV     #^CXNUDC,RMDC(R0)       ;LOAD RMDC
5562
5563  010104  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
5564
5565  010112  012760  176210  000042          MOV     #^CXNUER2,RMER2(R0)     ;LOAD RMER2
5566                                  ;READ ALL REGISTERS
5567
5568  010120  016037  000000  001326          MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
```

```
5569
5570  010126  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5571
5572  010134  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5573
5574  010142  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5575
5576  010150  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5577
5578  010156  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5579                                  ;CHECK EACH REGISTER CONTENT FOR ONE BITS WRITTEN & READ
5580  010164  042737  177701  001326          BIC     #^CILF76,RMCS1I         ;CLEAR ALL BITS NOT WRITTEN
5581  010172  042737  161577  001360          BIC     #XNUOF,RMOFI
5582  010200  042737  176000  001362          BIC     #XNUDC,RMDCI
5583  010206  042737  001567  001370          BIC     #XNUER2,RMER2I
5584  010214  005002                          CLR     R2              ;ACCUMULATE ONES IN R2
5585  010216  053702  001326                  BIS     RMCS1I,R2               ;ACCUMULATE ALL ONE BITS
5586  010222  053702  001334                  BIS     RMDAI,R2
5587  010226  053702  001360                  BIS     RMOFI,R2
5588  010232  053702  001362                  BIS     RMDCI,R2
5589  010236  053702  001342                  BIS     RMER1I,R2
5590  010242  053702  001370                  BIS     RMCR2I,R2
5591  010246  022702  177777                  CMP     #-1,R2          ;SEE IF EACH BIT POSITION WAS ONE
5592  010252  001410                          BEQ     20$             ;BRANCH IF NONE STUCK
5593                                  ;ONE OR MORE BIT POSITIONS ARE NOT ONE
5594  010254  010237  001142                  MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
5595  010260  012737  177777  001140          MOV     #-1,$GDDAT              ;EXPECTED RESULT
5596  010266  104006                          ERROR   6
5597  010270  052702  000002                  BIS     #BIT1,R2        ;SET ERROR FLAG
5598  010274                          20$:
5599
5600  010274  005702                          TST     R2              ;ANY ERRORS DETECTED ??
5601  010276  001131                          BNE     30$             ;YES - DONT DO BIT TEST
5602  010300  012702  000001                  MOV     #1,R2           ;R2=BIT POSITION
5603  010304                          25$:
5604  010304  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5605  010312  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5606                                  ;WRITE THE BIT PATTERN IN SELECTED DEVICE REGISTERS
5607
5608  010316  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5609
5610  010322  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
5611
5612  010326  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
5613
5614  010332  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
5615
5616  010336  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
5617                                  ;READ BACK THE REGISTERS
5618
5619  010342  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5620
5621  010350  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5622
5623  010356  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5624
```

```
5625  010364  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5626
5627  010372  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5628                                  ;CHECK REGISTER CONTENTS FOR CORRECT PATTERN
5629  010400  005003                          CLR     R3                      ;R3=ACCUMULATED ONE BIT
5630  010402  012704  177777                  MOV     #-1,R4                  ;R4=ACCUMULATED ZERO BITS
5631  010406  013705  001334                  MOV     RMDAI,R5                ;GET ANY GOOD BITS FROM RMDA
5632  010412  050503                          BIS     R5,R3
5633  010414  005105                          COM     R5
5634  010416  040504                          BIC     R5,R4
5635  010420  013705  001360                  MOV     RMOFI,R5                ;GET GOOD BITS FROM RMOF
5636  010424  042705  161577                  BIC     #XNUOF,R5
5637  010430  050503                          BIS     R5,R3
5638  010432  005105                          COM     R5
5639  010434  042705  161577                  BIC     #XNUOF,R5
5640  010440  040504                          BIC     R5,R4
5641  010442  013705  001362                  MOV     RMDCI,R5                ;GET GOOD BITS FROM RMDC
5642  010446  042705  176000                  BIC     #XNUDC,R5
5643  010452  050503                          BIS     R5,R3
5644  010454  005105                          COM     R5
5645  010456  042705  176000                  BIC     #XNUDC,R5
5646  010462  040504                          BIC     R5,R4
5647  010464  013705  001342                  MOV     RMER1I,R5               ;GET GOOD BITS FROM RMER1
5648  010470  050503                          BIS     R5,R3
5649  010472  005105                          COM     R5
5650  010474  040504                          BIC     R5,R4
5651  010476  013705  001370                  MOV     RMER2I,R5               ;GET GOOD BITS FROM RMER2
5652  010502  042705  001567                  BIC     #XNUER2,R5
5653  010506  050503                          BIS     R5,R3
5654  010510  005105                          COM     R5
5655  010512  042705  001567                  BIC     #XNUER2,R5
5656  010516  040504                          BIC     R5,R4
5657  010520  010205                          MOV     R2,R5                   ;RESET ALL ONES IN R3 EXCEPT
5658  010522  005105                          COM     R5                      ;FOR THE  EST BIT
5659  010524  040503                          BIC     R5,R3
5660  010526  040204                          BIC     R2,R4                   ;RESET TEST BIT IN R4
5661  010530  050403                          BIS     R4,R3                   ;COMBINE ACCUMULATED 1'S + 0'S
5662  010532  020302                          CMP     R3,R2                   ;IS PATTERN OK??
5663  010534  001406                          BEQ     26$                     ;YES!!
5664  010536  010237  001140                  MOV     R2,$GDDAT                       ;SAVE TEST PATTERN
5665  010542  010337  001142                  MOV     R3,$BDDAT                       ;SAVE RESULT
5666  010546  104007                          ERROR   7                       ;BIT INTERFERENCE IN TRISTATE BUS
5667  010550  000404                          BR      30$                     ;SKIP TO NEXT
5668  010552                          26$:
5669                                  ;ADVANCE R2 TO THE NEXT PATTERN AND REPEAT TEST
5670  010552  006302                          ASL     R2                      ;SHIFT THE BIT
5671  010554  001402                          BEQ     30$                     ;EXIT IF DONE
5672  010556  000137  010304                  JMP     25$
5673  010562                          30$:
5674
5675                                  ;;**********************************************************************
5676                                  ;*TEST 6         REGISTER SELECT TEST
5677
5678                                  ;;**********************************************************************
5679  010562  000004                  TST6:   SCOPE
5680  010564  012737  000006  001226          MOV     #6,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
```

```
5681
5682  010572  000240                          NOP
5683  010574  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
5684  010602  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
5685  010610  012737  010624  001122          MOV     #T6,$LPADR  ;LOAD LOOP ON TEST ADDRESS
5686  010616  012737  010624  001124          MOV     #T6,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
5687  010624                              T6:
5688  010624  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
5689  010630  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
5690  010634  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
5691                                  ;THE FOLLOWING TABLE GIVES MASSBUS REGISTER SELECT VALUES FOR
5692                                  ;EACH DEVICE REGISTER
5693
5694                                  ;       REGISTER        REG SEL
5695                                  ;       NAME            (16,8,4,2,1)
5696                                  ;
5697                                  ;       RMCS1           00000
5698                                  ;       RMDS            00001
5699                                  ;       RMER1           00010
5700                                  ;       RMMR1           00011
5701                                  ;       RMAS            00100
5702                                  ;       RMDA            00101
5703                                  ;       RMDT            00110
5704                                  ;       RMLA            00111
5705                                  ;       RMSN            01000
5706                                  ;       RMOF            01001
5707                                  ;       RMDC            01010
5708                                  ;       RMHR            01011
5709                                  ;       RMMR2           01100
5710                                  ;       RMER2           01101
5711                                  ;       RMEC1           01110
5712                                  ;       RMEC2           01111
5713
5714                                  ;EACH REGISTER SELECT LINE IS TESTED FOR A STUCK AT ONE,
5715                                  ;STUCK AT ZERO FAULT. AS AN EXAMPLE, TO TEST REG SEL 1.
5716                                  ;FOR S-A-0, RMER1 IS WRITTEN WITH ZEROS. THEN THE REGISTER
5717                                  ;THAT HAS THE SAME SELECT VALUE, EXCEPT FOR THE SELECT LINE
5718                                  ;BEING TESTED, IS WRITTEN WITH ONES. IN THIS EXAMPLE,
5719                                  ;RMMR1 IS WRITTEN WITH ONES. IF SELECT LINE 1 IS S-A-0,
5720                                  ;THE ALL ONES WORD WILL BE WRITTEN IN RMER1, AND RMER1
5721                                  ;WILL NOT BE 0 WHEN READ BACK.
5722
5723  010640  005002                          CLR     R2              ;R2= ZEROS SOURCE
5724  010642  012703  177777                  MOV     #-1,R3          ;R3= ONES  SOURCE
5725                                  ;TEST REG SEL 1 FOR S-A-0
5726  010646  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5727  010654  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5728
5729  010660  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
5730
5731  010664  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
5732
5733  010670  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
5734
5735  010674  010360  000036                  MOV     R3,RMHR(R0)     ;LOAD RMHR
5736
```

```
5737  010700  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5738
5739  010706  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5740  010714  020337  001342                  CMP     R3,RMER1I
5741  010720  001007                          BNE     10$
5742  010722  052737  176000  001362          BIS     #XNUDC,RMDCI
5743  010730  020337  001362                  CMP     R3,RMDCI
5744  010734  001001                          BNE     10$
5745  010736  104010                          ERROR   10              ;REG SEL 1 IS S-A-0
5746  010740                          10$:
5747
5748                                  ;TEST REG SEL 1 FOR S-A-1
5749  010740  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5750  010746  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5751
5752  010752  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5753
5754  010756  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
5755
5756  010762  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
5757
5758  010766  010360  000016                  MOV     R3,RMAS(R0)     ;LOAD RMAS
5759
5760  010772  010360  000030                  MOV     R3,RMSN(R0)     ;LOAD RMSN
5761
5762  010776  010360  000040                  MOV     R3,RMMR2(R0)    ;LOAD RMMR2
5763
5764  011002  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5765
5766  011010  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5767
5768  011016  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5769  011024  020337  001334                  CMP     R3,RMDAI
5770  011030  001015                          BNE     20$
5771  011032  052737  161577  001360          BIS     #XNUOF,RMOFI
5772  011040  020337  001360                  CMP     R3,RMOFI
5773  011044  001007                          BNE     20$
5774  011046  052737  001567  001370          BIS     #XNUER2,RMER2I
5775  011054  020337  001370                  CMP     R3,RMER2I
5776  011060  001001                          BNE     20$
5777  011062  104011                          ERROR   11              ;REG SEL 1 IS S-A-1
5778  011064                          20$:
5779
5780                                  ;TEST REG SEL 2 FOR S-A-0
5781  011064  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5782  011072  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5783
5784  011076  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5785
5786  011102  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
5787
5788  011106  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
5789
5790  011112  010360  0000_J                  MOV     R3,RMLA(R0)     ;LOAD RMLA
5791
5792  011116  010360  000036                  MOV     R3,RMHR(R0)     ;LOAD RMHR
```

```
5793
5794  011122  010360  000046                  MOV     R3,RMEC2(R0)    ;LOAD RMEC2
5795
5796  011126  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5797
5798  011134  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5799
5800  011142  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5801  011150  020337  001334                  CMP     R3,RMDAI
5802  011154  001015                          BNE     30$
5803  011156  052737  161577  001360          BIS     #XNUOF,RMOFI
5804  011164  020337  001360                  CMP     R3,RMOFI
5805  011170  001007                          BNE     30$
5806  011172  052737  001567  001370          BIS     #XNUER2,RMER2I
5807  011200  020337  001370                  CMP     R3,RMER2I
5808  011204  001001                          BNE     30$
5809  011206  104012                          ERROR   12              ;REG SEL 2 IS S-A-0
5810  011210                          30$:
5811
5812                                  ;TEST REG SEL 2 FOR S-A-1
5813  011210  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5814  011216  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5815
5816  011222  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
5817
5818  011226  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
5819
5820  011232  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
5821
5822  011240  010360  000030                  MOV     R3,RMSN(R0)     ;LOAD RMSN
5823
5824  011244  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5825
5826  011252  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5827  011260  052737  177701  001342          BIS     #^CILF76,RMER1I
5828  011266  020337  001342                  CMP     R3,RMER1I
5829  011272  001007                          BNE     40$
5830  011274  052737  176000  001362          BIS     #XNUDC,RMDCI
5831  011302  020337  001362                  CMP     R3,RMDCI
5832  011306  001001                          BNE     40$
5833  011310  104013                          ERROR   13              ;REG SEL 2 IS S-A-1
5834  011312                          40$:
5835
5836                                  ;TEST REG SEL 4 FOR S-A-0
5837  011312  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5838  011320  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5839
5840  011324  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
5841
5842  011330  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
5843
5844  011334  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
5845
5846  011340  010360  000026                  MOV     R3,RMDT(R0)     ;LOAD RMDT
5847
5848  011344  010360  000042                  MOV     R3,RMER2(R0)    ;LOAD RMER2
```

```
5849
5850  011350  010360  000044                  MOV     R3,RMEC1(R0)    ;LOAD RMEC1
5851
5852  011354  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5853
5854  011362  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5855
5856  011370  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5857  011376  020337  001342                  CMP     R3,RMER1I
5858  011402  001015                          BNE     50$
5859  011404  052737  161577  001360          BIS     #XNUOF,RMOFI
5860  011412  020337  001360                  CMP     R3,RMOFI
5861  011416  001007                          BNE     50$
5862  011420  052737  176000  001362          BIS     #XNUDC,RMDCI
5863  011426  020337  001362                  CMP     R3,RMDCI
5864  011432  001001                          BNE     50$
5865  011434  104014                          ERROR   14              ;REG SEL 4 IS S-A-0
5866  011436                          50$:
5867
5868                          ;TEST REG SEL 4 FOR S-A-1
5869  011436  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5870  011444  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5871
5872  011450  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5873
5874  011454  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
5875
5876  011460  010360  000012                  MOV     R3,RMDS(R0)     ;LOAD RMDS
5877
5878  011464  010360  000032                  MOV     R3,RMOF(R0)     ;LOAD RMOF
5879
5880  011470  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5881
5882  011476  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5883  011504  020337  001334                  CMP     R3,RMDAI
5884  011510  001007                          BNE     60$
5885  011512  052737  001567  001370          BIS     #XNUER2,RMER2I
5886  011520  020337  001370                  CMP     R3,RMER2I
5887  011524  001001                          BNE     60$
5888  011526  104015                          ERROR   15              ;REG SEL 4 IS S-A-1
5889  011530                          60$:
5890
5891                          ;TEST REG SEL 8 FOR S-A-0
5892  011530  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5893  011536  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5894
5895  011542  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
5896
5897  011546  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5898
5899  011552  010360  000034                  MOV     R3,RMDC(R0)     ;LOAD RMDC
5900
5901  011556  010360  000042                  MOV     R3,RMER2(R0)    ;LOAD RMER2
5902
5903  011562  016037  000014  001342          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
5904
```

```
5905  011570  016037  000006  001334          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
5906  011576  020337  001342                  CMP     R3,RMER1I
5907  011602  001004                          BNE     70$
5908  011604  020337  001334                  CMP     R3,RMDAI
5909  011610  001001                          BNE     70$
5910  011612  104016                          ERROR   16              ;REG SEL 8 IS S-A-0
5911  011614                          70$:
5912
5913                                  ;TEST REG SEL 8 FOR S-A-1
5914  011614  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
5915  011622  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
5916
5917  011626  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
5918
5919  011632  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
5920
5921  011636  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
5922
5923  011642  010360  000012                  MOV     R3,RMDS(R0)     ;LOAD RMDS
5924
5925  011646  010360  000014                  MOV     R3,RMER1(R0)    ;LOAD RMER1
5926
5927  011652  010360  000006                  MOV     R3,RMDA(R0)     ;LOAD RMDA
5928
5929  011656  016037  000032  001360          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
5930
5931  011664  016037  000034  001362          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
5932
5933  011672  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
5934  011700  052737  161577  001360          BIS     #XNUOF,RMOFI
5935  011706  001015                          BNE     80$
5936  011710  022737  176000  001362          CMP     #XNUDC,RMDCI
5937  011716  020337  001362                  CMP     R3,RMDCI
5938  011722  001007                          BNE     80$
5939  011724  052737  001567  001370          BIS     #XNUER2,RMER2I
5940  011732  020337  001370                  CMP     R3,RMER2I
5941  011736  001001                          BNE     80$
5942  011740  104017                          ERROR   17              ;REG SEL 8 IS S-A-1
5943  011742                          80$:
5944
5945                                  ;REGISTER SELECT 16 IS TESTED BY THE ILR TEST
5946
5947                                  ;;*****************************************************************
5948                                  ;*TEST 7         DRIVE TYPE TEST
5949
5950                                  ;;*****************************************************************
5951  011742  000004                  T$T7:   SCOPE
5952  011744  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5953
5954  011752  000240                          NOP
5955  011754  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
5956  011762  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
5957  011770  012737  012004  001122          MOV     #T7,$LPADR  ;LOAD LOOP ON TEST ADDRESS
5958  011776  012737  012004  001124          MOV     #T7,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
5959  012004                          T7:
5960  012004  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
```

```
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
                                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE

012020  016037  000026  001142          MOV     RMDT(R0),$BDDAT ;STORE RMDT AT $BDDAT
012026  022737  020024  001142          CMP     #SNGPRT,$BDDAT  ;SINGLE PORT RM03??
012034  001422                          BEQ     10$             ;YES!!
012036  022737  024024  001142          CMP     #DULPRT,$BDDAT  ;DUAL PORT RM03??
012044  001416                          BEQ     10$             ;YES!!
012046  012737  020024  001174          MOV     #SNGPRT,$TMP0   ;LOAD ACCEPTABLE VALUES
012054  012737  024024  001176          MOV     #DULPRT,$TMP1
012062  010037  001136                  MOV     R0,$BDADR       ;LOAD BAD ADDRESS
012066  062737  000026  001136          ADD     #RMDT,$BDADR
012074  104057                          ERROR   57              ;DEVICE NOT AN RM03
012076  000137  057050                  JMP     $EOSP           ;GO TO NEXT DEVICE
012102                          10$:

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

                                ;;******************************************************************
012102  000004                  TST10:  SCOPE
012104  012737  000010  001226          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

012112  000240                          NOP
012114  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
012122  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
012130  012737  012144  001122          MOV     #T10,$LPADR  ;LOAD LOOP ON TEST ADDRESS
012136  012737  012144  001124          MOV     #T10,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
012144                          T10:
012144  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
012150  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
012154  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
012160  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
012166  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

012172  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
012200  042737  173777  001142          BIC     #^CDVA,$BDDAT   ;CLEAR ALL BUT DVA
012206  001006                          BNE     10$             ;BRANCH IF DVA SET
012210  012737  004000  001140          MOV     #DVA,$GDDAT     ;SETUP EXPECTED
012216  010037  001136                  MOV     R0,$BDADR       ;SETUP REG ADDRESS
012222  104060                          ERROR   60              ;DEVICE NOT AVAILABLE
012224                          10$:

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

                                ;;******************************************************************
012224  000004                  TST11:  SCOPE
012226  012737  000011  001226          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

012234  000240                          NOP
012236  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
012244  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
012252  012737  012266  001122          MOV     #T11,$LPADR  ;LOAD LOOP ON TEST ADDRESS
012260  012737  012266  001124          MOV     #T11,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
012266                          T11:
```

```
6017  012266  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
6018  012272  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6019  012276  013701  001456                MOV     TSTOUE,R1       ;R1 = POINTER TO DEVICE
6020  012302  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6021  012310  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6022  012314  005003                        CLR     R3              ;CLEAR ERROR FLAGS
6023  012316  010037  001136                MOV     R0,$BDAOR       ;SETUP REGISTER ADDRESS
6024  012322  062737  000036  001136        ADD     #RMHR,$BDAOR
6025                                ;WRITE ONES THEN ZEROS IN RMHR AND CHECK FOR S-A-1 BITS.
6026                                ;NOTE THAT IT IS NECESSARY TO WRITE SOME OTHER REGISTER IN
6027                                ;ORDER TO WRITE THE HOLDING REGISTER, AND RMDA IS USED FOR THIS
6028                                ;PURPOSE.
6029
6030  012330  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
6031
6032  012336  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
6033
6034  012344  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
6035  012352  005137  001142                  COM     $BDDAT          ;ANY ERROR??
6036  012356  001405                          BEQ     10$             ;NO!!
6037  012360  005037  001140                  CLR     $GDDAT          ;LOAD EXPECTED
6038  012364  104061                          ERROR   61
6039  012366  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAGS
6040  012372                          10$:
6041                                ;;*****************************************************************
6042                                ;WRITE ZEROS THEN ONES IN RMHR AND CHECK FOR S-A-0 BITS.
6043
6044  012372  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
6045
6046  012400  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
6047
6048  012406  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
6049  012414  005137  001142                  COM     $BDDAT          ;RMHR IS COMPLEMENTED WHEN READ
6050  012420  012737  177777  001140          MOV     #-1,$GDDAT      ;SETUP EXPECTED
6051  012426  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;ANY ERROR??
6052  012434  001403                          BEQ     20$             ;NO!!
6053  012436  104062                          ERROR   62
6054  012440  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
6055  012444                          20$:
6056                                ;;*****************************************************************
6057                                ;IF NO PREVIOUS ERRORS, WRITE AND READ SHIFTING ONE BIT PATTERN.
6058  012444  005703                          TST     R3              ;ANY FLAGS SET??
6059  012446  001025                          BNE     50$             ;YES!!
6060  012450  012702  000001                  MOV     #1,R2           ;R2=DATA PATTERN
6061
6062  012454                          30$:
6063
6064  012454  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
6065
6066  012462  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
6067
6068  012466  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
6069  012474  005137  001142                  COM     $BDDAT          ;RMHR IS COMPLEMENTED
6070  012500  023702  001142                  CMP     $BDDAT,R2       ;ANY ERROR??
6071  012504  001404                          BEQ     40$             ;NO!!
6072  012506  010237  001140                  MOV     R2,$GDDAT       ;SETUP EXPECTED
```

```
6073  012512  104063                        ERROR   63
6074  012514  000402                        BR      50$             ;DO NOT COLLECT ALL ERRORS
6075
6076  012516  006302                40$:    ASL     R2              ;SHIFT TO NEXT PATTERN
6077  012520  001355                        BNE     30$             ;CONTINUE IF NOT DONE
6078
6079  012522                        50$:
6080
6081                                ;;*****************************************************************
6082                                ;*TEST 12        CONTROL STATUS #1 TRANSFER TEST
6083
6084                                ;;*****************************************************************
6085  012522  000004                TST12:  SCOPE
6086  012524  012737  000012  001226        MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6087
6088  012532  000240                        NOP
6089  012534  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
6090  012542  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6091  012550  012737  012564  001122        MOV     #T12,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6092  012556  012737  012564  001124        MOV     #T12,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6093  012564                        T12:
6094  012564  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
6095  012570  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6096  012574  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6097  012600  005003                        CLR     R3              ;R3 = ERROR INDICATOR
6098  012602  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6099  012610  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6100                               ;WRITE ONES IN RMCS1, BITS 01-05, THEN CLEAR. READ AND
6101                               ;CHECK FOR S-A-1 BITS.
6102
6103  012614  012760  000076  000000                MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
6104  012622  012760  000040  000010                MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6105  012630  111160  000010                        MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6106
6107  012634  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
6108  012642  042737  177701  001142                BIC     #^CILF76,$BDDAT
6109  012650  001410                                BEQ     5$
6110  012652  005037  001140                        CLR     $GDDAT
6111  012656  010037  001136                        MOV     R0,$BDADR
6112  012662  062737  000000  001136                ADD     #RMCS1,$BDADR
6113  012670  104043                                ERROR   43              ;CANT CLEAR RMCS1
6114  012672                                5$:
6115                                ;WRITE ONES IN RMCS1, BITS 01-05, THEN WRITE ZEROS. READ AND CHECK FOR
6116                                ;S-A-1 BITS.
6117
6118  012672  012760  000076  000000                MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
6119
6120  012700  012760  000000  000000                MOV     #0,RMCS1(R0)    ;LOAD RMCS1
6121
6122  012706  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
6123  012714  042737  177701  001142                BIC     #^CILF76,$BDDAT
6124  012722  001412                                BEQ     10$
6125  012724  005037  001140                        CLR     $GDDAT
6126  012730  010037  001136                        MOV     R0,$BDADR
6127  012734  062737  000000  001136                ADD     #RMCS1,$BDADR
6128  012742  104023                                ERROR   23              ;CANT WRITE 0'S
```

```
                        T12     CONTROL STATUS #1 TRANSFER TEST

6129  012744  052703  000001            BIS     #BIT0,R3        ;SET ERROR FLAG
6130  012750                    10$:
6131                            ;WRITE ZEROS IN RMCS1, THEN ONES, READ AND CHECK S-A-0 BITS.
6132
6133  012750  012760  000000  000000    MOV     #0,RMCS1(R0)    ;LOAD RMCS1
6134
6135  012756  012760  000076  000000    MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
6136
6137  012764  016037  000000  001142    MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
6138  012772  042737  177701  001142    BIC     #^CILF76,$BDDAT
6139  013000  012737  000076  001140    MOV     #ILF76,$GDDAT
6140  013006  023737  001140  001142    CMP     $GDDAT,$BDDAT
6141  013014  001410                    BEQ     20$
6142  013016  010037  001136            MOV     R0,$BDADR
6143  013022  062737  000000  001136    ADD     #RMCS1,$BDADR
6144  013030  104024                    ERROR   24              ;CANT WRITE ONES
6145  013032  052703  000002            BIS     #BIT1,R3        ;SET ERROR FLAG
6146  013036                    20$:
6147                            ;WRITE A SHIFTING ONE BIT PATTERN IN RMCS1, READ AND CHECK FOR STUCK BITS.
6148  013036  005703                    TST     R3              ;OMIT IF ANY ERRORS
6149  013040  001035                    BNE     50$
6150  013042  012702  000002            MOV     #2,R2           ;R2 = TEST PATTERN
6151  013046                    30$:
6152  013046  010203                    MOV     R2,R3           ;R3 = EXPECTED RESULT, BITS 1-5
6153  013050  042703  177701            BIC     #^CILF76,R3
6154
6155  013054  012760  000000  000000    MOV     #0,RMCS1(R0)    ;LOAD RMCS1
6156
6157  013062  010260  000000            MOV     R2,RMCS1(R0)    ;LOAD RMCS1
6158
6159  013066  016037  000000  001142    MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
6160  013074  042737  177701  001142    BIC     #^CILF76,$BDDAT
6161  013102  020337  001142            CMP     R3,$BDDAT
6162  013106  001410                    BEQ     40$
6163  013110  010337  001140            MOV     R3,$GDDAT
6164  013114  010037  001136            MOV     R0,$BDADR
6165  013120  062737  000000  001136    ADD     #RMCS1,$BDADR
6166  013126  104025                    ERROR   25              ;CANT WRITE SHIFTING ONES
6167  013130  006302            40$:    ASL     R2              ;SHIFT TO NEXT BIT
6168  013132  001345                    BNE     30$             ;CONTINUE IF R2 NOT ZERO
6169  013134                    50$:
6170
6171                            ;;*****************************************************************
6172                            ;*TEST 13        ERROR REGISTER 1 TRANSFER TEST
6173
6174                            ;;*****************************************************************
6175  013134  000004            T$T13:  SCOPE
6176  013136  012737  000013  001226    MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6177
6178  013144  000240                    NOP
6179  013146  012737  000024  001120    MOV     #20.,$ICNT      ;20 ITERATIONS
6180  013154  112737  000001  001131    MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6181  013162  012737  013176  001122    MOV     #T13,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6182  013170  012737  013176  001124    MOV     #T13,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6183  013176                    T13:
6184  013176  012706  001100            MOV     #STACK,SP       ;LOAD THE STACK POINTER
```

```
6185  013202  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6186  013206  013701  00145F              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6187  013212  005003                      CLR     R3              ;CLEAR ERROR FLAG
6188                                ;WRITE ONES IN RMER1, CLEAR AND CHECK FOR S-A-1 BITS
6189
6190  013214  012760  177777  000014      MOV     #-1,RMER1(R0)   ;LOAD RMER1
6191  013222  012760  000040  000010      MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6192  013230  111160  000010              MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6193
6194  013234  016037  000014  001342      MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
6195  013242  013737  001342  001142      MOV     RMER1I,$BDDAT
6196  013250  042737  177760  001142      BIC     #^C<PAR!RMR!ILF!ILR>,$BDDAT
6197  013256  001410                      BEQ     10$
6198  013260  005037  001140              CLR     $GDDAT
6199  013264  010037  001136              MOV     R0,$BDADR
6200  013270  062737  000014  001136      ADD     #RMER1,$BDADR
6201  013276  104027                      ERROR   27              ;CANT CLEAR RMER1
6202  013300                        10$:
6203  013300  013737  001342  001142      MOV     RMER1I,$BDDAT
6204  013306  042737  074017  001142      BIC     #^C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,$BDDAT
6205  013314  001410                      BEQ     20$
6206  013316  005037  001140              CLR     $GDDAT
6207  013322  010037  001136              MOV     R0,$BDADR
6208  013326  062737  000014  001136      ADD     #RMER1,$BDADR
6209  013334  104030                      ERROR   30
6210  013336                        20$:
6211  013336  013737  001342  001142      MOV     RMER1I,$BDDAT
6212  013344  042737  147777  001142      BIC     #^C<OPI!DTE>,$BDDAT
6213  013352  001410                      BEQ     30$
6214  013354  005037  001140              CLR     $GDDAT
6215  013360  010037  001136              MOV     R0,$BDADR
6216  013364  062737  000014  001136      ADD     #RMER1,$BDADR
6217  013372  104031                      ERROR   31
6218  013374                        30$:
6219                                ;WRITE ONES THEN ZEROS IN RMER1, READ AND CHECK FOR S-A-1 BITS
6220
6221  013374  012760  177777  000014      MOV     #-1,RMER1(R0)   ;LOAD RMER1
6222
6223  013402  012760  000000  000014      MOV     #0,RMER1(R0)    ;LOAD RMER1
6224
6225  013410  016037  000014  001342      MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
6226  013416  013737  001342  001142      MOV     RMER1I,$BDDAT
6227  013424  042737  177770  001142      BIC     #^C<RMR!ILF!ILR>,$BDDAT
6228  013432  001412                      BEQ     40$
6229  013434  005037  001140              CLR     $GDDAT
6230  013440  010037  001136              MOV     R0,$BDADR
6231  013444  062737  000014  001136      ADD     #RMER1,$BDADR
6232  013452  104032                      ERROR   32
6233  013454  052703  000001              BIS     #BIT0,R3        ;SET ERROR FLAG
6234  013460  013737  001342  001142  40$:  MOV     RMER1I,$BDDAT
6235  013466  042737  074017  001142      BIC     #^C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,$BDDAT
6236  013474  001412                      BEQ     50$
6237  013476  005037  001140              CLR     $GDDAT
6238  013502  010037  001136              MOV     R0,$BDADR
6239  013506  062737  000014  001136      ADD     #RMER1,$BDADR
6240  013514  104033                      ERROR   33
```

```
6241  013516  052703  000001              BIS     #BIT0,R3        ;SET ERROR FLAG
6242  013522                          50$:
6243  013522  013737  001342  001142      MOV     RMER1I,$BDDAT
6244  013530  042737  147777  001142      BIC     #^C<OPI!DTE>,$BDDAT
6245  013536  001412                      BEQ     60$
6246  013540  005037  001140              CLR     $GDDAT
6247  013544  010037  001136              MOV     R0,$BDADR
6248  013550  062737  000014  001136      ADD     #RMER1,$BDADR
6249  013556  104034                      ERROR   34
6250  013560  052703  030000              BIS     #BIT,R3
6251  013564                          60$:
6252                                  ;WRITE ZEROS THEN ONES IN RMER1,READ AND CHECK FOR S-A-0 BITS
6253
6254  013564  012760  000000  000014      MOV     #0,RMER1(R0)    ;LOAD RMER1
6255
6256  013572  012760  177777  000014      MOV     #-1,RMER1(R0)   ;LOAD RMER1
6257
6258  013600  016037  000014  001142      MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
6259  013606  012737  177777  001140      MOV     #-1,$GDDAT
6260  013614  023737  001140  001142      CMP     $GDDAT,$BDDAT
6261  013622  001410                      BEQ     70$
6262  013624  010037  001136              MOV     R0,$BDADR
6263  013630  062737  000014  001136      ADD     #RMER1,$BDADR
6264  013636  104035                      ERROR   35
6265  013640  052703  000002              BIS     #BIT1,R3
6266  013644                          70$:
6267                                  ;WRITE A SHIFTING 1 BIT IN RMER1 AND CHECK FOR STUCK BITS
6268                                  ;NOTE:  DONT TEST UNSAFE OR PARITY
6269  013644  005703                      TST     R3              ;SKIP THIS PART IF ANY ERRORS
6270  013646  001045                      BNE     120$
6271  013650  012702  000001              MOV     #1,R2           ;R2 = TEST PATTERN
6272  013654                          80$:
6273  013654  012760  000040  000010      MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6274  013662  111160  000010              MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6275
6276  013666  010260  000014              MOV     R2,RMER1(R0)    ;LOAD RMER1
6277
6278  013672  016037  000014  001142      MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
6279  013700  032702  000010              BIT     #PAR,R2         ;DONT TEST PAR = 0
6280  013704  001003                      BNE     90$
6281  013706  042737  000010  001142      BIC     #PAR,$BDDAT
6282  013714  032702  040000          90$:BIT     #UNS,R2         ;DONT TEST UNS = 0
6283  013720  001003                      BNE     100$
6284  013722  042737  040000  001142      BIC     #UNS,$BDDAT
6285  013730  020237  001142          100$:CMP    R2,$BDDAT
6286  013734  001410                      BEQ     110$
6287  013736  010237  001140              MOV     R2,$GDDAT
6288  013742  010037  001136              MOV     R0,$BDADR
6289  013746  062737  000014  001136      ADD     #RMER1,$BDADR
6290  013754  104036                      ERROR   36
6291
6292  013756  006302                  110$:ASL    R2              ;SHIFT TO NEXT BIT
6293  013760  001335                      BNE     80$             ;CONTINUE IF R2 NOT ZERO
6294  013762                          120$:
6295
6296
```

```
                                        ;;*****************************************************************
                                        ;*TEST 14       ERROR REGISTER 2 TRANSFER TEST
                                        ;;*****************************************************************
013762  000004                  TST14:  SCOPE
013764  012737  000014  001226          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
013772  000240                          NOP
013774  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
014002  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
014010  012737  014024  001122          MOV     #T14,$LPADR  ;LOAD LOOP ON TEST ADDRESS
014016  012737  014024  001124          MOV     #T14,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
014024                          T14:
014024  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
014030  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
014034  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
014040  005003                          CLR     R3              ;RESET ERROR FLAGS
014042  010037  001136                  MOV     R0,$BDADR       ;SETUP BAD ADDRESS
014046  062737  000042  001136          ADD     #RMER2,$BDADR
014054  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED DATA
                                        ;;*****************************************************************
                                        ;WRITE ONES IN RMER2, CLEAR AND CHECK FOR S-A-1 BITS

014060  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
014066  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
014074  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

014100  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                ;       TEST UNUSED BITS FOR ZERO-FAILURE ON IF
014106  013737  001370  001142          MOV     RMER2I,$BDDAT
014114  042737  176210  001142          BIC     #^CXNUER2,$BDDAT
014122  001403                          BEQ     10$             ;BRANCH IF NO ERROR
014124  104044                          ERROR   44              ;UNUSED BITS NOT ZERO
014126  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
014132                          10$:
                                ;       TEST "OPE", "IVC", "LSC" FOR ZERO-FAILURE ON CS, IF
014132  013737  001370  001142          MOV     RMER2I,$BDDAT
014140  042737  143777  001142          BIC     #^C<OPE!IVC!LSC>,$BDDAT
014146  001403                          BEQ     20$             ;BRANCH IF NO ERROR
014150  104045                          ERROR   45              ;CANT CLEAR OPE, IVC, LSC
014152  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
014156                          20$:
                                ;       TEST "LBC", "DPE" FOR ZERO-FAILURE ON DS, IF
014156  013737  001370  001142          MOV     RMER2I,$BDDAT
014164  042737  175767  001142          BIC     #^C<LBC!DPE>,$BDDAT
014172  001403                          BEQ     30$             ;BRANCH IF NO ERROR
014174  104046                          ERROR   46              ;CANT CLEAR LBC, DPE
014176  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
014202                          30$:

                                        ;;*****************************************************************
                                        ;WRITE ONES IN RMER2 THEN WRITE ZEROS AND CHECK FOR S-A-1 BITS.

014202  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
```

```
MD-11-DZRMJA-A, RM03 DISKLESS DIAGNOSTIC      MACY11 30(1046)  01-AUG-77  11:17  PAGE 137                    SEQ 0140
DZRMJA.P11    01-AUG-77 11:10          T14    ERROR REGISTER 2 TRANSFER TEST

  6353  014210  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
  6354
  6355  014216  016037  000042  001370        MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
  6356
  6357                                  ;     TEST "OPE", "IVC", "LSC" FOR ZERO-FAILURE ON CS, IF
  6358  014224  013737  001370  001142        MOV     RMER2I,$BDDAT
  6359  014232  042737  143777  001142        BIC     #^C<OPE!IVC!LSC>,$BDDAT
```

```
  6360  014240  001403                    BEQ     40$             ;BRANCH IF NO ERROR
```

```
 6361  014242  104047                          ERROR   47              ;CANT WRITE ZEROS
 6362  014244  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
 6363  014250                              40$:
 6364                                      ;   TEST "LBC", "DPE" FOR ZERO-FAILURE ON DS, IF
 6365  014250  013737  001370  001142          MOV     RMER2I,$BDDAT
```

```
6366  014256  042737  175767  001142          BIC     #↑C<LBC!DPE>,$BDDAT
6367  014264  001403                          BEQ     50$             ;BRANCH IF NO ERROR
6368  014266  104050                          ERROR   50              ;CP*T WRITE ZEROS
6369  014270  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
6370  014274                          50$:
6371
6372                                  ;;*****************************************************************
6373                                  ;WRITE ZEROS IN RMER2 THEN WRITE ONES AND CHECK FOR S-A-0 BITS.
6374
6375  014274  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
6376
6377  014302  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
6378
6379  014310  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
6380                          ;       TEST UNUSED BITS FOR ZERO-FAILURE ON IF
6381  014316  013737  001370  001142          MOV     RMER2I,$BDDAT
6382  014324  042737  176210  001142          BIC     #↑CXNUER2,$BDDAT
6383  014332  001403                          BEQ     60$             ;BRANCH IF NO ERROR
6384  014334  104044                          ERROR   44              ;UNUSED BITS NOT ZERO
6385  014336  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
6386  014342                          60$:
6387                          ;       TEST USED BITS FOR ONE-FAILURE ON IF
6388  014342  013737  001370  001142          MOV     RMER2I,$BDDAT
6389  014350  042737  001567  001142          BIC     #XNUER2,$BDDAT
6390  014356  012737  176210  001140          MOV     #↑CXNUER2,$GDDAT
6391  014364  023737  001140  001142          CMP     $GDDAT,$BDDAT
6392  014372  001403                          BEQ     70$             ;BRANCH IF NO ERROR
6393  014374  104051                          ERROR   51              ;CANT WRITE ONES
6394  014376  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
6395  014402                          70$:
6396
6397                                  ;;*****************************************************************
6398                                  ;IF NO PREVIOUS ERROR, TEST BIT INTERFERENCE WITH SHIFTING ONE BIT.
6399  014402  005703                          TST     R3              ;ANY ERRORS?
6400  014404  001044                          BNE     120$            ;YES!!
6401  014406  012702  000001                  MOV     #1,R2           ;R2=DATA PATTERN
6402  014412                          80$:
6403  014412  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6404  014420  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6405
6406  014424  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
6407
6408  014430  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
6409  014436  032702  040000                  BIT     #SKI,R2         ;IS SKI BEING SET?
6410  014442  001003                          BNE     90$             ;YES!!
6411  014444  042737  040000  001142          BIC     #SKI,$BDDAT     ;DONT TEST SKI FOR ZERO
6412  014452  032702  000200          90$:    BIT     #DVC,R2         ;IS DVC BEING SET??
6413  014456  001003                          BNE     100$            ;YES!!
6414  014460  042737  000200  001142          BIC     #DVC,$BDDAT     ;DONT TEST DVC FOR ZERO
6415  014466  010237  001140          100$:   MOV     R2,$GDDAT
6416  014472  042737  001567  001140          BIC     #XNUER2,$GDDAT  ;UNUSED BITS SHOULD BE ZERO
6417  014500  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;ANY ERRORS??
6418  014506  001401                          BEQ     110$            ;NO!!
6419  014510  104052                          ERROR   52              ;CANT WRITE SHIFTING ONES
6420
6421  014512  006302                  110$:   ASL     R2              ;SHIFT TO NEXT DATA BIT
```

```
6422  014514  001336                        BNE     80$             ;CONTINUE IF NOT DONE
6423
6424  014516                        120$:
6425
6426                                ;;*****************************************************************
6427                                ;*TEST 15        CLEAR OFFSET STUCK ACTIVE TEST
6428
6429                                ;;*****************************************************************
6430  014516  000004                TST15:  SCOPE
6431  014520  012737  000015  001226        MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6432
6433  014526  000240                        NOP
6434  014530  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
      014536  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6436  014544  012737  014560  001122        MOV     #T15,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6437  014552  012737  014560  001124        MOV     #T15,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6438  014560                        T15:
6439  014560  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
6440  014564  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6441  014570  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6442  014574  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6443  014602  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6444
6445  014606  012760  177777  000032        MOV     #-1,RMOF(R0)    ;LOAD RMOF
6446
6447  014614  016037  000032  001142        MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
6448  014622  042737  177577  001142        BIC     #^COFD,$BDDAT
6449  014630  001011                        BNE     10$             ;BRANCH IF OFD IS A ONE
6450  014632  012737  000200  001140        MOV     #OFD,$GDDAT     ;SETUP ERROR MESSAGE
6451  014640  010037  001136                MOV     R0,$BDADR
6452  014644  062737  000032  001136        ADD     #RMOF,$BDADR
6453  014652  104172                        ERROR   172             ;CANT SET OFD BIT
6454  014654                        10$:                            ;END OF TEST
6455
6456
6457                                ;;*****************************************************************
6458                                ;*TEST 16        OFFSET REGISTER TRANSFER TEST
6459
6460                                ;;*****************************************************************
6461  014654  000004                TST16:  SCOPE
6462  014656  012737  000016  001226        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6463
6464  014664  000240                        NOP
6465  014666  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
6466  014674  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6467  014702  012737  014716  001122        MOV     #T16,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6468  014710  012737  014716  001124        MOV     #T16,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6469  014716                        T16:
6470  014716  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
6471  014722  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6472  014726  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6473  014732  005003                        CLR     R3              ;RESET ERROR FLAGS
6474  014734  010037  001136                MOV     R0,$BDADR       ;SETUP BAD ADDRESS
6475  014740  062737  000032  001136        ADD     #RMOF,$BDADR
6476  014746  005037  001140                CLR     $GDDAT          ;SETUP EXPECTED DATA
6477                                ;;*****************************************************************
```

```
                                        ;WRITE ONES THEN ZEROS IN RMOF AND CHECK FOR S-A-1 BITS.
                                                MOV     #-1,RMOF(R0)    ;LOAD RMOF
015744  012760  177777  000032                MOV     #0,RMOF(R0)     ;LOAD RMOF
015752  012760  000000  000032                MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
                                        ;       CHECK UNUSED BITS OF RMOF FOR ZERO
015760  013737  001360  001142                MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
015766  042737  016200  001142                BIC     #^CXNUOF,$BDDAT
015774  001403                              BEQ     10$             ;BRANCH IF NO ERROR
015776  104053                              ERROR   53              ;UNUSED BITS NOT ZERO RMOF
016000  052703  000001                      BIS     #BIT0,R3        ;SET ERROR FLAG
016004                                  10$:
                                        ;       CHECK USED BITS OF RMOF FOR ZERO
016004  013737  001360  001142                MOV     RMOFI,$BDDAT    ;GET USED BITS
016012  042737  161577  001142                BIC     #XNUOF,$BDDAT
016020  001403                              BEQ     20$             ;BRANCH IF NO ERROR
016022  104054                              ERROR   54              ;USED BITS RMOF S-A-1
016024  052703  000001                      BIS     #BIT0,R3        ;SET ERROR FLAG
016030                                  20$:

                                        ;;**************************************************************
                                        ;WRITE ZEROS THEN ONES ON RMOF AND CHECK FOR S-A-0 BITS.
016030  012760  000000  000032                MOV     #0,RMOF(R0)     ;LOAD RMOF

016036  012760  177777  000032                MOV     #-1,RMOF(R0)    ;LOAD RMOF

016044  016037  000032  001360                MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
                                        ;       CHECK UNUSED BITS OF RMOF FOR ZERO.
016052  013737  001360  001142                MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
016060  042737  016200  001142                BIC     #^CXNUOF,$BDDAT
016066  001403                              BEQ     30$             ;BRANCH IF NO ERROR
016070  104053                              ERROR   53              ;UNUSED BITS OF RMOF NOT ZERO
016072  052703  000001                      BIS     #BIT0,R3
016076                                  30$:
                                        ;       CHECK USED BITS OF RMOF FOR ONE.
016076  013737  001360  001142                MOV     RMOFI,$BDDAT    ;GET USED BITS
016104  042737  161577  001142                BIC     #XNUOF,$BDDAT
016112  012737  016200  001140                MOV     #FMT16!ECI!HCI!OFD,$GDDAT
016120  023737  001140  001142                CMP     $GDDAT,$BDDAT
016126  001403                              BEQ     40$             ;BRANCH IF NO ERROR
016130  104055                              ERROR   55              ;USED BITS RMOF S-A-0
016132  052703  000002                      BIS     #BIT1,R3
016136                                  40$:

                                        ;;**************************************************************
                                        ;IF NO PREVIOUS ERRORS, TEST RMOF WITH SHIFTING ONE BIT PATTERN.
016136  005703                              TST     R3              ;ANY ERROR??
016140  001025                              BNE     70$             ;YES!!
016142  012702  000001                      MOV     #1,R2           ;STARTING DATA PATTERN
016146                                  50$:

016146  012760  000000  000032                MOV     #0,RMOF(R0)     ;LOAD RMOF
```

```
6534  015170  010260  000032          MOV     R2,RMOF(R0)     ;LOAD RMOF
6535
6536  015174  016037  000032  001142  MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
6537  015200  010203                  MOV     R2,R3           ;SETUP EXPECTED RESULT
6538  015204  042703  161577          BIC     #XMOF,R3        ;CLEAR UNUSED BITS
6539  015210  020337  001142          CMP     R3,$BDDAT       ;COMPARE EXPECTED & RECEIVED
6540  015214  001403                  BEQ     60$             ;BRANCH IF NO ERROR
6541  015216  010337  001140          MOV     R3,$GDDAT       ;LOAD EXPECTED
6542  015222  104056                  ERROR   56              ;CANT WRITE SHIFTING 1 BIT
6543  015224  006302          60$:    ASL     R2              ;SHIFT TO NEXT BIT
6544  015226  001355                  BNE     50$             ;CONTINUE IF NOT DONE
6545
6546  015230          70$:
6547
6548                  ;;*****************************************************************
6549                  ;*TEST 17        SERIAL NUMBER TEST
6550
6551                  ;;*****************************************************************
6552  015230  000004  TST17:  SCOPE
6553  015232  012737  000017  001226  MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6554
6555  015240  000240                  NOP
6556  015242  012737  000024  001120  MOV     #20.,$ICNT      ;20 ITERATIONS
6557  015250  112737  000001  001131  MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6558  015256  012737  015272  001122  MOV     #T17,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6559  015264  012737  015272  001124  MOV     #T17,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6560  015272          T17:
6561  015272  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
6562  015276  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6563  015302  013701  001456          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6564  015306  012760  000040  000010  MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6565  015314  111160  000010          MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6566  015320  010037  001136          MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR TYPEOUT
6567  015324  062737  000030  001136  ADD     #RMSN,$BDADR
6568  015332  012702  000031          MOV     #25.,R2         ;READ RMSN 25 TIMES
6569                  ;READ RMSN AND USE THE RESULT AS EXPECTED VALUE
6570
6571  015336  016037  000030  001140  MOV     RMSN(R0),$GDDAT ;STORE RMSN AT $GDDAT
6572  015344          10$:
6573                  ;READ RMSN AND COMPARE WITH INITIAL VALUE
6574
6575  015344  016037  000030  001142  MOV     RMSN(R0),$BDDAT ;STORE RMSN AT $BDDAT
6576  015352  023737  001140  001142  CMP     $GDDAT,$BDDAT
6577  015360  001401                  BEQ     20$             ;BRANCH IF SERIAL NUMBER CONSISTENT
6578  015362  104136                  ERROR   136             ;SERIAL NUMBER INCONSISTENT
6579  015364          20$:
6580                  ;DECREMENT COUNT AND CONTINUE IF NOT DONE
6581  015364  005302                  DEC     R2
6582  015366  100366                  BPL     10$
6583  015370          30$:                                    ;END OF TEST
6584
6585                  ;;*****************************************************************
6586                  ;*TEST 20        CONTROL BUS PARITY DETECTION TEST
6587
6588                  ;;*****************************************************************
6589  015370  000004  TST20:  SCOPE
```

```
6590  015372  012737  000020  001226          MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6591
6592  015400  000240                          NOP
6593  015402  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
6594  015410  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6595  015416  012737  015432  001122          MOV     #T20,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6596  015424  012737  015432  001124          MOV     #T20,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6597  015432                              T20:
6598  015432  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
6599  015436  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6600  015442  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6601                                      ;SETUP FOR FIRST TEST LOOP (NO ERROR)
6602  015446  005037  001140                  CLR     $GDDAT          ;"PAR" SHOULD BE ZERO
6603  015452  111137  001412                  MOVB    (R1),RMCS20     ;SETUP RMCS2 VALUE
6604  015456  042737  177770  001412          BIC     #^CUNTMSK,RMCS20
6605  015464  012737  000001  001440          MOV     #1,RMHR0        ;INITIALIZE DATA PATTERN
6606  015472  000402                          BR      6$              ;SKIP INCREMENT FIRST TIME
6607  015474  006337  001440              5$: ASL     RMHR0           ;SHIFT TO NEXT PATTERN
6608  015500                              6$:
6609                                      ;CLEAR AND VERIFY THAT "PAR" IS RESET
6610  015500  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6611  015506  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6612
6613  015512  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
6614
6615  015520  016037  000042  001370          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
6616  015526  042737  177767  001142          BIC     #^CPAR,$BDDAT   ;DID "PAR" RESET?
6617  015534  001415                          BEQ     20$             ;YES!!
6618  015536  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
6619  015542  062737  000014  001136          ADD     #RMER1,$BDADR
6620  015550  032737  000010  001370          BIT     #DPE,RMER2I     ;IS "DPE" SET??
6621  015556  001002                          BNE     10$             ;YES!!
6622  015560  104067                          ERROR   67              ;CANT CLEAR "PAR"
6623  015562  000453                          BR      50$
6624  015564  104070                  10$:    ERROR   70              ;CANT CLEAR "PAR", "DPE"
6625  015566  000451                          BR      50$
6626  015570                              20$:
6627
6628                                      ;WRITE TEST PATTERN AND VERIFY "PAR" STATUS
6629
6630  015570  013760  001412  000010          MOV     RMCS20,RMCS2(R0)        ;LOAD RMCS2
6631
6632  015576  013760  001440  000036          MOV     RMHR0,RMHR(R0)  ;LOAD RMHR
6633
6634  015604  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
6635  015612  042737  177767  001142          BIC     #^CPAR,$BDDAT
6636  015620  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "PAR" CORRECT??
6637  015626  001410                          BEQ     40$             ;YES!!
6638  015630  032737  000020  001412          BIT     #PAT,RMCS20     ;SHOULD "PAR" BE SET?
6639  015636  001002                          BNE     30$             ;YES!!
6640  015640  104071                          ERROR   71              ;"PAR" SHOULD NOT BE SET
6641  015642  000423                          BR      50$
6642  015644  104072                  30$:    ERROR   72              ;"PAR" SHOULD BE SET
6643  015646  000421                          BR      50$             ;SKIP TO NEXT
6644                                      ;GO TO NEXT PATTERN
6645  015650  005737  001440              40$:    TST     RMHR0           ;IS DATA PATTERN COMPLETE??
```

```
                                        BNE     5$                      ;NO!!
                                        BIT     #PAT,RMCS20             ;IS TEST COMPLETE??
                                        BNE     50$                     ;YES!!
                                ;SETUP FOR SECOND TEST LOOP (ERROR)
                                        BIS     #PAT,RMCS20             ;TURN ON BAD PARITY
                                        MOV     #PAR,$GDDAT             ;EXPECT ERROR
                                        MOV     #1,RMHRO                ;INITIALIZE DATA PATTERN
                                        BR      6$                      ;START LOOP

                                50$:                                    ;END OF TEST

                                ;;*****************************************************************
                                ;*TEST 21        CONTROL BUS PARITY GENERATION TEST

                                ;;*****************************************************************
                                TST21:  SCOPE
                                        MOV     #21,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

                                        NOP
                                        MOV     #20.,$ICNT              ;20 ITERATIONS
                                        MOVB    #1,$ERMAX               ;ONE ERROR ALLOWED
                                        MOV     #T21,$LPADR     ;LOAD LOOP ON TEST ADDRESS
                                        MOV     #T21,$LPERR     ;LOAD LOOP ON ERROR ADDRESS
                                T21:
                                        MOV     #STACK,SP               ;LOAD THE STACK POINTER
                                        MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS OF UUT
                                        MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE

                                ;SETUP FOR TEST (NO ERROR)
                                        MOV     #1,RMHRO                ;INITIALIZE DATA PATTERN
                                        CLR     $GDDAT                  ;MCPE SHOULD BE ZERO
                                        BR      20$                     ;DONT SHIFT FIRST TIME
                                10$:
                                ;SHIFT  ATA PATTERN
                                        ASL     RMHRO
                                20$:
                                ;CLEAR, THEN TRANSFER DATA TO RM03
                                        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
                                        MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

                                        MOV     RMHRO,RMHR(R0)  ;LOAD RMHR
                                ;TRANSFER DATA TO RH, VERIFY NO "MCPE" ERROR

                                        MOV     RMHR(R0),RMHRI  ;STORE RMHR IN INPUT BUFFER

                                        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
                                        BIC     #^CMCPE,$BDDAT  ;WAS BAD PARITY DETECTED??
                                        BEQ     30$                     ;NO!!
                                        ERROR   73                      ;BAD PARITY DETECTED BY RH
                                        BR      40$

                                30$:
                                ;GO TO NEXT PATTERN
                                        TST     RMHRO                   ;DONE ALL PATTERNS??
                                        BNE     10$                     ;NO!!
```

```
6702  016066                                  40$:                            ;END OF TEST
6703
6704                                          ;;*****************************************************************
6705                                          ;*TEST 22       RMDA,RMDC FAULT TEST
6706
6707                                          ;;*****************************************************************
6708  016066  000004                          T$T22:  SCOPE
6709  016070  012737  000022  001226                  MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6710
6711  016076  000240                                  NOP
6712  016100  012737  000024  001120                  MOV     #20.,$ICNT      ;20 ITERATIONS
6713  016106  112737  000001  001131                  MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6714  016114  012737  016130  001122                  MOV     #T22,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6715  016122  012737  016130  001124                  MOV     #T22,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6716  016130                                  T22:
6717  016130  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
6718  016134  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6719  016140  013701  001456                          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6720  016144  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6721  016152  111160  000010                          MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6722
6723                                          ;WRITE ZEROS, THEN ONES IN RMDA,RMDC-READ AND TEST FOR S-A-0
6724
6725  016156  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
6726
6727  016164  012760  000000  000034                  MOV     #0,RMDC(R0)     ;LOAD RMDC
6728
6729  016172  012760  177777  000006                  MOV     #-1,RMDA(R0)    ;LOAD RMDA
6730
6731  016200  012760  177777  000034                  MOV     #-1,RMDC(R0)    ;LOAD RMDC
6732
6733  016206  016037  000006  001334                  MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
6734
6735  016214  016037  000034  001362                  MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
6736  016222  022737  177777  001334                  CMP     #-1,RMDAI       ;IS ANY REGISTER ALL ONES??
6737  016230  001410                                  BEQ     10$             ;YES!!
6738  016232  052737  176000  001362                  BIS     #XNUDC,RMDCI    ;SET UNUSED BITS
6739  016240  022737  177777  001362                  CMP     #-1,RMDCI
6740  016246  001401                                  BEQ     10$             ;YES!!
6741  016250  104042                                  ERROR   42              ;RMDC AND RMDA S-A-0
6742
6743  016252                                  10$:
6744
6745
6746
6747                                          ;;*****************************************************************
6748                                          ;*TEST 23       DISK ADDRESS TRANSFER TEST
6749
6750                                          ;;*****************************************************************
6751  016252  000004                          T$T23:  SCOPE
6752  016254  012737  000023  001226                  MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6753  016262  000240                                  NOP
6754  016264  012737  000024  001120                  MOV     #20.,$ICNT      ;20 ITERATIONS
6755  016272  112737  000001  001131                  MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6756  016300  012737  016314  001122                  MOV     #T23,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6757  016306  012737  016314  001124                  MOV     #T23,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
```

```
 6758  016314                                  T23:
 6759  016314  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
 6760  016320  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 6761  016324  013701  001456                          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 6762  016330  005003                                  CLR     R3              ;R3 = ERROR INDICATOR
 6763  016332  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 6764  016340  111160  000010                          MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 6765                                  ;WRITE ONES IN RMDA, THEN WRITE ZEROS, READ BACK AND CHECK FOR S-A-1 BITS
 6766
 6767  016344  012760  177777  000006                  MOV     #-1,RMDA(R0)    ;LOAD RMDA
 6768
 6769  016352  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
 6770
 6771  016360  016037  000006  001142                  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
 6772  016366  005737  001142                          TST     $BDDAT
 6773  016372  001412                                  BEQ     10$
 6774  016374  005037  001140                          CLR     $GDDAT
 6775  016400  010037  001136                          MOV     R0,$BDADR
 6776  016404  062737  000006  001136                  ADD     #RMDA,$BDADR
 6777  016412  104020                                  ERROR   20
 6778  016414  052703  000001                          BIS     #BIT0,R3        ;SET ERROR FLAG
 6779                                  ;WRITE ZEROS IN RMDA, THEN WRITE ONES, READ BACK AND CHECK FOR S-A-0 BITS
 6780  016420                          10$:
 6781
 6782  016420  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
 6783
 6784  016426  012760  177777  000006                  MOV     #-1,RMDA(R0)    ;LOAD RMDA
 6785
 6786  016434  016037  000006  001142                  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
 6787  016442  023727  001142  177777                  CMP     $BDDAT,#-1
 6788  016450  001413                                  BEQ     20$
 6789  016452  012737  177777  001140                  MOV     #-1,$GDDAT
 6790  016460  010037  001136                          MOV     R0,$BDADR
 6791  016464  062737  000006  001136                  ADD     #RMDA,$BDADR
 6792  016472  104021                                  ERROR   21
 6793  016474  052703  000002                          BIS     #BIT1,R3        ;SET ERROR FLAG
 6794                                  ;WRITE A SHIFTING 1 BIT PATTERN IN RMDA, READ, AND CHECK FOR STUCK BITS
 6795  016500  005703                  20$:    TST     R3              ;OMIT BIT TEST IF ANY ERROR
 6796  016502  001027                          BNE     50$
 6797  016504  012702  000001                  MOV     #1,R2           ;R2 = TEST PATTERN
 6798  016510                          30$:
 6799
 6800  016510  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
 6801
 6802  016516  010260  000006                          MOV     R2,RMDA(R0)     ;LOAD RMDA
 6803
 6804  016522  016037  000006  001142                  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
 6805  016530  023702  001142                          CMP     $BDDAT,R2
 6806  016534  001410                                  BEQ     40$
 6807  016536  010237  001140                          MOV     R2,$GDDAT
 6808  016542  010037  001136                          MOV     R0,$BDADR
 6809  016546  062737  000006  001136                  ADD     #RMDA,$BDADR
 6810  016554  104022                                  ERROR   22
 6811  016556  006302                  40$:    ASL     R2              ;SHIFT TO NEXT BIT
 6812  016560  001353                          BNE     30$             ;CONTINUE IF R2 NOT ZERO
 6813  016562                          50$:
```

```
                                        ;;*****************************************************************
                                        ;*TEST 24        DESIRED CYLINDER TRANSFER TEST
                                        ;;*****************************************************************
016562  000004                  ↑ST24:  SCOPE
016564  012737  000024  001226          MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
016572  000240                          NOP
016574  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
016602  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
016610  012737  016624  001122          MOV     #T24,$LPADR  ;LOAD LOOP ON TEST ADDRESS
016616  012737  016624  001124          MOV     #T24,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
016624                          T24:
016624  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
016630  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
016634  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
016640  005003                          CLR     R3              ;RESET ERROR FLAGS
016642  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
016650  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
016654  005037  001140                  CLR     $GDDAT          ;LOAD EXPECTED
016660  010037  001136                  MOV     R0,$BDADR       ;LOAD REG ADDRESS
016664  062737  000034  001136          ADD     #RMDC,$BDADR
                                        ;WRITE ONES IN RMDC AND VERIFY THAT UNUSED BITS ARE ZERO
016672  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
016700  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
016706  042737  001777  001142          BIC     #↑CXNUDC,$BDDAT         ;CLEAR ALL USED BITS
016714  001403                          BEQ     5$                      ;BRANCH IF NO ERROR
016716  104134                          ERROR   134             ;UNUSED BITS OF RMDC NOT ZERO
016720  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
016724                          5$:
                                        ;WRITE ONES IN RMDC, THEN WRITE ZEROS, READ AND CHECK FOR S-A-1 BITS
016724  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
016732  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
016740  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
016746  042737  176000  001142          BIC     #XNUDC,$BDDAT   ;CLEAR UNUSED BITS
016754  001403                          BEQ     10$             ;BRANCH IF NO ERROR
016756  104037                          ERROR   37              ;CANT WRITE ONES RMDC
016760  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
016764                          10$:
                                        ;WRITE ZEROS, THEN ONES IN RMDC, READ AND CHECK FOR S-A-0 BITS
016764  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
016772  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
017000  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
017006  052737  176000  001142          BIS     #XNUDC,$BDDAT   ;SET UNUSED BITS
017014  012737  177777  001140          MOV     #-1,$GDDAT      ;LOAD EXPECTED RESULT
017022  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS RMDC ALL ONES ??
```

```
6870  017030  001403                          BEQ     20$             ;YES !!
6871  017032  104040                          ERROR   40
6872  017034  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
6873  017040                          20$:
6874                                  ;OMIT BIT TEST IF ANY ERRORS
6875  017040  005703                          TST     R3
6876  017042  001026                          BNE     60$
6877  017044  012702  000001                  MOV     #1,R2           ;R2 = TEST PATTERN
6878  017050                          30$:
6879                                  ;TEST RMDC WITH SHIFTING ONE BIT
6880
6881  017050  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
6882
6883  017056  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
6884  017062  010203                          MOV     R2,R3           ;R3 = EXPECTED RESULT
6885  017064  042703  176000                  BIC     #XMUDC,R3       ;CLEAR ANY UNUSED BITS
6886
6887  017070  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
6888  017076  023703  001142                  CMP     $BDDAT,R3
6889  017102  001404                          BEQ     50$
6890  017104  010337  001140                  MOV     R3,$GDDAT
6891  017110  104041                          ERROR   41
6892  017112  000402                          BR      60$             ;SKIP TO NEXT
6893
6894  017114  006302                  50$:    ASL     R2              ;SHIFT TO NEXT BIT
6895  017116  001354                          BNE     30$             ;CONTINUE IF R2 NOT ZERO
6896
6897  017120                          60$:
6898
6899                                  ;;*****************************************************************
6900                                  ;*TEST 25        ILLEGAL REGISTER TEST
6901
6902                                  ;;*****************************************************************
6903  017120  000004                  T25:    SCOPE
6904  017122  012737  000025  001226          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6905  017130  000240                          NOP
6906  017132  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
6907  017140  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
6908  017146  012737  017162  001122          MOV     #T25,$LPADR  ;LOAD LOOP ON TEST ADDRESS
6909  017154  012737  017162  001124          MOV     #T25,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
6910  017162                          T25:
6911  017162  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
6912  017166  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
6913  017172  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
6914  017176  005037  001140                  CLR     $GDDAT          ;"ILR" SHOULD BE ZERO
6915  017202  012702  000000                  MOV     #0,R2           ;R2=REGISTER INDEX
6916  017206                          10$:
6917                                  ;;*****************************************************************
6918                                  ;CLEAR AND VERIFY THAT "ILR" STATUS IS ZERO
6919  017206  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
6920  017214  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
6921
6922  017220  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
6923  017226  042737  177775  001142          BIC     #^CILR,$BDDAT
6924  017234  001411                          BEQ     20$             ;BRANCH IF ILR IS RESET
6925  017236  005037  001140                  CLR     $GDDAT          ;SETUP GOOD DATA, REG ADR
```

```
6926  017242  010037  001136                MOV     R0,$BDADR
6927  017246  062737  000014  001136        ADD     #RMER1,$BDADR
6928  017254  104064                        ERROR   64                      ;CANT CLEAR ILR
6929  017256  000551                        BR      140$
6930  017260                        20$:
6931                                ;;**************************************************************************
6932                                ;WRITE THE REGISTER (INDEX=R2) AND TEST ILR STATUS
6933  017260  010003                        MOV     R0,R3                   ;R3=REG ADDRESS
6934  017262  060203                        ADD     R2,R3
6935  017264  013746  000004                MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
6936  017270  013746  000006                MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
6937  017274  012737  017336  000004        MOV     #40$,ERRVEC             ;SETUP FOR BUS TIMEOUT
6938  017302  012737  000300  000006        MOV     #PR6,ERRVEC+2
6939  017310  005004                        CLR     R4                      ;R4=REGISTER VALUE
6940  017312  022702  000010                CMP     #RMCS2,R2
6941  017316  001001                        BNE     30$
6942  017320  111104                        MOVB    (R1),R4                 ;SELECT DRIVE IF RMCS2
6943  017322  010413                30$:    MOV     R4,(R3)                 ;WRITE TEST REGISTER
6944  017324  012637  000006                MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
6945  017330  012637  000004                MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
6946  017334  000416                        BR      60$
6947  017336  012716  017344        40$:    MOV     #45$,(SP)               ;DUMMY RTI ADDRESS
6948  017342  000002                        RTI                             ;RESTORE PRIORITY
6949  017344                        45$:
6950  017344  012637  000006                MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
6951  017350  012637  000004                MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
6952  017354  020227  000046                CMP     R2,#RMEC2               ;WERE ALL RM03 REGISTERS READ??
6953  017360  101003                        BHI     50$                     ;YES!!
6954  017362  010337  001136                MOV     R3,$BDADR
6955  017366  104074                        ERROR   74                      ;UNEXPECTED BUS TIMEOUT
6956  017370  000504                50$:    BR      140$
6957
6958  017372                        60$:
6959
6960  017372  016037  000014  001142        MOV     RMER1(R0),$BDDAT                ;STORE RMER1 AT $BDDAT
6961  017400  042737  177775  001142        BIC     #^CILR,$BDDAT
6962  017406  023737  001140  001142        CMP     $GDDAT,$BDDAT           ;IS "ILR" OK??
6963  017414  001411                        BEQ     80$                     ;YES!!
6964  017416  010337  001174                MOV     R3,$TMP0                ;SAVE ADDRESS
6965  017422  032737  000002  001140        BIT     #ILR,$GDDAT             ;SHOULD "ILR" BE SET??
6966  017430  001002                        BNE     70$                     ;YES!!
6967  017432  104065                        ERROR   65                      ;"ILR" SHOULD NOT BE SET
6968  017434  000401                        BR      80$
6969  017436  104066                70$:    ERROR   66                      ;"ILR" SHOULD BE SET
6970
6971  017440                        80$:
6972                                ;ADVANCE TO THE NEXT REGISTER ADDRESS
6973  017440  062702  000002                ADD     #2,R2                   ;INCREMENT INDEX
6974  017444  022702  000050                CMP     #RMBAE,R2               ;IS THIS RMBAE??
6975  017450  001033                        BNE     100$                    ;NO!!
6976  017452  013746  000004                MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
6977  017456  013746  000006                MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
6978  017462  012737  017564  000004        MOV     #130$,ERRVEC            ;SETUP FOR TIMEOUT
6979  017470  012737  000300  000006        MOV     #PR6,ERRVEC+2
6980
6981  017476  012760  001400  000000        MOV     #A17!A16,RMCS1(R0)              ;LOAD RMCS1
```

```
6982
6983  017504  016037  000050  001376          MOV     RMBAE(R0),RMBAEI        ;STORE RMBAE IN REGISTER INPUT BUFFER
6984  017512  042737  177774  001376          BIC     #^C<BIT1!BIT0>,RMBAEI
6985  017520  001003                          BNE     90$             ;BRANCH IF RH70 CONTROLLER
6986  017522  052737  000002  001140          BIS     #ILR,$GDDAT     ;"ILR" SHOULD BE SET
6987  017530                          90$:
6988  017530  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
6989  017534  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
6990  017540  022702  000054          100$:   CMP     #RMCS3+2,R2     ;DONE ALL LEGAL REGISTERS
6991  017544  001003                          BNE     110$
6992  017546  052737  000002  001140          BIS     #ILR,$GDDAT     ;"ILR" SHOULD BE SET
6993  017554  022702  000076          110$:   CMP     #76,R2          ;DONE ALL TESTS
6994  017560  103410                          BLO     140$            ;YES!!
6995  017562  000611                  120$:   BR      10$
6996
6997  017564  012716  017572          130$:   MOV     #135$,(SP)      ;DUMMY RTI ADDRESS
6998  017570  000002                          RTI                     ;RESTORE PRIORITY
6999  017572                          135$:
7000  017572  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
7001  017576  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
7002  017602                          140$:                           ;END OF TEST
7003
7004                                  ;;**********************************************************************
7005                                  ;*TEST 26       RESET GO BY INIT TEST
7006
7007                                  ;;**********************************************************************
7008  017602  000004                  T$T26:  SCOPE
7009  017604  012737  000026  001226          MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7010
7011  017612  000240                          NOP
7012  017614  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
7013  017622  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7014  017630  012737  017644  001122          MOV     #T26,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7015  017636  012737  017644  001124          MOV     #T26,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7016  017644                          T26:
7017  017644  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7018  017650  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7019  017654  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7020  017660  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7021  017666  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7022  017672  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR MSG
7023                                  ;SET GO, INITIALIZE AND VERIFY THAT GO IS RESET
7024
7025  017676  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
7026  017704  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7027  017712  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7028
7029  017716  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
7030  017724  042737  177776  001142          BIC     #^CGO,$BDDAT
7031  017732  001403                          BEQ     10$             ;BRANCH IF GO IS RESET
7032  017734  005037  001140                  CLR     $GDDAT
7033  017740  104137                          ERROR   137             ;GO NOT CLEARED BY INIT
7034
7035  017742                          10$:                            ;END OF TEST
7036
7037                                  ;;**********************************************************************
```

```
                                        ;*TEST 27       DIAGNOSTIC MODE TEST
                                        ;
                                        ;;*****************************************************************
020072                                  TST27:  SCOPE
017744  012737  000027  001226                  MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                        ;
017752  000240                          T27:    NOP
017754  012737  000024  001120                  MOV     #20.,$ICNT      ;20 ITERATIONS
017762  112737  000001  001131                  MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
017770  012737  020004  001122                  MOV     #T27,$LPADR  ;LOAD LOOP ON TEST ADDRESS
017776  012737  020004  001124                  MOV     #T27,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
020004
020004  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
020010  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
020014  013701  001456                          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
020020  010037  001136                          MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
020024  062737  000024  001136                  ADD     #RMMR1,$BDADR
020032  005003                                  CLR     R3              ;INITIALIZE ERROR FLAGS
                                        ;INITIALIZE AND VERIFY THAT "DMD" IS RESET
020034  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
020042  111160  000010                          MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
                                        ;
020046  016037  000024  001142                  MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
020054  042737  177776  001142                  BIC     #^CDMD,$BDDAT
020062  001403                                  BEQ     10$             ;BRANCH IF "DMD" IS ZERO
020064  005037  001140                          CLR     $GDDAT
020070  104075                                  ERROR   75              ;CANT CLEAR "DMD"
020072                                  10$:
                                        ;SET AND RESET "DMD" USING REGISTER TRANSFER-VERIFY "DMD" NOT S-A-1
                                        ;
020072  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
                                        ;
020100  012760  000000  000024                  MOV     #0,RMMR1(R0)    ;LOAD RMMR1
                                        ;
020106  016037  000024  001142                  MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
020114  042737  177776  001142                  BIC     #^CDMD,$BDDAT
020122  001405                                  BEQ     20$             ;BRANCH IF DMD NOT S-A-1
020124  005037  001140                          CLR     $GDDAT
020130  104076                                  ERROR   76              ;CANT WRITE DMD=0
020132  052703  000001                          BIS     #BIT0,R3        ;SET ERROR FLAG
020136                                  20$:
                                        ;RESET AND SET "DMD" USING REGISTER TRANSFER-VERIFY "DMD" NOT S-A-0
                                        ;
020136  012760  000000  000024                  MOV     #0,RMMR1(R0)    ;LOAD RMMR1
                                        ;
020144  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
                                        ;
020152  016037  000024  001142                  MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
020160  042737  177776  001142                  BIC     #^CDMD,$BDDAT
020166  001006                                  BNE     30$             ;BRANCH IF DMD NOT S-A-0
020170  012737  000001  001140                  MOV     #DMD,$GDDAT
020176  104077                                  ERROR   77              ;CAN'T WRITE DMD=1
020200  052703  000002                          BIS     #BIT1,R3        ;SET ERROR FLAG
020204                                  30$:
                                        ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE WITH SHIFTING
                                        ;ONE BIT PATTERN
```

```
7094  020204  005703                          TST     R3              ;ANY ERRORS DETECTED??
7095  020206  001027                          BNE     60$             ;YES!!
7096  020210  012702  000002                  MOV     #BIT1,R2        ;INITAILIZE DATA PATTERN
7097  020214                          40$:
7098
7099  020214  012760  000000  000024          MOV     #0,RMMR1(R0)    ;LOAD RMMR1
7100
7101  020222  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7102
7103  020226  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
7104  020234  042737  177776  001142          BIC     #^CDMD,$BDDAT
7105  020242  001407                          BEQ     50$             ;BRANCH IF DMD NOT SET
7106  020244  010237  001174                  MOV     R2,$TMP0        ;SAVE DATA
7107  020250  010237  001174                  MOV     R2,$TMP0        ;SAVE DATA
7108  020254  005037  001140                  CLR     $GDDAT          ;DMD SHOULD BE ZERO
7109  020260  104100                          ERROR   100             ;DMD SET BY WRONG BIT
7110  020262                          50$:
7111                                  ;       SHIFT TO NEXT DATA BIT AND CONTINUE TEST IF NOT DONE
7112  020262  006302                          ASL     R2
7113  020264  001353                          BNE     40$
7114  020266                          60$:                            ;END OF TEST
7115                                  ;;**********************************************************************
7116                                  ;*TEST 30       MOL TEST
7117
7118                                  ;;**********************************************************************
7119  020266  000004                  TST30:  SCOPE
7120  020270  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7121
7122  020276  000240                          NOP
7123  020300  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
7124  020306  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7125  020314  012737  020330  001122          MOV     #T30,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7126  020322  012737  020330  001124          MOV     #T30,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7127  020330                          T30:
7128  020330  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7129  020334  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7130  020340  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7131  020344  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7132  020352  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7133  020356  010037  001136                  MOV     R0,$BDADR       ;R0=REGISTER ADDRESS
7134  020362  062737  000012  001136          ADD     #RMDS,$BDADR
7135  020370  005003                          CLR     R3              ;R3=ERROR FLAG
7136                                  ;SET DIAGNOSTIC MODE AND VERIFY THAT "MOL" IS ZERO
7137
7138  020372  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7139
7140  020400  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7141  020406  042737  167777  001142          BIC     #^CMOL,$BDDAT
7142  020414  001405                          BEQ     10$
7143  020416  005037  001140                  CLR     $GDDAT
7144  020422  104101                          ERROR   101             ;"MOL" NOT ZERO
7145  020424  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7146  020430                          10$:
7147                                  ;SET MAINTENANCE UNIT READY AND VERIFY THAT "MOL" IS ONE
7148
7149  020430  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
```

```
7150
7151  020436  012760  001001  000024          MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
7152
7153  020444  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7154  020452  042737  167777  001142          BIC     #↑CMOL,$BDDAT
7155  020460  001006                          BNE     20$
7156  020462  012737  010000  001140          MOV     #MOL,$GDDAT
7157  020470  104102                          ERROR   102             ;"MOL" NOT ONE
7158  020472  052703  000002                  BIS     #BIT1,R3
7159  020476                          20$:
7160                                  ;IF NO PREVIOUS ERROR, VERIFY VOLUME VALID IS RESET AND
7161                                  ;TEST FOR BIT INTERFERENCE
7162  020476  005703                          TST     R3              ;ANY ERROR DETECTED??
7163  020500  001057                          BNE     70$             ;YES!!
7164
7165  020502  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7166  020510  042737  177677  001142          BIC     #↑CVV,$BDDAT
7167  020516  001403                          BEQ     25$             ;BRANCH IF VV RESET
7168  020520  005037  001140                  CLR     $GDDAT
7169  020524  104135                          ERROR   135             ;VV NOT RESET
7170  020526                          25$:
7171  020526  012702  000001                  MOV     #1,R2           ;INITIALIZE DATA PATTERN
7172  020532                          30$:
7173
7174  020532  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7175
7176  020540  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7177
7178  020544  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7179  020552  042737  167777  001142          BIC     #↑CMOL,$BDDAT
7180  020560  005003                          CLR     R3              ;SETUP EXPECTED "MOL"
7181  020562  032702  001000                  BIT     #MUR,R2
7182  020566  001402                          BEQ     40$
7183  020570  052703  010000                  BIS     #MOL,R3         ;"MOL" SHOULD BE ONE
7184  020574  020337  001142          40$:    CMP     R3,$BDDAT       ;IS MOL OK??
7185  020600  001405                          BEQ     50$             ;YES!!
7186  020602  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
7187  020606  010337  001140                  MOV     R3,$GDDAT
7188  020612  104103                          ERROR   103             ;MUR SET BY WRONG BIT
7189  020614                          50$:
7190                                  ;SHIFT TO NEXT PATTERN
7191  020614  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
7192  020620  001002                          BNE     60$
7193  020622  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
7194  020626  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
7195  020630  001403                          BEQ     70$             ;BRANCH IF DONE
7196  020632  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
7197  020636  000735                          BR      30$             ;CONTINUE
7198
7199  020640                          70$:                            ;END OF TEST
7200
7201                                  ;;*****************************************************************
7202                                  ;*TEST 31        WRITE LOCK TEST
7203
7204                                  ;;*****************************************************************
7205  020640  000004                  TST31:  SCOPE
```

```
7206  020642  012737  000031  001226          MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7207
7208  020650  000240                          NOP
7209  020652  012737  000024  001120          MOV     #20.,$ICNT      ;20 IIERATIONS
7210  020660  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7211  020666  012737  020702  001122          MOV     #T31,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7212  020674  012737  020702  001124          MOV     #T31,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7213  020702                              T31:
7214  020702  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7215  020706  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7216  020712  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7217  020716  005003                          CLR     R3              ;R3=ERROR FLAG
7218  020720  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7219  020724  062737  000012  001136          ADD     #RMDS,$BDADR
7220  020732  005037  001140                  CLR     $GDDAT          ;WRL SHOULD BE ZERO
7221  020736  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7222  020744  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7223                              ;SET DIAGNOSTIC MODE AND VERIFY "WRL" IS ZERO
7224
7225  020750  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7226
7227  020756  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7228  020764  042737  173777  001142          BIC     #^CWRL,$BDDAT
7229  020772  001403                          BEQ     10$             ;BRANCH IF WRL IS ZERO
7230  020774  104104                          ERROR   104             ;WRL NOT ZERO
7231  020776  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7232  021002                              10$:
7233                              ;SET MAINTENANCE WRITE PROTECT AND VERIFY "WRL" IS ONE
7234
7235  021002  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7236
7237  021010  012760  000011  000024          MOV     #DMD!MWP,RMMR1(R0)      ;LOAD RMMR1
7238
7239  021016  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7240  021024  042737  173777  001142          BIC     #^CWRL,$BDDAT
7241  021032  001006                          BNE     20$             ;BRANCH IF WRL IS NOE
7242  021034  012737  004000  001140          MOV     #WRL,$GDDAT     ;WRL SHOULD BE SET
7243  021042  104105                          ERROR   105             ;CANT SET WRL
7244  021044  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
7245  021050                              20$:
7246                              ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE ON "MWP"
7247  021050  005703                          TST     R3              ;ANY OTHER ERROR??
7248  021052  001045                          BNE     70$             ;YES!!
7249  021054  012702  000001                  MOV     #1,R2           ;INITIALIZE DATA PATTERN
7250
7251  021060                              30$:
7252                              ;       TRANSFER DATA TO RMMR1, READ RMDS AND VERIFY WRL
7253
7254  021060  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7255
7256  021066  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7257
7258  021072  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7259  021100  042737  173777  001142          BIC     #^CWRL,$BDDAT   ;CLEARUP RECEIVED "WRL"
7260  021106  005003                          CLR     R3              ;GENERATE EXPECTED "WRL"
7261  021110  032702  000010                  BIT     #MWP,R2
```

```
7262  021114  001402                          BEQ     40$
7263  021116  052703  004000                  BIS     #WRL,R3         ;WRL SHOULD BE SET
7264  021122  020337  001142          40$:    CMP     R3,$BDDAT       ;IS WRL OK??
7265  021126  001405                          BEQ     50$             ;YES!!
7266  021130  010337  001140                  MOV     R3,$GDDAT       ;SAVE EXPECTED
7267  021134  010237  001174                  MOV     R2,$TMP0        ;SAVE DATA PATTERN
7268  021140  104106                          ERROR   106             ;BIT INTERFERENCE WITH MWP
7269  021142                          50$:
7270                                  ;       ADVANCE TO NEXT DATA BIT
7271  021142  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
7272  021146  001002                          BNE     60$
7273  021150  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
7274  021154  006302                  60$:    ASL     R2              ;SHIFT DATA BIT
7275  021156  001403                          BEQ     70$             ;EXIT IF DONE
7276  021160  052702  000001                  BIS     #DMD,R2         ;KEEP DIAGNOSTIC MODE ON
7277  021164  000735                          BR      30$             ;CONTINUE TEST
7278
7279  021166                          70$:                            ;END OF TEST
7280
7281                                  ;;**********************************************************************
7282                                  ;*TEST 32        DRIVE FAULT TEST
7283
7284                                  ;;**********************************************************************
7285  021166  000004                  T$T32:  SCOPE
7286  021170  012737  000032  001226          MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7287
7288  021176  000240                          NOP
7289  021200  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
7290  021206  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7291  021214  012737  021230  001122          MOV     #T32,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7292  021222  012737  021230  001124          MOV     #T32,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7293  021230                          T32:
7294  021230  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7295  021234  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7296  021240  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7297  021244  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7298  021252  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7299  021256  005003                          CLR     R3              ;INITIALIZE ERROR FLAGS
7300  021260  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7301  021264  062737  000042  001136          ADD     #RMER2,$BDADR
7302  021272  005037  001140                  CLR     $GDDAT          ;"DVC" AND "UNS" SHOULD BE ZERO
7303                                  ;SET AND RESET MAINTENANCE DRIVE FAULT, VERIFY THAT "DVC" IS NOT
7304                                  ;STUCK-AT-ONE.
7305
7306  021276  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7307
7308  021304  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
7309
7310  021312  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
7311
7312  021320  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7313  021326  042737  177577  001142          BIC     #^CDVC,$BDDAT   ;IS "DVC" RESET??
7314  021334  001406                          BEQ     10$             ;YES!!
7315  021336  104107                          ERROR   107             ;CANT RESET DVC
7316  021340  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7317  021344  012737  040000  001140          MOV     #UNS,$GDDAT
```

```
7318  021352                                  10$:
7319
7320                                          ;VERIFY THAT "UNS" IS SAME AS "DVC"
7321
7322  021352  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
7323  021360  042737  137777  001142          BIC     #^CUNS,$BDDAT
7324  021366  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "UNS" OK??
7325  021374  001414                          BEQ     30$             ;YES!!
7326  021376  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7327  021402  062737  000014  001136          ADD     #RMER1,$BDADR
7328  021410  032737  040000  001140          BIT     #UNS,$GDDAT     ;SHOULD "UNS" BE ON??
7329  021416  001002                          BNE     20$             ;YES!!
7330  021420  104110                          ERROR   110             ;UNS IS SET, DVS IS RESET
7331  021422  000401                          BR      30$
7332  021424                                  20$:
7333  021424  104111                          ERROR   111             ;UNS IS RESET, DVC IS SET
7334  021426                                  30$:
7335
7336                                          ;RESET AND SET "MDF", VERIFY THAT "DVC" IS NOT S-A-0.
7337  021426  012737  000200  001140          MOV     #DVC,$GDDAT     ;DVC SHOULD BE ON
7338
7339  021434  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7340
7341  021442  012760  000101  000024          MOV     #DMD!MDF,RMMR1(R0)      ;LOAD RMMR1
7342
7343  021450  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7344  021456  042737  177577  001142          BIC     #^CDVC,$BDDAT
7345  021464  001012                          BNE     40$             ;BRANCH IF DVC IS SET
7346  021466  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7347  021472  062737  000042  001136          ADD     #RMER2,$BDADR
7348  021500  104112                          ERROR   112             ;CANT SET DVC
7349  021502  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
7350  021506  005037  001140                  CLR     $GDDAT          ;UNS SHOULD BE OFF
7351  021512                                  40$:
7352
7353                                          ;VERIFY THAT "UNS" IS SAME AS "DVC"
7354  021512  005737  001140                  TST     $GDDAT          ;CHANGE DVC TO UNS
7355  021516  001403                          BEQ     50$
7356  021520  012737  040000  001140          MOV     #UNS,$GDDAT
7357  021526                                  50$:
7358
7359  021526  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
7360  021534  042737  137777  001142          BIC     #^CUNS,$BDDAT
7361  021542  023737  001140  001142          CMP     $GDDAT,$BDDAT
7362  021550  001414                          BEQ     70$             ;BRANCH IF UNS IS OK
7363  021552  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7364  021556  062737  000014  001136          ADD     #RMER1,$BDADR
7365  021564  032737  040000  001140          BIT     #UNS,$GDDAT     ;SHOULD UNS BE ON??
7366  021572  001002                          BNE     60$             ;YES!!
7367  021574  104113                          ERROR   113             ;UNS IS SET, DVC IS RESET
7368  021576  000401                          BR      70$
7369  021600                                  60$:
7370  021600  104114                          ERROR   114             ;UNS IS RESET, DVC IS SET
7371  021602                                  70$:
7372
7373                                          ;IF THERE WERE NO PREVIOUS ERRORS, TEST FOR BIT INTERFERENCE ON
```

```
7374                                        ;"MDF"
7375  021602  005703                        TST     R3
7376  021604  001056                        BNE     120$            ;BRANCH IF ANY OTHER ERRORS
7377  021606  012702  000001                MOV     #1,R2           ;INITIALIZE TEST PATTERN
7378  021612                              80$:
7379
7380  021612  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7381
7382  021620  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
7383
7384  021626  010260  000024                MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7385
7386  021632  016037  000042  001142        MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7387  021640  042737  177577  001142        BIC     #^CDVC,$BDDAT   ;GET RESULTS
7388  021646  005037  001140                CLR     $GDDAT          ;SETUP EXPECTED
7389  021652  032702  000100                BIT     #MDF,R2         ;WAS MDF SET??
7390  021656  001403                        BEQ     90$             ;NO!!
7391  021660  052737  000200  001140        BIS     #DVC,$GDDAT     ;YES-DVC SHOULD BE ON
7392  021666  023737  001140  001142  90$:  CMP     $GDDAT,$BDDAT
7393  021674  001410                        BEQ     100$            ;BRANCH IF DVC IS OK
7394  021676  010037  001136                MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7395  021702  062737  000042  001136        ADD     #RMER2,$BDADR
7396  021710  010237  001174                MOV     R2,$TMP0        ;SAVE TEST PATTERN
7397  021714  104115                        ERROR   115             ;MDF SET BY WRONG BIT
7398  021716                              100$:
7399                                      ;SHIFT TO NEXT BIT POSITION
7400  021716  042702  000001                BIC     #DMD,R2         ;DONT SHIFT DMD
7401  021722  001002                        BNE     110$
7402  021724  012702  000001                MOV     #DMD,R2         ;DONT TRUNCATE TEST
7403  021730  006302                      110$:  ASL     R2              ;SHIFT
7404  021732  001403                        BEQ     120$            ;EXIT IF DONE
7405  021734  052702  000001                BIS     #DMD,R2         ;KEEP DMD ON
7406  021740  000724                        BR      80$             ;CONTINUE
7407
7408  021742                              120$:                           ;END OF TEST
7409
7410                                      ;;*****************************************************************
7411                                      ;*TEST 33        SEEK ERROR TEST
7412
7413                                      ;;*****************************************************************
7414  021742  000004                      T$T33:  SCOPE
7415  021744  012737  000033  001226              MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7416
7417  021752  000240                              NOP
7418  021754  012737  000024  001120              MOV     #20.,$ICNT      ;20 ITERATIONS
7419  021762  112737  000001  001131              MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7420  021770  012737  022004  001122              MOV     #T33,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7421  021776  012737  022004  001124              MOV     #T33,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7422  022004                              T33:
7423  022004  012706  001100                      MOV     #STACK,SP       ;LOAD THE STACK POINTER
7424  022010  013700  001276                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7425  022014  013701  001456                      MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7426  022020  012760  000040  000010              MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7427  022026  111160  000010                      MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7428  022032  005003                              CLR     R3              ;CLEAR ERROR FLAGS
7429  022034  010037  001136                      MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
```

```
7430  022040  062737  000042  001136          ADD     #RMER2,$BDADR
7431
7432                                  ;SET DIAGNOSTIC MODE AND VERIFY THAT "SKI" CAN BE RESET
7433
7434  022046  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7435
7436  022054  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
7437
7438  022062  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7439  022070  042737  137777  001142          BIC     #^CSKI,$BDDAT
7440  022076  001405                          BEQ     10$             ;BRANCH IF SKI IS RESET
7441  022100  005037  001140                  CLR     $GDDAT          ;SKI SHOULD BE ZERO
7442  022104  104116                          ERROR   116             ;CANT RESET "SKI"
7443  022106  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7444  022112                          10$:
7445
7446                                  ;SET MAINTENANCE SEEK ERROR AND VERIFY THAT "SKI" CAN BE SET
7447
7448  022112  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7449
7450  022120  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
7451
7452  022126  012760  000201  000024          MOV     #DMD!MSER,RMMR1(R0)     ;LOAD RMMR1
7453
7454  022134  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7455  022142  042737  137777  001142          BIC     #^CSKI,$BDDAT
7456  022150  001005                          BNE     20$             ;BRANCH IF SKI IS SET
7457  022152  012737  040000  001140          MOV     #SKI,$GDDAT     ;CANT SET SKI
7458  022160  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
7459  022164                          20$:
7460
7461                                  ;IF NO PREVIOUS ERROR, CHECK FOR BIT INTERFERENCE SETTING MAINTENANCE
7462                                  ;SEEK ERROR.
7463  022164  005703                          TST     R3
7464  022166  001051                          BNE     70$             ;BRANCH IF ANY OTHER ERRORS
7465  022170  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
7466  022174                          30$:
7467
7468  022174  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7469
7470  022202  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
7471
7472  022210  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7473
7474  022214  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
7475  022222  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;GET SKI STATUS
7476  022230  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED RESULT
7477  022234  032702  000200                  BIT     #MSER,R2
7478  022240  001403                          BEQ     40$
7479  022242  052737  040000  001140          BIS     #SKI,$GDDAT     ;SKI SHOULD BE ON
7480  022250  023737  001140  001142  40$:    CMP     $GDDAT,$BDDAT
7481  022256  001403                          BEQ     50$             ;BRANCH IF SKI IS OK
7482  022260  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
7483  022264  104120                          ERROR   120             ;SKI SET BY WRONG BIT
7484  022266                          50$:
7485                                  ;       ADVANCE TEST PATTERN IN R2
```

```
7486  022266  042702  000001              BIC     #DMD,R2         ;DONT SHIFT DMD BIT
7487  022272  001002                      BNE     60$
7488  022274  012702  000001              MOV     #DMD,R2         ;DONT TRUNCATE TEST
7489  022300  006302                60$:  ASL     R2              ;SHIFT TO NEXT BIT
7490  022302  001403                      BEQ     70$             ;EXIT IF DONE
7491  022304  052702  000001              BIS     #DMD,R2         ;KEEP DMD ON
7492  022310  000731                      BR      30$             ;CONTINUE TEST
7493
7494  022312                        70$:                          ;END OF TEST
7495
7496                                ;;*****************************************************************
7497                                ;*TEST 34        PIP TEST
7498
7499                                ;;*****************************************************************
7500  022312  000004                TST34:  SCOPE
7501  022314  012737  000034  001226        MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7502
7503  022322  000240                        NOP
7504  022324  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
7505  022332  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7506  022340  012737  022354  001122        MOV     #T34,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7507  022346  012737  022354  001124        MOV     #T34,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7508  022354                        T34:
7509  022354  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
7510  022360  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7511  022364  013701  001456              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7512  022370  012760  000040  000010      MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7513  022376  111160  000010              MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7514  022402  005003                      CLR     R3              ;RESET ERROR FLAGS
7515  022404  010037  001136              MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7516  022410  062737  000012  001136      ADD     #RMDS,$BDADR
7517
7518                                ;SET MAINTENANCE ON CYLINDER "MOC" AND VERIFY THAT "PIP" CAN BE RESET.
7519
7520  022416  012760  000001  000024      MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7521
7522  022424  012760  000401  000024      MOV     #DMD!MOC,RMMR1(R0)      ;LOAD RMMR1
7523
7524  022432  016037  000012  001142      MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7525  022440  042737  157777  001142      BIC     #^CPIP,$BDDAT
7526  022446  001405                      BEQ     10$             ;BRANCH IF PIP IS RESET
7527  022450  005037  001140              CLR     $GDDAT
7528  022454  104121                      ERROR   121             ;CANT RESET PIP
7529  022456  052703  000001              BIS     #BIT0,R3        ;SET ERROR FLAG
7530  022462                        10$:
7531                                ;RESET MAINTENANCE ON CYLINDER AND VERIFY THAT "PIP" CAN BE SET
7532
7533  022462  012760  000001  000024      MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7534
7535  022470  016037  000012  001142      MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7536  022476  042737  157777  001142      BIC     #^CPIP,$BDDAT
7537  022504  001006                      BNE     20$             ;BRANCH IF PIP IS SET
7538  022506  012737  020000  001140      MOV     #PIP,$GDDAT
7539  022514  104122                      ERROR   122             ;CANT SET PIP
7540  022516  052703  000002              BIS     #BIT1,R3        ;SET ERROR FLAG
7541  022522                        20$:
```

```
7542                                        ;IF NO PREVIOUS ERROR, TEST FOR ADJACENT BIT SETTING "MOC"
7543  022522  005703                        TST     R3
7544  022524  001046                        BNE     70$             ;BRANCH IF ANY PREVIOUS ERROR
7545  022526  012702  000001                MOV     #1,R2           ;INITIALIZE TEST PATTERN
7546  022532                              30$:
7547                                        ;       WRITE THE TEST PATTERN, CHECK MOC USING PIP
7548
7549  022532  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7550
7551  022540  010260  000024                MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7552
7553  022544  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
7554  022552  042737  157777  001142        BIC     #^CPIP,$BDDAT   ;GET PIP STATUS
7555  022560  005037  001140                CLR     $GDDAT          ;SETUP EXPECTED RESULT
7556  022564  032702  000400                BIT     #MOC,R2
7557  022570  001003                        BNE     40$
7558  022572  052737  020000  001140        BIS     #PIP,$GDDAT     ;PIP SHOULD BE SET
7559  022600  023737  001140  001142  40$:  CMP     $GDDAT,$BDDAT
7560  022606  001403                        BEQ     50$             ;BRANCH IF PIP OK
7561  022610  010237  001174                MOV     R2,$TMP0        ;SAVE TEST PATTERN
7562  022614  104123                        ERROR   123             ;MOC SET BY WRONG BIT
7563  022616                              50$:
7564                                        ;       ADVANCE THE TEST PATTERN
7565  022616  042702  000001                BIC     #DMD,R2         ;DONT SHIFT DMD
7566  022622  001002                        BNE     60$
7567  022624  012702  000001                MOV     #DMD,R2         ;DONT TRUNCATE TEST
7568  022630  006302                      60$:  ASL     R2              ;SHIFT BIT
7569  022632  001403                        BEQ     70$             ;EXIT IF DONE
7570  022634  052702  000001                BIS     #DMD,R2         ;KEEP DMD ON
7571  022640  000734                        BR      30$             ;CONTINUE TEST
7572
7573  022642                              70$:                            ;END OF TEST
7574
7575                                        ;;*****************************************************************
7576                                        ;#TEST 35       EBL TEST
7577
7578                                        ;;*****************************************************************
7579  022642  000004                      T35:  SCOPE
7580  022644  012737  000035  001226        MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7581
7582  022652  000240                        NOP
7583  022654  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
7584  022662  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7585  022670  012737  022704  001122        MOV     #T35,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7586  022676  012737  022704  001124        MOV     #T35,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7587  022704                              T35:
7588  022704  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
7589  022710  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7590  022714  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7591  022720  005003                        CLR     R3              ;RESET ERROR FLAGS
7592  022722  010037  001136                MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7593  022726  062737  000024  001136        ADD     #RMMR1,$BDADR
7594  022734  005037  001140                CLR     $GDDAT          ;SETUP EXPECTED RESULT
7595                                        ;CLEAR AND VERIFY THAT END OF BLOCK IS RESET
7596  022740  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7597  022746  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
```

```
7598
7599  022752  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
7600  022760  042737  157777  001142          BIC     #↑CEBL,$BDDAT
7601  022766  001403                          BEQ     10$             ;BRANCH IF EBL IS RESET
7602  022770  104124                          ERROR   124             ;CANT CLEAR EBL
7603  022772  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7604  022776                          10$:
7605                                  ;SET AND RESET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-1.
7606
7607  022776  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7608
7609  023004  012760  020001  000024          MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
7610
7611  023012  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7612
7613  023020  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
7614  023026  042737  157777  001142          BIC     #↑CEBL,$BDDAT
7615  023034  001403                          BEQ     20$             ;BRANCH IF EBL IS RESET
7616  023036  104125                          ERROR   125             ;DEBL APPEARS S-A-1
7617  023040  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
7618  023044                          20$:
7619                                  ;RESET AND SET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-0
7620
7621  023044  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7622
7623  023052  012760  020001  000024          MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
7624
7625  023060  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
7626  023066  042737  157777  001142          BIC     #↑CEBL,$BDDAT
7627  023074  001006                          BNE     30$             ;BRANCH IF EBL IS SET
7628  023076  012737  020000  001140          MOV     #EBL,$GDDAT
7629  023104  104126                          ERROR   126             ;CANT SET EBL
7630  023106  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
7631  023112                          30$:
7632                                  ;IF NO PREVIOUS ERRORS, TEST FOR ADJACENT BIT INTERFERENCE ON "DEBL".
7633  023112  005703                          TST     R3
7634  023114  001042                          BNE     70$             ;BRANCH IF ANY ERROR
7635  023116  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
7636  023122                          40$:
7637                                  ;       WRITE, READ AND VERIFY THE TEST PATTERN IN R2
7638
7639  023122  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7640
7641  023130  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
7642
7643  023134  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
7644  023142  042737  157777  001142          BIC     #↑CEBL,$BDDAT
7645  023150  010203                          MOV     R2,R3           ;GENERATE EXPECTED RESULT
7646  023152  042703  157777                  BIC     #↑CEBL,R3
7647  023156  020337  001142                  CMP     R3,$BDDAT
7648  023162  001405                          BEQ     50$             ;BRANCH IF EBL IS OK
7649  023164  010337  001140                  MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
7650  023170  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
7651  023174  104127                          ERROR   127             ;DEBL SET(RESET) BY WRONG BIT
7652  023176                          50$:
7653                                  ;       SHIFT TO NEXT BIT POSITION
```

```
7654  023176  042702  000001          BIC     #DMD,R2         ;DONT SHIFT DMD
7655  023202  001002                  BNE     60$
7656  023204  012702  000001          MOV     #DMD,R2         ;DONT TRUNCATE DMD
7657  023210  006302              60$: ASL     R2              ;SHIFT TO NEXT BIT
7658  023212  001403                  BEQ     70$             ;EXIT IF DONE
7659  023214  052702  000001          BIS     #DMD,R2         ;KEEP DMD ON
7660  023220  000740                  BR      40$             ;CONTINUE TEST
7661
7662  023222                      70$:                        ;END OF TEST
7663
7664                              ;;**************************************************************
7665                              ;*TEST 36       LAST SECTOR, LAST TRACK TEST
7666
7667                              ;;**************************************************************
7668  023222  000004              TST36:  SCOPE
7669  023224  012737  000036  001226      MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7670  023232  000240                      NOP
7671  023234  012737  000024  001120      MOV     #20.,$ICNT      ;20 ITERATIONS
7672  023242  112737  000001  001131      MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7673  023250  012737  023264  001122      MOV     #T36,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7674  023256  012737  023264  001124      MOV     #T36,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7675  023264                      T36:
7676  023264  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
7677  023270  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7678  023274  013701  001456              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7679  023300  012760  000040  000010      MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7680  023306  111160  000010              MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7681  023312  005002                      CLR     R2              ;INITIALIZE TEST PATTERN
7682  023314  010037  001136              MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
7683  023320  062737  000024  001136      ADD     #RMMR1,$BDADR
7684  023326  005037  001434              CLR     RMOF0           ;START IN 18 BIT MODE
7685  023332                      10$:
                                  ;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
                                  ;
                                  ;LST=   002035          002037

7695
7696  023332  013760  001434  000032      MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
7697
7698  023340  010260  000006              MOV     R2,RMDA(R0)     ;LOAD RMDA
7699
7700  023344  016037  000024  001352      MOV     RMMR1(R0),RMMR1I        ;STORE RMMR1 IN INPUT BUFFER
7701  023352  013737  001352  001142      MOV     RMMR1I,$BDDAT   ;VERIFY "LS"
7702  023360  042737  177773  001142      BIC     #^tCLS,$BDDAT
7703  023366  005037  001140              CLR     $GDDAT          ;GENERATE EXPECTED "LS"
7704  023372  032737  010000  001434      BIT     #FMT16,RMOF0    ;16 BIT MODE??
7705  023400  001007                      BNE     20$             ;YES!!
7706  023402  122702  000035              CMPB    #035,R2
7707  023406  001012                      BNE     30$
7708  023410  052737  000004  001140      BIS     #LS,$GDDAT      ;LS SHOULD BE ON-18 BIT MODE
7709  023416  000406                      BR      30$
```

```
7710  023420  122702  000037          20$:    CMPB    #037,R2
7711  023424  001003                          BNE     30$
7712  023426  052737  000004  001140          BIS     #LS,$GDDAT      ;LS SHOULD BE ON-16 BIT MODE
7713  023434  023737  001140  001142  30$:    CMP     $GDDAT,$BDDAT
7714  023442  001407                          BEQ     40$             ;BRANCH IF LS IS CORRECT
7715  023444  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
7716  023450  013737  001434  001176          MOV     RMOF0,$TMP1     ;SAVE OFFSET REGISTER FOR ERROR
7717  023456  104130                          ERROR   130             ;LS NOT CORRECT FOR RMDA
7718  023460  000453                          BR      80$             ;SKIP TO NEXT
7719  023462  013737  001352  001142  40$:    MOV     RMMR1I,$BDDAT   ;VERIFY "LST"
7720  023470  042737  177775  001142          BIC     #^CLST,$BDDAT
7721  023476  005037  001140                  CLR     $GDDAT          ;GENERATE EXPECTED "LST"
7722  023502  032737  010000  001434          BIT     #FMT16,RMOF0    ;16 BIT MODE??
7723  023510  001007                          BNE     50$             ;YES!!
7724  023512  022702  002035                  CMP     #002035,R2
7725  023516  001012                          BNE     60$
7726  023520  052737  000002  001140          BIS     #LST,$GDDAT     ;LST SHOULD BE ON-18 BIT MODE
7727  023526  000406                          BR      60$
7728  023530  022702  002037          50$:    CMP     #002037,R2
7729  023534  001003                          BNE     60$
7730  023536  052737  000002  001140          BIS     #LST,$GDDAT     ;LST SHOULD BE ON-16 BIT MODE
7731  023544  023737  001140  001142  60$:    CMP     $GDDAT,$BDDAT
7732  023552  001404                          BEQ     70$
7733  023554  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
7734  023560  104131                          ERROR   131             ;LST NOT CORRECT FOR RMDA
7735  023562  000412                          BR      80$             ;SKIP TO NEXT
7736
7737  023564                          70$:
7738                                  ;       ADVANCE TO NEXT TEST PATTERN, CHANGE TO 16 BIT MODE IF ALL
7739                                  ;18 BIT TESTS DONE.
7740  023564  005202                          INC     R2              ;INCREMENT PATTERN
7741  023566  001261                          BNE     10$             ;CONTINUE IF NOT DONE
7742  023570  032737  010000  001434          BIT     #FMT16,RMOF0    ;DONE 16 BIT TOO??
7743  023576  001004                          BNE     80$             ;YES!!
7744  023600  012737  010000  001434          MOV     #FMT16,RMOF0    ;DO 16 BIT FORMAT TEST
7745  023606  000651                          BR      10$
7746
7747  023610                          80$:                            ;END OF TEST
7748
7749                                  ;;*****************************************************************
7750                                  ;*TEST 37       RMDA COUNT TEST
7751
7752                                  ;;*****************************************************************
7753  023610  000004                  TST37:  SCOPE
7754  023612  012737  000037  001226          MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7755
7756  023620  000240                          NOP
7757  023622  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
7758  023630  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7759  023636  012737  023652  001122          MOV     #T37,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7760  023644  012737  023652  001124          MOV     #T37,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7761  023652                          T37:
7762  023652  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7763  023656  013700  001272                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7764  023662  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7765  023666  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
```

J13


```
 7766  023672  062737  000006  001136          ADD     #RMDA,$BDADR
 7767  023700  005037  001434                  CLR     RMOF0           ;START WITH 18 BIT FORMAT
 7768  023704  012737  000035  024326          MOV     #035,11O$       ;SETUP LAST SECTOR FOR 18 BIT
 7769  023712  012737  000001  001140          MOV     #1,$GDDAT       ;SETUP FIRST COUNT
 7770                          ;;**************************************************************
```

```
                                ;INCREMENT SECTOR COUNT USING DIAGNOSTIC END OF BLOCK STARTING AT
                                ;SECTOR 0 AND CONTINUING UNTIL TRACK ADDRESS INCREMENTS
```

```
 7773  023720                          10$:
 7774                                  ;       .CLEAR THE MASSBUS
 7775                                  ;       .SET FORMAT
 7776                                  ;       .LOAD SECTOR AND TRACK ADDRESS
 7777                                  ;       .ENABLE DEBUG CLOCK
 7778                                  ;       .SET GO BIT
 7779  023720  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 7780  023726  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 7781
 7782  023732  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 7783
 7784  023740  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 7785
 7786  023746  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 7787
 7788  023754  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 7789
 7790  023762  013760  001434  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
 7791  023770  013737  001434  001174          MOV     RMOF0,$TMP0     ;SAVE OFFSET FOR ERROR TYPE
 7792
 7793  023776  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 7794
 7795  024004  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
```

```
 7796  024012                          25$:
 7797
 7798                                  ;       SET AND RESET EBL TO INCREMENT RMDA THEN VERIFY RMDA.
 7799
 7800  024012  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
 7801
 7802  024020  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 7803
 7804  024026  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
 7805  024034  023737  001142  001140          CMP     $BDDAT,$GDDAT
 7806  024042  001402                          BEQ     30$             ;BRANCH IF RMDA OK
 7807  024044  104132                          ERROR   132             ;SECTOR COUNT NOT INCREMENTING
 7808  024046  000416                          BR      50$             ;OUT OF SYNC-SKIP TO NEXT
 7809  024050                          30$:
 7810                                  ;       ADVANCE EXPECTED SECTOR COUNT AND CONTINUE IF ONE CYCLE NOT
 7811                                  ;COMPLETE
 7812  024050  005237  001140                  INC     $GDDAT          ;INCREMENT EXPECTED SECTOR
 7813  024054  123737  001142  024326          CMPB    $BDDAT,110$     ;WAS THE LAST SECTOR JUST COUNTED??
 7814  024062  001004                          BNE     40$             ;NO!!
 7815  024064  105037  001140                  CLRB    $GDDAT          ;YES-NEXT SECTOR SHOULD BE ZERO
 7816  024070  105237  001141                  INCB    $GDDAT+1        ;INCREMENT TRACK ADDRESS
 7817  024074  105737  001142          40$:    TSTB    $BDDAT          ;HAS A FULL CYCLE BEEN COUNTED??
 7818  024100  001401                          BEQ     50$             ;YES-DO NEXT
 7819  024102  000743                          BR      25$             ;CONTINUE SECTOR TEST
 7820
 7821  024104                          50$:
 7822                                  ;;**************************************************************
 7823                                  ;INCREMENT TRACK COUNT USING DIAGNOSTIC END OF BLOCK.  START AT TRACK 0,
 7824                                  ;LAST SECTOR AND COUNT ONE COMPLETE TRACK CYCLE.
 7825  024104  013737  024326  001410          MOV     110$,RMDA0      ;STARTING TRACK ADDRESS
 7826  024112  012737  000400  001140          MOV     #000400,$GDDAT  ;FIRST VALUE AFTER INCREMENT
 7827
 7828  024120                          60$:
 7829
 7830                                  ;       .CLEAR THE MASSBUS
 7831                                  ;       .SET FORMAT
 7832                                  ;       .LOAD LAST SECTOR ADDRESS AND TEST TRACK ADDRESS
 7833                                  ;       .ENABLE DEBUG CLOCK
 7834                                  ;       .SET GO BIT
 7835
 7836  024120  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 7837  024126  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 7838
 7839  024132  013760  001434  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
 7840
 7841  024140  013760  001410  000006          MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
 7842
 7843  024146  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 7844
 7845  024154  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 7846
 7847  024162  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
 7848                                  ;CLOCK RMDA USING DIAGNOSTIC END OF BLOCK
 7849
 7850  024170  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
 7851
```

```
7852  024176  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
7853                                  ;       VERIFY RMDA ACCORDING TO $GDDAT
7854
7855  024204  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
7856  024212  023737  001140  001142          CMP     $GDDAT,$BDDAT
7857  024220  001402                          BEQ     70$
7858  024222  104133                          ERROR   133             ;TRACK COUNT NOT INCREMENTING
7859  024224  000420                          BR      90$             ;OUT OF SYNC-SKIP TO NEXT
7860  024226                          70$:
7861                                  ;       SETUP FOR NEXT INCREMENT OF RMDA TRACK ADDRESS
7862  024226  105237  001141                  INCB    $GDDAT+1        ;ADVANCE EXPECTED TRACK
7863  024232  123727  001143  000004          CMPB    $BDDAT+1,#4     ;WAS THE LAST TRACK JUST COUNTED??
7864  024240  001002                          BNE     80$             ;NO!!
7865  024242  005037  001140                  CLR     $GDDAT          ;YES-NEXT TRACK, SECTOR SHOULD BE ZERO
7866  024246  013737  001142  001410  90$:    MOV     $BDDAT,RMDA0    ;HAS A FULL CYCLE BEEN COUNTED??
7867  024254  001404                          BEQ     90$             ;YES!!
7868  024256  113737  024326  001410          MOVB    110$,RMDA0      ;INCREMENT FROM LAST SECTOR
7869  024264  000715                          BR      60$
7870  024266                          90$:
7871                                  ;;*****************************************************************
7872                                  ;REPEAT THE TEST IN 16 BIT FORAT
7873  024266  032737  010000  001434          BIT     #FMT16,RMOF0    ;DONE BOTH FORMATS??
7874  024274  001013                          BNE     100$            ;YES!!
7875  024276  012737  010000  001434          MOV     #FMT16,RMOF0    ;SET FORMAT BIT FOR 16
7876  024304  012737  000037  024326          MOV     #037,110$       ;SET LAST SECTOR FOR 16
7877  024312  012737  000001  001140          MOV     #1,$GDDAT       ;SET FIRST COUNT VALUE
7878  024320  000137  023720                  JMP     10$             ;REPEAT TEST
7879
7880  024324  000401                  100$:   BR      120$
7881
7882  024326  000000                  110$:   .WORD                   ;STORAGE FOR LAST SECTOR VALUE
7883
7884  024330                          120$:                           ;END OF TEST
```

```
7885
7886
7887                                    ;;*****************************************************************
7888                                    ;*TEST 40        RMDC COUNT TEST
7889
7890                                    ;;*****************************************************************
7891  024330  000004                    TST40:  SCOPE
7892  024332  012737  000040  001226            MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7893
7894  024340  000240                            NOP
7895  024342  012737  000024  001120            MOV     #20.,$ICNT      ;20 ITERATIONS
7896  024350  112737  000001  001131            MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7897  024356  012737  024372  001122            MOV     #T40,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7898  024364  012737  024372  001124            MOV     #T40,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7899  024372                            T40:
7900  024372  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
7901  024376  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7902  024402  013701  001456                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
7903  024406  012760  000040  000010            MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7904  024414  111160  000010                    MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7905                                    ;CLEAR RMDC, SET FORMAT AND SETUP PROGRAM PARAMETERS
7906  024420  010037  001136                    MOV     R0,$BOADR       ;SETUP REGISTER ADDRESS
7907  024424  062737  000034  001136            ADD     #RMDC,$BOADR
7908  024432  005037  001434                    CLR     RMOF0           ;START WITH 18 BIT FORMAT
7909  024436  012737  002035  001410            MOV     #002035,RMDA0   ;LOAD LAST SECTOR/TRACK VALUE
7910  024444  012737  000001  001140  10$:      MOV     #1,$GOBIT       ;LOAD FIRST INCREMENTAL VALUE
7911
7912  024452  012760  000000  000034            MOV     #0,RMDC(R0)     ;LOAD RMDC
7913
7914  024460  013760  001434  000032            MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
7915  024466  013737  001434  001174            MOV     RMOF0,$TMP0     ;SAVE FOR ERROR MSG
7916  024474                            20$:
7917
7918                                    ;       .CLEAR THE MASSBUS
7919                                    ;       .SET OFFSET
7920                                    ;       .LOAD LAST SECTOR AND TRACK ADDRESS
7921                                    ;       .ENABLE DEBUG CLOCK
7922                                    ;       .SET GO BIT
7923
7924  024474  012760  000040  000010            MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7925  024502  111160  000010                    MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7926
7927  024506  013760  001434  000032            MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
7928
7929  024514  013760  001410  000006            MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
7930
7931  024522  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
7932
7933  024530  012760  040001  000024            MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
7934
7935  024536  012760  000001  000000            MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
7936                                    ;CLOCK THE CYLINDER ADDRESS USING DEBL
7937
7938  024544  012760  060001  000024            MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
7939
7940  024552  012760  040001  000024            MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
```

```
7941
7942  024560  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
7943  024566  023737  001140  001142          CMP     $GDDAT,$BDDAT
7944  024574  001402                          BEQ     30$             ;BRANCH IF RMDC=RMDC+1
7945  024576  104140                          ERROR   140             ;CANT INCREMENT RMDC
7946  024600  000427                          BR      60$             ;OUT OF SYNC-SKIP TO END
7947  024602                          30$:
7948                                  ;ADVANCE EXPECTED RESULT FOR NEXT INCREMENT
7949  024602  005237  001140                  INC     $GDDAT          ;ADVANCE NEXT RESULT
7950  024606  022737  002000  001140          CMP     #1777+1,$GDDAT  ;SHOULD NEXT VALUE BE ZERO??
7951  024614  001002                          BNE     40$             ;NO!!
7952  024616  005037  001140                  CLR     $GDDAT          ;YES-RMDC SHOULD OVERFLOW
7953  024622  005737  001142          40$:    TST     $BDDAT          ;IS ONE CYCLE COMPLETE??
7954  024626  001401                          BEQ     50$             ;YES!!
7955  024630  000721                          BR      20$             ;CONTINUE
7956  024632                          50$:
7957                                  ;REPEAT TEST IN 16 BIT MODE-ELSE DONE
7958  024632  032737  010000  001434          BIT     #FMT16,RMOF0    ;DONE BOTH MODES??
7959  024640  001007                          BNE     60$             ;YES!!
7960  024642  012737  010000  001434          MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
7961  024650  012737  002037  001410          MOV     #002037,RMDA0   ;LOAD LAST SECTOR/TRACK VALUE
7962  024656  000672                          BR      10$             ;REPEAT TEST
7963
7964  024660                          60$:                            ;END OF TEST
7965
7966                                  ;;*****************************************************************
7967                                  ;*TEST 41        LBT TEST
7968
7969                                  ;;*****************************************************************
7970  024660  000004                  T$T41:  SCOPE
7971  024662  012737  000041  001226          MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7972
7973  024670  000240                          NOP
7974  024672  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
7975  024700  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
7976  024706  012737  024722  001122          MOV     #T41,$LPADR  ;LOAD LOOP ON TEST ADDRESS
7977  024714  012737  024722  001124          MOV     #T41,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
7978  024722                          T41:
7979  024722  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
7980  024726  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
7981  024732  013701  001456                  MOV     TSTGUE,R1       ;R1 = POINTER TO DEVICE
7982  024736  010037  001136                  MOV     R0,$BDAOR       ;SETUP REGISTER ADDRESS
7983  024742  062737  000012  001136          ADD     #RMOS,$BDAOR
7984                                  ;       CLEAR THE MASSBUS
7985                                  ;       SET FORMAT
7986                                  ;       LOAD LAST TRACK AND SECTOR
7987                                  ;       LOAD LAST CYLINDER
7988
7989                                  ;       VERIFY THAT "LBT" IS RESET
7990  024750  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
7991  024756  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
7992
7993  024762  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
7994
7995  024770  012760  002035  000006          MOV     #002035,RMDA(R0)        ;LOAD RMDA
7996
```

```
 7997  024776  012760  001466  000034          MOV     #822.,RMDC(R0)  ;LOAD RMDC
 7998
 7999  025004  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 8000  025012  042737  175777  001142          BIC     #↑CLBT,$BDDAT
 8001  025020  001403                          BEQ     10$             ;BRANCH IF LBT IS RESET
 8002  025022  005037  001140                  CLR     $GDDAT          ;LBT SHOULD BE ZERO
 8003  025026  104141                          ERROR   141             ;CANNOT RESET "LBT"
 8004
 8005  025030                          10$:
 8006                                  ;       ENABLE DEBUG CLOCK
 8007                                  ;       SET GO
 8008                                  ;       FORCE EBL
 8009                                  ;
 8010                                  ;       VERIFY THAT LBT IS SET
 8011
 8012  025030  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 8013
 8014  025036  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 8015
 8016  025044  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 8017
 8018  025052  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
 8019
 8020  025060  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
 8021
 8022  025066  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 8023
 8024  025074  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 8025  025102  042737  175777  001142          BIC     #↑CLBT,$BDDAT
 8026  025110  001004                          BNE     20$             ;BRANCH IF LBT IS SET
 8027  025112  012737  002000  001140          MOV     #LBT,$GDDAT
 8028  025120  104142                          ERROR   142             ;CANT SET LBT
 8029
 8030  025122                          20$:                            ;END OF TEST
 8031
 8032                                  ;;*****************************************************************
 8033                                  ;*TEST 42        COMPOSITE ERROR TEST
 8034
 8035                                  ;;*****************************************************************
 8036  025122  000004                  T$T42:  SCOPE
 8037  025124  012737  000042  001226          MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8038
 8039  025132  000240                          NOP
 8040  025134  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
 8041  025142  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 8042  025150  012737  025164  001122          MOV     #T42,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 8043  025156  012737  025164  001124          MOV     #T42,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 8044  025164                          T42:
 8045  025164  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 8046  025170  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 8047  025174  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 8048  025200  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 8049  025206  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 8050  025212  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
 8051  025216  062737  000012  001136          ADD     #RMDS,$BDADR
 8052                                  ;USING DIAGNOSTIC MODE, CLEAR ALL ERRORS AND VERIFY THAT COMPOSITE
```

```
8053                                    ;ERROR IS RESET.
8054
8055  025224  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8056
8057  025232  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
8058
8059  025240  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
8060
8061  025246  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8062  025254  042737  137777  001142            BIC     #↑CERR,$BDDAT
8063  025262  001403                            BEQ     10$             ;BRANCH IF ERR IS RESET
8064  025264  005037  001140                    CLR     $GDDAT
8065  025270  104143                            ERROR   143             ;CANT READ ZERO FROM ERR
8066  025272  012737  040000  001140  10$:      MOV     #ERR,$GDDAT
8067                                    ;SET BOTH ERROR REGISTERS AND VERIFY THAT COMPOSITE ERROR IS SET
8068
8069  025300  012760  177777  000014            MOV     #-1,RMER1(R0)   ,LOAD RMER1
8070
8071  025306  012760  177777  000042            MOV     #-1,RMER2(R0)   ;LOAD RMER2
8072
8073  025314  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8074  025322  042737  137777  001142            BIC     #↑CERR,$BDDAT
8075  025330  001001                            BNE     20$             ;BRANCH IF ERR IS SET
8076  025332  104144                            ERROR   144             ;CANT SET ERR
8077  025334                                20$:
8078                                    ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER1
8079  025334  012702  000001                    MOV     #1,R2           ;INITIALIZE TEST PATTERN
8080  025340                                30$:
8081                                    ;       WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
8082  025340  012760  000040  000010            MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
8083  025346  111160  000010                    MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
8084
8085  025352  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8086
8087  025360  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
8088
8089  025366  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
8090
8091  025374  010260  000014                    MOV     R2,RMER1(R0)    ;LOAD RMER1
8092
8093  025400  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8094  025406  042737  137777  001142            BIC     #↑CERR,$BDDAT
8095  025414  001005                            BNE     40$             ;BRANCH IF COMPOSITE ERROR SET
8096  025416  010237  001174                    MOV     R2,$TMP0        ;SAVE RMER1 TEST PATTERN
8097  025422  005037  001176                    CLR     $TMP1           ;SAVE RMER2 TEST PATTERN
8098  025426  104145                            ERROR   145             ;ERR NOT SET BY RMER1 ERROR
8099  025430                                40$:
8100                                    ;       ADVANCE THE TEST PATTERN FOR RMER1
8101  025430  006302                            ASL     R2
8102  025432  001342                            BNE     30$             ;CONTINUE IF TEST NOT DONE
8103  025434                                50$:
8104                                    ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER2
8105  025434  012702  000001                    MOV     #1,R2           ;INITIALIZE TEST PATTERN
8106  025440                                60$:
8107                                    ;       WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
8108  025440  012760  000040  000010            MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
```

F14


```
8109  025446  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
8110
8111  025452  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8112
8113  025460  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
8114
8115  025466  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
8116
8117  025474  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
8118
8119  025500  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8120  025506  042737  137777  001142          BIC     #^CERR,$BDDAT
8121  025514  012737  040000  001140          MOV     #ERR,$GDDAT     ;SETUP EXPECTED VALUE FOR COMP ERROR
8122  025522  032702  001567                  BIT     #XNUER2,R2
8123  025526  001402                          BEQ     65$             ;BRANCH IF TEST BIT IS A USED BIT
8124  025530  005037  001140                  CLR     $GDDAT          ;TEST BIT IS NOT USED - ERR SHOULD BE 0
8125  025534  023737  001140  001142  65$:    CMP     $GDDAT,$BDDAT
8126  025542  001405                          BEQ     70$             ;BRANCH IF COMP ERROR IS OK
8127  025544  005037  001174                  CLR     $TMP0           ;SAVE RMER1 TEST PATTERN
8128  025550  010237  001176                  MOV     R2,$TMP1        ;SAVE RMER2 TEST PATTERN
8129  025554  104145                          ERROR   145             ;ERR NOT SET BY RMER2 ERROR
8130  025556                          70$:
8131                                  ;       ADVANCE THE TEST PATTERN FOR RMER2
8132  025556  006302                          ASL     R2
8133  025560  001327                          BNE     60$             ;CONTINUE IF TEST NOT DONE
8134
8135  025562                          80$:                            ;END OF TEST
8136
8137                                  ;;**********************************************************************
8138                                  ;*TEST 43        WRITE GO TEST
8139
8140                                  ;;**********************************************************************
8141  025562  000004                  T$T43:  SCOPE
8142  025564  012737  000043  001226          MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8143
8144  025572  000240                          NOP
8145  025574  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
8146  025602  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
8147  025610  012737  025624  001122          MOV     #T43,$LPADR  ;LOAD LOOP ON TEST ADDRESS
8148  025616  012737  025624  001124          MOV     #T43,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
8149  025624                          T43:
8150  025624  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
8151  025630  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
8152  025634  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
8153  025640  010037  001136                  MOV     R0,$BDADR       ;COPY RMCS1 ADDRESS
8154  025644  005002                          CLR     R2              ;INITIALIZE FUNCTION CODE
8155  025646                          10$:
8156
8157                                  ;CLEAR THE MASSBUS, SET DIAGNOSTIC MODE AND ENABLE DEBUG CLOCK
8158  025646  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
8159  025654  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
8160
8161  025660  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8162
8163  025666  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8164
```

```
 8165  025674  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 8166
 8167  025702  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 8168                          ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
 8169  025710  010203                          MOV     R2,R3           ;SETUP FUNCTION CODE
 8170  025712  052703  000001                  BIS     #GO,R3
 8171
 8172  025716  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
 8173
 8174  025722  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 8175  025730  032737  000001  001142          BIT     #GO,$BDDAT
 8176  025736  001007                          BNE     20$             ;BRANCH IF GO IS SET
 8177                          ;REPORT THE ERROR-CANT SET GO WITH THIS FUNCTION CODE
 8178  025740  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
 8179  025746  010337  001140                  MOV     R3,$GDDAT       ;SAVE FUNCTION CODE
 8180  025752  104146                          ERROR   146             ;CANT SET GO
 8181  025754  000405                          BR      30$
 8182  025756                        20$:
 8183                          ;ADVANCE R2 TO THE NEXT FUNCTION CODE
 8184  025756  062702  000002                  ADD     #2,R2
 8185  025762  022702  000076                  CMP     #ILF76,R2
 8186  025766  103327                          BHIS    10$
 8187
 8188  025770                        30$:                            ;END OF TEST
 8189
 8190                          ;;*****************************************************************
 8191                          ;*TEST 44        BRANCH MULTIPLEXOR TEST
 8192
 8193                          ;;*****************************************************************
 8194  025770  000004          T$T44:  SCOPE
 8195  025772  012737  000044  001226          MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 8196
 8197  026000  000240                          NOP
 8198  026002  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
 8199  026010  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 8200  026016  012737  026032  001122          MOV     #T44,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 8201  026024  012737  026032  001124          MOV     #T44,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 8202  026032                        T44:
 8203  026032  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 8204  026036  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 8205  026042  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 8206  026046  010037  001136                  MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
 8207  026052  062737  000040  001136          ADD     #RMMR2,$BDADR
 8208  026060  012702  026312                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
 8209                          ;CLEAR THE MASSBUS AND SET DEBUG CLOCK ENABLE
 8210  026064                        10$:
 8211  026064  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 8212  026072  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 8213
 8214  026076  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 8215
 8216  026104  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
 8217
 8218  026112  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 8219
 8220  026120  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
```

```
8221                                    ;THE TEST BIT SHOULD BE ONE BECAUSE THE ADDRESS IS ALL ONES WHEN
8222                                    ;THE COMMAND SEQUENCER IS INITIALIZED.
8223
8224  026126  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
8225  026134  032737  010000  001142          BIT     #TST,$BDDAT
8226  026142  001010                          BNE     15$                     ;BRANCH IF TEST BIT IS ON
8227  026144  042737  167777  001142          BIC     #^CTST,$BDDAT           ;SETUP FOR ERROR TYPE
8228  026152  012737  010000  001140          MOV     #TST,$GDDAT
8229  026160  104147                          ERROR   147                     ;TEST BIT NOT SET
8230  026162  000452                          BR      40$                     ;SKIP REST OF TEST
8231  026164                          15$:
8232                                    ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO THE DEVICE,
8233                                    ;THEN STEP THE COMMAND SEQUENCER ACCORDING TO THE TABLE.
8234
8235  026164  111203                          MOVB    (R2),R3
8236  026166  052703  000001                 BIS     #GO,R3
8237  026172  042703  177700                 BIC     #^CFNCMSK,R3            ;R3=FUNCTION CODE, GO BIT
8238
8239  026176  010360  000000                 MOV     R3,RMCS1(R0)            ;LOAD RMCS1
8240  026202  010337  001174                 MOV     R3,$TMP0                ;SAVE R3 FOR ERROR MSG
8241
8242  026206  116203  000001                 MOVB    1(R2),R3                ;GET CLOCK COUNT IN R3
8243  026212  042703  177400                 BIC     #^C377,R3
8244  026216                          20$:
8245
8246  026216  012760  141001  000024          MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
8247
8248  026224  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8249  026232  005303                          DEC     R3                      ;DECREMENT CLOCK COUNT
8250  026234  001370                          BNE     20$                     ;ISSUE CLOCKS TILL ZERO
8251
8252                                    ;GET THE TEST BIT AND COMPARE IT WITH THE TABLE ENTRY
8253
8254  026236  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
8255  026244  042737  167777  001142          BIC     #^CTST,$BDDAT
8256  026252  016237  000002  001140          MOV     2(R2),$GDDAT
8257  026260  023737  001140  001142          CMP     $GDDAT,$BDDAT
8258  026266  001402                          BEQ     30$                     ;BRANCH IF TEST BIT OK
8259  026270  104150                          ERROR   150                     ;TEST BIT IS INCORRECT
8260  026272  000406                          BR      40$                     ;SKIP REST OF TEST
8261  026274                          30$:
8262                                    ;MOVE THE TABLE POINTER AND CONTINUE IF NEXT ENTRY POSITIVE
8263  026274  062702  000004                 ADD     #4,R2
8264  026300  105762  000001                 TSTB    1(R2)
8265  026304  100401                          BMI     40$                     ;BRANCH IF DONE TEST
8266  026306  000666                          BR      10$                     ;REPEAT TEST
8267  026310  000436                  40$:    BR      200$                    ;JUMP OVER TABLE
8268
8269  026312                          100$:
8270
8271                                    ;TABLE OF FUNCTION CODES, CLOCK COUNTS, AND TEST BITS
8272
8273  026312     000                          .BYTE   NOP                     ;MUX ADDRESS=DATA COMMAND
8274  026313     001                          .BYTE   1
8275  026314  000000                          .WORD   0                       ;TEST BIT=0
8276
```

```
8277  026316  000                   .BYTE   NOP             ;MUX ADDRESS=UNIT READY
8278  026317  002                   .BYTE   2
8279  026320  000000                .WORD   0               ;TEST BIT=0
8280
8281  026322  010                   .BYTE   DRVCLR          ;MUX ADDRESS=F4
8282  026323  001                   .BYTE   1
8283  026324  010000                .WORD   TST             ;TEST BIT=1
8284
8285  026326  050                   .BYTE   WCD             ;MUX ADDRESS=F4
8286  026327  001                   .BYTE   1
8287  026330  000000                .WORD   0               ;TEST BIT=0
8288
8289  026332  012                   .BYTE   RLEASE          ;MUX ADDRESS=F4
8290  026333  001                   .BYTE   1
8291  026334  010000                .WORD   TST             ;TEST BIT=1
8292
8293  026336  052                   .BYTE   WCH             ;MUX ADDRESS=F4
8294  026337  001                   .BYTE   1
8295  026340  000000                .WORD   0               ;TEST BIT=0
8296
8297  026342  020                   .BYTE   RIP             ;MUX ADDRESS=F4
8298  026343  001                   .BYTE   1
8299  026344  010000                .WORD   TST             ;TEST BIT=1
8300
8301  026346  060                   .BYTE   WD              ;MUX ADDRESS=F4
8302  026347  001                   .BYTE   1
8303  026350  000000                .WORD   0               ;TEST BIT=0
8304
8305  026352  022                   .BYTE   PAKACK          ;MUX ADDRESS=F4
8306  026353  001                   .BYTE   1
8307  026354  010000                .WORD   TST             ;TEST BIT=1
8308
8309  026356  062                   .BYTE   WH              ;MUX ADDRESS=F4
8310  026357  001                   .BYTE   1
8311  026360  000000                .WORD   0               ;TEST BIT=0
8312
8313  026362  030                   .BYTE   SEARCH          ;MUX ADDRESS=F4
8314  026363  001                   .BYTE   1
8315  026364  010000                .WORD   TST             ;TEST BIT=1
8316
8317  026366  070                   .BYTE   RD              ;MUX ADDRESS=F4
8318  026367  001                   .BYTE   1
8319  026370  000000                .WORD   0               ;TEST BIT=0
8320
8321  026372  032                   .BYTE   ILF32           ;MUX ADDRESS=F4
8322  026373  001                   .BYTE   1
8323  026374  010000                .WORD   TST             ;TEST BIT=1
8324
8325  026376  072                   .BYTE   RH              ;MUX ADDRESS=F4
8326  026377  001                   .BYTE   1
8327  026400  000000                .WORD   0               ;TEST BIT=0
8328
8329  026402  000                   .BYTE                   ;END OF TABLE
8330  026403  377                   .BYTE   .-1
8331  026404  000000                .WORD
8332  026406              200$:                             ;END OF TEST
```

```
8333
8334                                    ;;*****************************************************************
8335                                    ;*TEST 45        SET/RESET GO TEST
8336
8337                                    ;;*****************************************************************
8338  026406  000004                    T$T45:  SCOPE
8339  026410  012737  000045  001226            MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8340
8341  026416  000240                            NOP
8342  026420  012737  000024  001120            MOV     #20.,$ICNT      ;20 ITERATIONS
8343  026426  112737  000001  001131            MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
8344  026434  012737  026450  001122            MOV     #T45,$LPADR  ;LOAD LOOP ON TEST ADDRESS
8345  026442  012737  026450  001124            MOV     #T45,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
8346  026450                            T45:
8347  026450  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
8348  026454  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
8349  026460  013701  001456                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
8350  026464  012702  027106                    MOV     #200$,R2        ;INITIALIZE FUNCTION CODE POINTER
8351                                    ;CLEAR, THEN SET DIAGNOSTIC MODE, CLEAR COMPOSITE ERROR, SET MEDIUM
8352                                    ;ON LINE AND ENABLE DEBUG CLOCK
8353  026470                            10$:
8354  026470  012760  000040  000010            MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
8355  026476  111160  000010                    MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
8356
8357  026502  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8358
8359  026510  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8360
8361  026516  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
8362
8363  026524  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
8364                                    ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1 AND VERIFY GO IS SET
8365  026532  111203                            MOVB    (R2),R3         ;GET FUNCTION CODE
8366  026534  042703  177701                    BIC     #^C1F76,R3      ;CLEAR UNUSED BITS
8367  026540  052703  000001                    BIS     #GO,R3          ;SET GO
8368
8369  026544  010360  000000                    MOV     R3,RMCS1(R0)    ;LOAD RMCS1
8370
8371  026550  016037  000000  001142            MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
8372  026556  032737  000001  001142            BIT     #GO,$BDDAT
8373  026564  001011                            BNE     20$             ;BRANCH IF GO IS SET
8374  026566  042737  177700  001142            BIC     #^CFNCMSK,$BDDAT
8375  026574  010337  001140                    MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
8376  026600  010037  001136                    MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
8377  026604  104151                            ERROR   151             ;CANT SET GO
8378  026606  000536                            BR      100$
8379
8380  026610                            20$:
8381                                    ;GET READY STATUS AND VERIFY THAT IT IS THE COMPLEMENT OF GO
8382  026610  005037  001140                    CLR     $GDDAT          ;EXPECT DRY TO BE OFF
8383  026614  032737  000001  001142            BIT     #GO,$BDDAT      ;WAS GO SET??
8384  026622  001003                            BNE     30$             ;YES!!
8385  026624  012737  000200  001140            MOV     #DRY,$GDDAT     ;GO WAS NOT SET, DRY SHOULD BE
8386  026632                            30$:
8387
8388  026632  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
```

```
 8389  026640  042737  177577  001142          BIC     #↑CDRY,$BDDAT
 8390  026646  023737  001140  001142          CMP     $GDDAT,$BDDAT
 8391  026654  001406                          BEQ     40$             ;BRANCH IF DRY IS OK
 8392  026656  010037  001136                  MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
 8393  026662  062737  000012  001136          ADD     #RMDS,$BDADR
 8394  026670  104152                          ERROR   152             ;DRY NOT COMPLEMENT OF GO
 8395  026672                          40$:
 8396                                  ;STEP THE DEBUG CLOCK AND VERIFY THAT GO REMAINS SET
 8397  026672  116204  000001                  MOVB    1(R2),R4        ;GET NUMBER OF CLOCK CYCLES
 8398  026676  042704  177400                  BIC     #↑C377,R4
 8399  026702                          50$:
 8400
 8401  026702  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 8402
 8403  026710  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 8404
 8405  026716  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 8406  026724  042737  177700  001142          BIC     #↑CFNCMSK,$BDDAT  ;CLEAR UNUSED BITS
 8407  026732  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
 8408  026736  010337  001140                  MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
 8409                                  ;DECREMENT CLOCK COUNT AND EXIT LOOP IF ZERO
 8410  026742  005304                          DEC     R4
 8411  026744  001406                          BEQ     60$
 8412  026746  032737  000001  001142          BIT     #GO,$BDDAT      ;IS GO STILL SET??
 8413  026754  001352                          BNE     50$             ;YES!!
 8414  026756  104153                          ERROR   153             ;GO RESET EARLY
 8415  026760  000451                          BR      100$            ;OUT OF SYNC=SKIP TO NEXT
 8416  026762                          60$:
 8417                                  ;GO SHOULD NOW BE RESET AND DRY SHOULD BE SET
 8418  026762  032737  000001  001142          BIT     #GO,$BDDAT      ;IS GO RESET??
 8419  026770  001405                          BEQ     70$             ;YES!!
 8420  026772  042737  000001  001140          BIC     #GO,$GDDAT      ;SETUP EXPECTED RESULT
 8421  027000  104154                          ERROR   154             ;GO DID NOT RESET
 8422  027002  000440                          BR      100$
 8423  027004                          70$:
 8424  027004  012737  000200  001140          MOV     #DRY,$GDDAT     ;EXPECT DRIVE READY TO BE SET
 8425  027012  032737  000001  001142          BIT     #GO,$BDDAT      ;DID GO RESET??
 8426  027020  001402                          BEQ     80$             ;YES!!
 8427  027022  005037  001140                  CLR     $GDDAT          ;GO IS SET-
 8428  027026                          80$:
 8429
 8430  027026  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 8431  027034  042737  177577  001142          BIC     #↑CDRY,$BDDAT
 8432  027042  010037  001136                  MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
 8433  027046  062737  000012  001136          ADD     #RMDS,$BDADR
 8434  027054  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS DRIVE READY OK??
 8435  027062  001401                          BEQ     90$             ;YES!!
 8436  027064  104152                          ERROR   152             ;DRY NOT COMPLEMENT OF GO
 8437  027066                          90$:
 8438                                  ;ADVANCE TO THE NEXT FUNCTION CODE TO BE TESTED-EXIT IF DONE
 8439  027066  062702  000002                  ADD     #2,R2           ;MOVE TABLE POINTER
 8440  027072  105762  000001                  TSTB    1(R2)           ;END OF TABLE??
 8441  027076  100402                          BMI     100$            ;YES!!
 8442  027100  000137  026470                  JMP     10$             ;TEST THIS FUNCTION CODE
 8443  027104  000423                  100$:   BR      300$            ;GO TO NEXT TEST
 8444
```

```
8445  027106                          200$:
8446                                  ;;*****************************************************************
8447                                  ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
8448  027106  002                             .BYTE   ILF02           ;ILLEGAL FUNCTION CODE #2
8449  027107  001                             .BYTE   1
8450
8451  027110  004                             .BYTE   SEEK            ;SEEK COMMAND
8452  027111  001                             .BYTE   1
8453
8454  027112  006                             .BYTE   RECAL           ;RECALIBRATE COMMAND
8455  027113  001                             .BYTE   1
8456
8457  027114  014                             .BYTE   OFFSET          ;OFFSET COMMAND
8458  027115  001                             .BYTE   1
8459
8460  027116  016                             .BYTE   RTC             ;RETURN TO CENTER LINE COMMAND
8461  027117  001                             .BYTE   1
8462
8463  027120  024                             .BYTE   ILF24           ;ILLEGAL FUNCTION CODE #24
8464  027121  001                             .BYTE   1
8465
8466  027122  026                             .BYTE   ILF26           ;ILLEGAL FUNCTION CODE #26
8467  027123  001                             .BYTE   1
8468
8469  027124  034                             .BYTE   ILF34           ;ILLEGAL FUNCTION CODE #34
8470  027125  001                             .BYTE   1
8471
8472  027126  036                             .BYTE   ILF36           ;ILLEGAL FUNCTION CODE #36
8473  027127  001                             .BYTE   1
8474
8475  027130  042                             .BYTE   ILF42           ;ILLEGAL FUNCTION CODE #42
8476  027131  001                             .BYTE   1
8477
8478  027132  044                             .BYTE   ILF44           ;ILLEGAL FUNCTION CODE #44
8479  027133  001                             .BYTE   1
8480
8481  027134  046                             .BYTE   ILF46           ;ILLEGAL FUNCTION CODE #46
8482  027135  001                             .BYTE   1
8483
8484  027136  054                             .BYTE   ILF54           ;ILLEGAL FUNCTION CODE #54
8485  027137  001                             .BYTE   1
8486
8487  027140  056                             .BYTE   ILF56           ;ILLEGAL FUNCTION CODE #56
8488  027141  001                             .BYTE   1
8489
8490  027142  064                             .BYTE   ILF64           ;ILLEGAL FUNCTION CODE #64
8491  027143  001                             .BYTE   1
8492
8493  027144  066                             .BYTE   ILF66           ;ILLEGAL FUNCTION CODE #66
8494  027145  001                             .BYTE   1
8495
8496  027146  074                             .BYTE   ILF74           ;ILLEGAL FUNCTION CODE #74
8497  027147  001                             .BYTE   1
8498
8499  027150  076                             .BYTE   ILF76           ;ILLEGAL FUNCTION CODE #76
8500  027151  001                             .BYTE   1
```

```
                                        .BYTE                           ;END OF TABLE
027153  377                           .BYTE   #-1

027154                          300$:                                   ;END OF TEST

                                ;;*****************************************************************
                                ;*TEST 46        END 1 RESET GO TEST
                                ;;*****************************************************************
027154  000004                  TST46:  SCOPE
027156  012737  000046  001226          MOV     #46,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

027164  000240                          NOP
027166  012737  000024  001120          MOV     #20.,$ICNT              ;20 ITERATIONS
027174  112737  000001  001131          MOVB    #1,$ERMAX               ;ONE ERROR ALLOWED
027202  012737  027216  001122          MOV     #T46,$LPADR     ;LOAD LOOP ON TEST ADDRESS
027210  012737  027216  001124          MOV     #T46,$LPERR     ;LOAD LOOP ON ERROR ADDRESS
027216                          T46:
027216  012706  001100                  MOV     #STACK,SP               ;LOAD THE STACK POINTER
027222  013700  001276                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS OF UUT
027226  013701  001456                  MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
027232  012702  027476                  MOV     #100$,R2                ;INITIALIZE TABLE POINTER
027236  010037  001136                  MOV     R0,$BOADR               ;COPY RMCS1 ADDRESS

027242                          10$:
                                ;CLEAR MASSBUS, THEN SET MEDIUM ON LINE AND ENABLE DEBUG CLOCK
027242  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
027250  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

027254  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1

027262  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1

027270  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1

027276  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
                                ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
027304  111203                  MOVB    (R2),R3                 ;GET FUNCTION CODE FROM
027306  042703  177701          BIC     #^C<CIL!F76>,R3         ;TABLE AND SET GO
027312  052703  000001          BIS     #GO,R3
027316  010337  001140          MOV     R3,$GDDAT               ;SAVE FUNCTION CODE FOR MSG

027322  010360  000000          MOV     R3,RMCS1(R0)    ;LOAD RMCS1

027326  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
027334  032737  000001  001142          BIT     #GO,$BDDAT
027342  001005                  BNE     20$                     ;BRANCH IF GO IS SET
027344  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
027352  104151                  ERROR   151                     ;GO DID NOT SET
027354  000447                  BR      60$                     ;OUT OF SYNC-SKIP
027356                          20$:
                                ;GET THE NUMBER OF CLOCK CYCLES FROM THE TABLE, SAVE EXPECTED STATUS
027356  116204  000001          MOVB    1(R2),R4                ;R4=CLOCK COUNT
027362  042704  177400          BIC     #^C377,R4
027366                          30$:
```

```
                                        ;STEP THE DEBUG CLOCK AND VERIFY GO STATUS ON UNTIL CLOCK COUNT EXPIRES.
                                        
027366  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
                                        
027374  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                        
027402  016037  000000  001142            MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
027410  042737  177700  001142            BIC     #^CFNCMSK,$BDDAT
027416  005304                            DEC     R4
027420  001406                            BEQ     40$             ;BRANCH IF GO SHOULD BE OFF
027422  032737  000001  001142            BIT     #GO,$BDDAT
027430  001356                            BNE     30$             ;CONTINUE IF GO IS ON
027432  104153                            ERROR   153             ;GO RESET EARLY
027434  000417                            BR      60$             ;OUT OF SYNC-SKIP
027436                                  40$:
                                        ;VERIFY THAT GO RESET AT END1
027436  032737  000001  001142            BIT     #GO,$BDDAT      ;DID GO RESET??
027444  001405                            BEQ     50$             ;YES!!
027446  042737  000001  001140            BIC     #GO,$GDDAT
027454  104154                            ERROR   154             ;GO NOT RESET AT END1
027456  000406                            BR      60$
027460                                  50$:
                                        ;GET THE NEXT FUNCTION CODE FROM THE TABLE
027460  062702  000002                    ADD     #2,R2
027464  105762  000001                    TSTB    1(R2)
027470  100401                            BMI     60$             ;BRANCH IF END OF TABLE
027472  000663                            BR      10$             ;TEST THIS FUNCTION CODE
027474  000404                          60$:    BR      200$            ;JUMP OVER TABLE
                                        
027476                                  100$:
                                        ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
                                        
027476      012                             .BYTE   RLEASE          ;RELEASE COMMAND
027477      002                             .BYTE   2
                                        
027500      030                             .BYTE   SEARCH          ;SEARCH COMMAND
027501      002                             .BYTE   2
                                        
027502      032                             .BYTE   ILF32           ;ILLEGAL FUNCTON #32
027503      002                             .BYTE   2
                                        
027504      000                             .BYTE                   ;END OF TABLE
027505      377                             .BYTE   #-1
027506                                  200$:                       ;END OF TEST
                                        
                                        ;;*****************************************************************
                                        ;*TEST 47       SET PULSE TEST
                                        
                                        ;;*****************************************************************
027506  000004                        ;ST47:  SCOPE
027510  012737  000047  001226            MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                        
027516  000240                            NOP
027520  012737  000024  001120            MOV     #20.,$ICNT      ;20 ITERATIONS
027526  112737  000001  001131            MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
027534  012737  027550  001122            MOV     #T47,$LPADR  ;LOAD LOOP ON TEST ADDRESS
```

```
8613  027542  012737  027550  001124          MOV     #T47,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
8614  027550                              T47:
8615  027550  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
8616  027554  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
8617  027560  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
8618  027564  010037  001136                  MOV     R0,$BDADR       ;COPY REG ADDRESS FOR MSG
8619  027570  062737  000024  001136          ADD     #RMMR1,$BDADR
8620  027576  012702  030042                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
8621  027602                              10$:
8622                                      ;CLEAR THE MASS BUS, ENABLE DEBUG CLOCK, AND RESET ERROR REGISTERS
8623  027602  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
8624  027610  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
8625
8626  027614  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
8627
8628  027622  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8629
8630  027630  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
8631
8632  027636  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
8633                                      ;VERIFY THAT CONTINUE, "CONT" IS RESET AFTER CLEAR
8634
8635  027644  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
8636  027652  042737  177677  001142          BIC     #^CCONT,$BDDAT
8637  027660  001404                          BEQ     20$             ;BRANCH IF CONT WAS CLEARED
8638  027662  005037  001140                  CLR     $GDDAT          ;FOR ERROR MSG
8639  027666  104155                          ERROR   155             ;CANT CLEAR CONTINUE
8640  027670  000463                          BR      70$
8641  027672                              20$:
8642                                      ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO RMCS1
8643  027672  111203                          MOVB    (R2),R3
8644  027674  052703  000001                  BIS     #GO,R3
8645  027700  042703  177700                  BIC     #^CFNCMSK,R3    ;R3=FUNCTION CODE AND GO
8646
8647  027704  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
8648  027710  010337  001174                  MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
8649                                      ;GET THE CLOCK COUNT FROM THE TABLE
8650  027714  116203  000001                  MOVB    1(R2),R3
8651  027720  042703  177400                  BIC     #^C377,R3
8652                                      ;GET THE BIT STREAM FOR CONTINUE FROM THE TABLE
8653  027724  016204  000002                  MOV     2(R2),R4
8654  027730                              30$:
8655                                      ;STEP THE COMMAND SEQUENCER AND VERIFY CONTINUE STATUS
8656
8657  027730  012760  141001  000024          MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
8658
8659  027736  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
8660
8661  027744  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
8662  027752  042737  177677  001142          BIC     #^CCONT,$BDDAT
8663  027760  005037  001140                  CLR     $GDDAT          ;GENERATE EXPECTED CONTINUE
8664  027764  032704  000001                  BIT     #BIT0,R4
8665  027770  001403                          BEQ     40$
8666  027772  012737  000100  001140          MOV     #CONT,$GDDAT
8667  030000  023737  001140  001142  40$:    CMP     $GDDAT,$BDDAT
8668  030006  001402                          BEQ     50$             ;BRANCH IF CONTINUE IS OK
```

```
8669  030010  104156                          ERROR   156             ;CONTINUE IS INCORRECT
8670  030012  000412                          BR      70$             ;SKIP
8671  030014                          50$:
8672                                  ;DECREMENT CLOCK COUNT AND SHIFT BIT STREAM
8673  030014  005303                          DEC     R3
8674  030016  001402                          BEQ     60$             ;BRANCH IF CLOCK COUNT EXPIRED
8675  030020  006204                          ASR     R4              ;SHIFT TO NEXT CONTINUE BIT
8676  030022  000742                          BR      30$             ;TEST NEXT CLOCK CYCLE
8677  030024                          60$:
8678                                  ;ADVANCE TABLE POINTER-EXIT IF DONE
8679  030024  062702  000004                  ADD     #4,R2
8680  030030  105762  000001                  TSTB    1(R2)
8681  030034  100401                          BMI     70$             ;EXIT IF CLOCK COUNT NEGATIVE
8682  030036  000661                          BR      10$             ;CONTINUE TEST
8683  030040  000442                  70$:    BR      200$            ;JUMP OVER TABLE
8684
8685  030042                          100$:
8686                                  ;TABLE OF FUNCTION CODES, CLOCK COUNTS AND CONTINUE BITS FOR TEST
8687
8688  030042      000                         .BYTE   NOP             ;NOP COMMAND
8689  030043      004                         .BYTE   4               ;4 CLOCKS
8690  030044  000000                          .WORD   ^B0000          ;CONTINUE=0000
8691
8692  030046      002                         .BYTE   ILF02           ;ILLEGAL FUNCTION 2
8693  030047      002                         .BYTE   2
8694  030050  000000                          .WORD   ^B00
8695
8696  030052      004                         .BYTE   SEEK            ;SEEK COMMAND
8697  030053      002                         .BYTE   2
8698  030054  000000                          .WORD   ^B00
8699
8700  030056      006                         .BYTE   RECAL           ;RECALIBRATE COMMAND
8701  030057      002                         .BYTE   2
9702  030060  000000                          .WORD   ^B00
8703
8704  030062      010                         .BYTE   DRVCLR          ;DRIVE CLEAR COMMAND
8705  030063      002                         .BYTE   2
8706  030064  000001                          .WORD   ^B01
8707
8708  030066      012                         .BYTE   RLEASE          ;RELEASE COMMAND
8709  030067      003                         .BYTE   3
8710  030070  000000                          .WORD   ^B000
8711
8712  030072      014                         .BYTE   OFFSET          ;OFFSET COMMAND
8713  030073      002                         .BYTE   2
8714  030074  000000                          .WORD   ^B00
8715
8716  030076      016                         .BYTE   RTC             ;RETURN TO CENTER COMMAND
8717  030077      002                         .BYTE   2
8718  030100  000000                          .WORD   ^B00
8719
8720  030102      020                         .BYTE   RIP             ;READ IN PRESET COMMAND
8721  030103      004                         .BYTE   4
8722  030104  000016                          .WORD   ^B1110
8723
8724  030106      022                         .BYTE   PAKACK          ;PACK ACKNOWLEDGE
```

```
8725  030107  004                     .BYTE   4
8726  030110  000016                  .WORD   ↑B1110
8727
8728  030112  024                     .BYTE   ILF24           ;ILLEGAL FUNCTION 24
8729  030113  002                     .BYTE   2
8730  030114  000000                  .WORD   ↑B00
8731
8732  030116  026                     .BYTE   ILF26           ;ILLEGAL FUNCTION 26
8733  030117  002                     .BYTE   2
8734  030120  000000                  .WORD   ↑B00
8735
8736  030122  030                     .BYTE   SEARCH          ;SEARCH COMMAND
8737  030123  003                     .BYTE   3
8738  030124  000000                  .WORD   ↑B000
8739
8740  030126  032                     .BYTE   ILF32           ;ILLEGAL FUNCTION 32
8741  030127  003                     .BYTE   3
8742  030130  000000                  .WORD   ↑B000
8743
8744  030132  034                     .BYTE   ILF34           ;ILLEGAL FUNCTION 34
8745  030133  002                     .BYTE   2
8746  030134  000000                  .WORD   ↑B00
8747
8748  030136  036                     .BYTE   ILF36           ;ILLEGAL FUNCTION 36
8749  030137  002                     .BYTE   2
8750  030140  000000                  .WORD   ↑B00
8751
8752  030142  000                     .BYTE                   ;END OF TABLE
8753  030143  377                     .BYTE   -1
8754  030144  000000                  .WORD
8755
8756  030146                  200$:                           ;END OF TEST
8757
8758                          ;;*****************************************************************
8759                          ;*TEST 50        SET/RESET IVC TEST
8760
8761                          ;;*****************************************************************
8762  030146  000004          TST50:  SCOPE
8763  030150  012737  000050  001226          MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8764
8765  030156  000240                  NOP
8766  030160  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
8767  030166  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
8768  030174  012737  030210  001122          MOV     #T50,$LPADR  ;LOAD LOOP ON TEST ADDRESS
8769  030202  012737  030210  001124          MOV     #T50,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
8770  030210                  T50:
8771  030210  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
8772  030214  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
8773  030220  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
8774  030224  010037  001136                  MOV     R0,$BOADR       ;SETUP REG ADDRESS
8775  030230  062737  000042  001136          ADD     #RMER2,$BOADR
8776  030236  005002                  CLR     R2              ;R2=FUNCTION CODE
8777  030240                  10$:
8778                          ;INITIALIZE AND VERIFY THAT IVC STATUS IS ZERO.
8779  030240  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
8780  030246  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
```

```
                                                        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
                                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                                        MOV     #0,RMER1(R0)    ;LOAD RMER1
                                                        MOV     #0,RMER2(R0)    ;LOAD RMER2
                                                        MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
                                                        BIC     #^tCIVC,$BDDAT
                                                        BEQ     20$             ;BRANCH IF IVC IS ZERO
                                                        CLR     $GDDAT
                                                        ERROR   157             ;CANT CLEAR IVC
                                                        BR      40$             ;SKIP REST OF TEST
                                                20$:
                                                ;LOAD THE FUNCTION CODE WITH GO BIT, STEP THE COMMAND SEQUENCER OFF
                                                ;ADDRESS 0 AND VERIFY IVC STATUS.
                                                        MOV     R2,R3           ;SETUP FUNCTION CODE
                                                        BIS     #GO,R3
                                                        MOV     R3,RMCS1(R0)    ;LOAD RMCS1
                                                        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
                                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                                        MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
                                                        BIC     #^tCIVC,$BDDAT  ;SET ACTUAL STATUS
                                                        MOV     FNCDTB(R2),$GDDAT  ;SETUP EXPECTED STATUS FROM
                                                        BIC     #^tCIVC,$GDDAT  ;FUNCTION CODE TABLE
                                                        CMP     $GDDAT,$BDDAT
                                                        BEQ     30$             ;BRANCH IF IVC IS OK
                                                        MOV     R2,$TMP0        ;SAVE FUNCTION CODE FOR MSG
                                                        ERROR   160             ;IVC IS INCORRECT
                                                30$:
                                                ;ADVANCE FUNCTION CODE AND REPEAT TEST IF NOT DONE
                                                        ADD     #2,R2
                                                        CMP     #ILF76,R2
                                                        BLO     40$             ;BRANCH IF DONE TEST
                                                        BR      10$

                                                40$:                            ;END OF TEST

                                                ;;*****************************************************************
                                                ;*TEST 51        SET LSC TEST

                                                ;;*****************************************************************
                                                T$T51:  SCOPE
                                                        MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                                        NOP
                                                        MOV     #20.,$ICNT      ;20 ITERATIONS
                                                        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
                                                        MOV     #T51,$LPADR  ;LOAD LOOP ON TEST ADDRESS
                                                        MOV     #T51,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
```

```
                                TST1:
030502                                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
030502  012706  001100                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
030506  013700  001276                   MOV     TSTOUE,R1       ;R1 = POINTER TO DEVICE
030512  013701  001456                   MOV     R0,$BDADR
030516  010037  001136                   ADD     #RMER2,$BDADR
030522  062737  000042  001136
                                ;INITIALIZE AND VERIFY THAT LOSS OF SYSTEM CLOCK, "LSC", IS RESET
030530  012760  000040  000010           MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
030536  111160  000010                   MOVB    (R1),RMCS2(R0)  ;SELECT UNIT

030542  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1

030550  012760  040001  000024           MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1

030556  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1

030564  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2

030572  016037  000042  001142           MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
030600  042737  173777  001142           BIC     #^tCLSC,$BDDAT
030606  001403                           BEQ     10$             ;BRANCH IF LSC IS ZERO
030610  005037  001140                   CLR     $GDDAT
030614  104161                           ERROR   161             ;CANT CLEAR LSC
                                ;WITH DEBUG CLOCK ENABLED, SET GO AND WAIT FOR ONE SHOT TO SET
030616                          10$:

030616  012760  000001  000000           MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
030624  012737  000001  001524           MOV     #1,WATCH        ;SET WATCHDOG TIMER VALUE
030632  004777  150670                   JSR     PC,@CLOCK       ;START THE CLOCK
030636  005737  001524           20$:    TST     WATCH
030642  001375                           BNE     20$             ;WAIT FOR WATCH ZERO
030644  004777  150660                   JSR     PC,@STOP        ;STOP THE CLOCK
                                ;ONE SHOT SHOULD BE SET-DISABLE DIAGNOSTIC CLOCK AND LSC SHOULD SET.

030650  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1

030656  016037  000042  001142           MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
030664  042737  ; 3777  001142           BIC     #^tCLSC,$BDDAT
030672  001004                           BNE     30$             ;BRANCH IF LSC SET
030674  012737  004000  001140           MOV     #LSC,$GDDAT
030702  104162                           ERROR   162             ;CANT SET LSC

030704                          30$:                            ;END OF TEST

                                ;;*****************************************************************
                                ;*TEST 52        DECODE TEST
                                ;;*****************************************************************
030704  000004                  TST52:  SCOPE
030706  012737  000052  001226           MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

030714  000240                           NOP
030716  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
030724  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
030732  012737  030746  001122           MOV     #T52,$LPADR  ;LOAD LOOP ON TEST ADDRESS
```

```
8893  030740  012737  030746  001124          MOV     #T52,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
8894  030746                          T52:
8895  030746  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
8896  030752  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
8897  030756  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
8898  030762  005037  001416                  CLR     RMER10          ;NO ERROR FIRST TEST
8899  030766                          5$:
8900  030766  004737  031346                  JSR     PC,10$          ;INITIALIZE
8901
8902                                  ;EXECUTE A PACK ACKNOWLEDGE AND CHECK VOLUME VALID
8903
8904  030772  013760  001416  000014          MOV     RMER10,RMER1(R0)        ;LOAD RMER1
8905
8906  031000  012760  000023  000000          MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
8907  031006  012703  000003                  MOV     #3,R3
8908  031012                          10$:
8909
8910  031012  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
8911
8912  031020  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8913  031026  005303                          DEC     R3
8914  031030  001370                          BNE     10$             ;ISSUE NEXT CLOCK IF COUNT NOT 0
8915
8916  031032  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8917  031040  042737  177677  001142          BIC     #^CVV,$BDDAT
8918  031046  001414                          BEQ     20$             ;BRANCH IF VV IS ZERO
8919  031050  005737  001416                  TST     RMER10
8920  031054  001427                          BEQ     70$             ;BRANCH IF VV SHOULD BE SET
8921  031056  005037  001140                  CLR     $GDDAT          ;SETUP ERROR MESSAGE
8922  031062  010037  001136                  MOV     R0,$BDADR
8923  031066  062737  000012  001136          ADD     #RMDS,$BDADR
8924  031074  104163                          ERROR   163             ;DECODE SET WITH COMP ERROR ACTIVE
8925  031076  000522                          BR      80$             ;SKIP
8926  031100                          20$:
8927  031100  004737  031346                  JSR     PC,10$          ;INITIALIZE AND SET DIAGNOSTIC MODE
8928
8929                                  ;EXECUTE A READ IN PRESET AND CHECK VOLUME VALID
8930
8931  031104  013760  001416  000014          MOV     RMER10,RMER1(R0)        ;LOAD RMER1
8932
8933  031112  012760  000021  000000          MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
8934  031120  012703  000003                  MOV     #3,R3           ;R3=CLOCK COUNT
8935  031124                          30$:
8936
8937  031124  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
8938
8939  031132  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
8940  031140  005303                          DEC     R3
8941  031142  001370                          BNE     30$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
8942
8943  031144  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
8944  031152  042737  177677  001142          BIC     #^CVV,$BDDAT
8945  031160  001414                          BEQ     40$             ;BRANCH IF VOLUME VALID NOT SET
8946  031162  005737  001416                  TST     RMER10
8947  031166  001462                          BEQ     70$             ;BRANCH IF VOLUME VALID SHOULD BE SET
8948  031170  005037  001140                  CLR     $GDDAT          ;SETUP ERROR MESSAGE
```

```
 8949  031174  010037  001136                MOV     R0,$BDADR
 8950  031200  062737  000012  001136        ADD     #RMDS,$BDADR
 8951  031206  104163                        ERROR   163             ;DECODE SET WITH COMP ERROR ACTIVE
 8952  031210  000455                        BR      80$             ;SKIP
 8953  031212                          40$:
 8954  031212  004737  031346                JSR     PC,100$         ;INITIALIZE AND SET DIAGNOSTIC MODE
 8955
 8956                                  ;EXECUTE A WRITE CHECK DATA AND CHECK OCCUPIED
 8957
 8958  031216  013760  001416  000014        MOV     RMER10,RMER1(R0)        ;LOAD RMER1
 8959
 8960  031224  012760  000051  000000        MOV     #WCD!GO,RMCS1(R0)       ;LOAD RMCS1
 8961  031232  012703  000002                MOV     #2,R3           ;R3=CLOCK COUNT
 8962  031236                          50$:
 8963
 8964  031236  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 8965
 8966  031244  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 8967  031252  005303                        DEC     R3
 8968  031254  001370                        BNE     50$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
 8969
 8970  031256  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 8971  031264  042737  077777  001142        BIC     #^tCOCC,$BDDAT
 8972  031272  001414                        BEQ     60$             ;BRANCH IF OCCUPIED IS RESET
 8973  031274  005737  001416                TST     RMER10
 8974  031300  001415                        BEQ     70$             ;BRANCH IF OCCUPIED SHOULD BE SET
 8975  031302  005037  001140                CLR     $GDDAT          ;SETUP ERROR MESSAGE
 8976  031306  010037  001136                MOV     R0,$BDADR
 8977  031312  062737  000024  001136        ADD     #RMMR1,$BDADR
 8978  031320  104164                        ERROR   164             ;DECODE SET WITH COMP ERROR ACTIVE
 8979  031322  000410                        BR      80$
 8980  031324                          60$:
 8981                                  ;VOLUME VALID AND OCCUPIED DID NOT SET-SEE IF COMP ERROR WAS ACTIVE
 8982  031324  005737  001416                TST     RMER10
 8983  031330  001005                        BNE     80$             ;BRANCH IF COMP ERROR WAS SET
 8984                                  ;COULD NOT SET VV OR OCCUPIED-SUSPECT DECODE FLOP NOT SETTING
 8985  031332  104165                        ERROR   165             ;DECODE DOES NOT SET
 8986
 8987  031334                          70$:
 8988
 8989                                  ;REPEAT TEST WITH COMPOSITE ERROR ACTIVE-VERIFY THAT DECODE FLOP
 8990                                  ;DOES NOT SET, AS INDICATED BY VOLUME VALID AND OCCUPIED.
 8991  031334  012737  040000  001416        MOV     #UNS,RMER10     ;USE UNSAFE TO SET COMP ERROR
 8992  031342  000611                        BR      5$
 8993
 8994  031344  000513                  80$:  BR      200$            ;END OF TEST
 8995
 8996  031346                          100$:
 8997
 8998                                  ;SUBROUTINE USED DURING TEST
 8999
 9000                                  ;USING DIAGNOSTIC MODE, RESET VOLUME VALID AND COMPOSITE ERROR.
 9001                                  ;VERIFY THAT VV, ERR, AND OCC ARE ZERO.
 9002  031346  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 9003  031354  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 9004
```

```
9005  031360  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
9006
9007  031366  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9008
9009  031374  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9010
9011  031402  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
9012  031410  005037  001140                  CLR     $GODAT          ;SETUP FOR ERROR MSG
9013  031414  010037  001136                  MOV     R0,$BDADR
9014  031420  062737  000012  001136          ADD     #RMDS,$BDADR
9015
9016  031426  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9017  031434  042737  137777  001142          BIC     #↑CERR,$BDDAT
9018  031442  001402                          BEQ     110$            ;BRANCH IF COMP ERROR ZERO
9019  031444  104143                          ERROR   143             ;CANT CLEAR COMP ERROR
9020  031446  000447                          BR      140$            ;SKIP TEST
9021  031450                          110$:
9022
9023  031450  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9024  031456  042737  177677  001142          BIC     #↑CVV,$BDDAT
9025  031464  001402                          BEQ     120$            ;BRANCH IF VOLUME VALID ZERO
9026  031466  104135                          ERROR   135             ;CANT RESET VOLUME VALID
9027  031470  000436                          BR      140$            ;SKIP TEST
9028  031472                          120$:
9029
9030  031472  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
9031  031500  042737  077777  001142          BIC     #↑COCC,$BDDAT
9032  031506  001407                          BEQ     130$            ;BRANCH IF OCCUPIED ZERO
9033  031510  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
9034  031514  062737  000024  001136          ADD     #RMMR1,$BDADR
9035  031522  104166                          ERROR   166             ;CANT CLEAR OCCUPIED
9036  031524  000420                          BR      140$            ;SKIP TEST
9037  031526                          130$:
9038                                  ;TO VERIFY THAT THE DECODE FLOP IS RESET, LOAD AN ILLEGAL FUNCTION
9039                                  ;IN RMCS1 AND VERIFY THAT ILF DOES NOT SET.
9040
9041  031526  012760  000024  000000          MOV     #ILF24,RMCS1(R0)        ;LOAD RMCS1
9042
9043  031534  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
9044  031542  042737  177776  001142          BIC     #↑CILF,$BDDAT
9045  031550  001410                          BEQ     150$            ;BRANCH IF ILF IS ZERO
9046  031552  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
9047  031556  062737  000014  001136          ADD     #RMER1,$BDADR
9048  031564  104167                          ERROR   167             ;DECODE FLOP APPEARS ON
9049  031566  012716  031574          140$:   MOV     #200$,(SP)      ;DONT GO BACK TO TEST
9050
9051  031572  000207                  150$:   RTS     PC              ;RETURN TO TEST OR EXIT TEST
9052
9053  031574                          200$:
9054
9055                                  ;;*****************************************************************
9056                                  ;*TEST 53        SET/RESET VOLUME VALID TEST
9057
9058                                  ;;*****************************************************************
9059  031574  000004                  TST53:  SCOPE
9060  031576  012737  000053  001226          MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
9061
9062  031604  000240                          NOP
9063  031606  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9064  031614  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9065  031622  012737  031636  001122          MOV     #T53,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9066  031630  012737  031636  001124          MOV     #T53,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9067  031636                          T53:
9068  031636  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9069  031642  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9070  031646  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9071  031652  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
9072  031656  062737  000012  001136          ADD     #RMDS,$BDADR
9073  031664  012702  032076                  MOV     #100$,R2        ;R2=TABLE POINTER
9074  031670                          10$:
9075                                  ;INITIALIZE AND USE DIAGNOSTIC MODE TO RESET VOLUME VALID
9076  031670  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
9077  031676  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
9078
9079  031702  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
9080
9081  031710  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9082
9083  031716  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9084
9085  031724  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
9086
9087  031732  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9088  031740  042737  177677  001142          BIC     #^CVV,$BDDAT
9089  031746  001403                          BEQ     20$             ;BRANCH IF VOLUME VALID ZERO
9090  031750  005037  001140                  CLR     $GDDAT
9091  031754  104135                          ERROR   135             ;CANT RESET VV
9092  031756                          20$:
9093                                  ;EXECUTE THE FUNCTION CODE IN THE TABLE
9094  031756  111203                          MOVB    (R2),R3         ;GET FUNCTION CODE
9095  031760  042703  177701                  BIC     #^CILF76,R3
9096  031764  052703  000001                  BIS     #GO,R3
9097
9098  031770  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
9099  031774  116204  000001                  MOVB    1(R2),R4        ;GET CLOCK COUNT
9100  032000  042704  177400                  BIC     #^C377,R4
9101  032004                          30$:
9102
9103  032004  012760  141001  000024          MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
9104
9105  032012  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
9106  032020  005304                          DEC     R4
9107  032022  001370                          BNE     30$             ;ISSUE COCKS TIL R4 ZERO
9108
9109  032024  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9110  032032  042737  177677  001142          BIC     #^CVV,$BDDAT
9111  032040  001007                          BNE     40$             ;BRANCH IF VOLUME VALID SET
9112  032042  010337  001174                  MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
9113  032046  012737  000100  001140          MOV     #VV,$GDDAT
9114  032054  104170                          ERROR   170             ;CANT SET VOLUME VALID
9115  032056  000406                          BR      50$
9116  032060                          40$:
```

```
 9117                                   ;ADVANCE THE TABLE POINTER, EXIT IF DONE
 9118  032060  062702  000002                 ADD     #2,R2
 9119  032064  105762  000001                 TSTB    1(R2)
 9120  032070  100401                         BMI     50$             ;EXIT IF COUNT IS NEGATIVE
 9121  032072  000676                         BR      10$
 9122  032074  000403                   50$:  BR      200$            ;JUMP OVER TABLE
 9123
 9124  032076                         10$:
 9125                                   ;TABLE OF FUNCTION CODES AND CLOCK COUNTS
 9126  032076      020                        .BYTE   RIP             ;READ IN PRESET COMMAND
 9127  032077      003                        .BYTE   3
 9128
 9129  032100      022                        .BYTE   PAKACK          ;PACK ACKNOWLEDGE COMMAND
 9130  032101      003                        .BYTE   3
 9131
 9132  032102      000                        .BYTE                   ;END OF TABLE
 9133  032103      377                        .BYTE   -1
 9134
 9135  032104                         200$:                           ;END OF TEST
 9136
 9137                                   ;;*****************************************************************
 9138                                   ;*TEST 54        ILLEGAL FUNCTION TEST
 9139
 9140                                   ;;*****************************************************************
 9141  032104  000004                 T54:  SCOPE
 9142  032106  012737  000054  001226        MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 9143
 9144  032114  000240                        NOP
 9145  032116  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
 9146  032124  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 9147  032132  012737  032146  001122        MOV     #T54,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 9148  032140  012737  032146  001124        MOV     #T54,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 9149  032146                         T54:
 9150  032146  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
 9151  032152  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 9152  032156  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 9153  032162  005002                        CLR     R2              ;INITIALIZE FUNCTION CODE VALUE
 9154  032164                         10$:
 9155                                   ;SET VOLUME VALID USING SUBROUTINE
 9156  032164  004737  060526                JSR     PC,SETVV        ;GO SET VOLUME VALID
 9157  032170  000402                        BR      20$             ;BRANCH TO 20$ IF NO ERROR
 9158  032172  104000                        ERROR                   ;RETURN HERE IF ERROR
 9159  032174  000460                        BR      50$             ;SKIP TEST IF ERROR
 9160  032176  012704  000002         20$:  MOV     #2,R4           ;R4=CLOCK COUNT
 9161                                   ;EXECUTE THE TEST FUNCTION CODE AND VERIFY ILF
 9162
 9163  032202  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9164  032210  010203                        MOV     R2,R3           ;SETUP FUNCTION CODE IN R3
 9165  032212  052703  000001                BIS     #GO,R3
 9166
 9167  032216  010360  000000                MOV     R3,RMCS1(R0)    ;LOAD RMCS1
 9168  032222                         30$:
 9169
 9170  032222  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 9171
 9172  032230  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
9173  032236  005304                          DEC     R4
9174  032240  001370                          BNE     30$
9175
9176  032242  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
9177  032250  042737  177776  001142          BIC     #^CILF,$BDDAT   ;SETUP ACTUAL ILF STATUS
9178  032256  016237  065764  001140          MOV     FNCDTB(R2),$GDDAT  ;GET EXPECTED ILF STATUS
9179  032264  042737  177776  001140          BIC     #^CILF,$GDDAT
9180  032272  023737  001140  001142          CMP     $GDDAT,$BDDAT
9181  032300  001410                          BEQ     40$             ;BRANCH IF ILF IS OK
9182  032302  010037  001136                  MOV     R0,$BDADR       ;SETUP FOR ERROR MSG
9183  032306  062737  000014  001136          ADD     #RMER1,$BDADR
9184  032314  010237  001174                  MOV     R2,$TMP0
9185  032320  104171                          ERROR   171             ;ILF IS NOT CORRECT
9186  032322                          40$:
9187                                  ;ADVANCE TO THE NEXT FUNCTION CODE AND REPEAT TEST
9188  032322  062702  000002                  ADD     #2,R2
9189  032326  022702  000076                  CMP     #ILF76,R2
9190  032332  103401                          BLO     50$
9191  032334  000713                          BR      10$
9192  032336                          50$:                            ;END OF TEST
9193
9194                                  ;;*****************************************************************
9195                                  ;*TEST 55        OCCUPIED TEST
9196
9197                                  ;;*****************************************************************
9198  032336  000004                  TST55:  SCOPE
9199  032340  012737  000055  001226          MOV     #55,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9200
9201  032346  00024C                          NOP
9202  032350  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9203  032356  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9204  032364  012737  032400  001122          MOV     #T55,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9205  032372  012737  032400  001124          MOV     #T55,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9206  032400                          T55:
9207  032400  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9208  032404  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9209  032410  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9210  032414  005002                          CLR     R2                      ;INITIALIZE FUNCTION CODE
9211
9212  032416                          10$:
9213                                  ;GET THE DEVICE READY
9214                                  ;SET VOLUME VALID USING SUBROUTINE
9215  032416  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
9216  032422  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
9217  032424  104000                          ERROR                   ;RETURN HERE IF ERROR
9218  032426  000464                          BR      50$
9219  032430                          20$:
9220                                  ;ENABLE DEBUG CLOCK AND LOAD THE FUNCTION CODE
9221
9222  032430  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9223  032436  010203                          MOV     R2,R3                   ;ASSEMBLE FUNCTION CODE AND
9224  032440  052703  000001                  BIS     #GO,R3                  ;GO BIT IN R3
9225
9226  032444  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
9227  032450  012704  000002                  MOV     #2,R4                   ;R4=CLOCK COUNT
9228                                  ;STEP THE DEBUG CLOCK UNTIL SET PULSE IS ACTIVE
```

```
9229  032454                          30$:
9230
9231  032454  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9232
9233  032462  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9234  032470  005304                          DEC     R4
9235  032472  001370                          BNE     30$                     ;ISSUE NEXT CLOCK TIL R4 ZERO
9236                          ;VERIFY OCCUPIED STATUS
9237
9238  032474  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
9239  032502  042737  077777  001142          BIC     #^COCC,$BDDAT
9240  032510  005037  001140                  CLR     $GDDAT          ;GENERATE OCC FROM AOE
9241  032514  032762  001000  065764          BIT     #AOE,FNCDTB(R2)
9242  032522  001403                          BEQ     35$
```

```
DZRMJA.P11    01-AUG-77 11:10          T55     OCCUPIED TEST

 9243  032524  012737  100000  001140          MOV     #OCC,$GDDAT
 9244  032532  023737  001140  001142  35$:    CMP     $GDDAT,$BDDAT
 9245  032540  001411                          BEQ     40$                     ;BRANCH IF OCC IS OK
 9246  032542  010237  001174                  MOV     R2,$TMP0                ;SAVE FUNCTION CODE
 9247  032546  010037  001136                  MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS
 9248  032552  062737  000024  001136          ADD     #RMMR1,$BDADR
 9249  032560  104173                          ERROR   173                     ;OCCUPIED IS INCORRECT
 9250  032562  000406                          BR      50$
 9251  032564                          40$:
 9252                                  ;ADVANCE TO NEXT FUNCTIONCODE, EXIT IF DONE
 9253  032564  062702  000002                  ADD     #2,R2
 9254  032570  022702  000076                  CMP     #ILF76,R2
 9255  032574  103401                          BLO     50$                     ;EXIT IF DONE
 9256  032576  000707                          BR      10$
 9257
 9258  032600                          50$:                                    ;END OF TEST
 9259
 9260                                  ;;*********************************************************************
 9261                                  ;*TEST 56        READ IN PRESET TEST
 9262
 9263                                  ;;*********************************************************************
 9264  032600  000004                  TST56:  SCOPE
 9265  032602  012737  000056  001226          MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 9266
 9267  032610  000240                          NOP
 9268  032612  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
 9269  032620  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 9270  032626  012737  032642  001122          MOV     #T56,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 9271  032634  012737  032642  001124          MOV     #T56,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 9272  032642                          T56:
 9273  032642  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 9274  032646  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 9275  032652  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 9276                                  ;CLEAR AND ENABLE DEBUG CLOCK - LEAVE VOLUME VALID RESET
 9277  032656  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
 9278  032664  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
 9279
 9280  032670  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 9281
 9282  032676  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9283
 9284  032704  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 9285
 9286  032712  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 9287                                  ;LOAD ALL ONES IN RMDA, RMDC AND RMOF
 9288
 9289  032720  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
 9290
 9291  032726  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
 9292
 9293  032734  012760  177777  000032          MOV     #-1,RMOF(R0)    ;LOAD RMOF
 9294                                  ;LOAD READ IN PRESET COMMAND AND STEP THE CLOCK TILL SET PULSE
 9295
 9296  032742  012760  000021  000000          MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
 9297  032750  012702  000003                  MOV     #3,R2                   ;R2=CLOCK COUNT
 9298  032754                          10$:
```

```
9299
9300  032754  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9301
9302  032762  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9303  032770  005302                          DEC     R2
9304  032772  001370                          BNE     10$                     ;ISSUE 3 CLOCKS
9305                                  ;SEE IF RMDA OR RMDC OR RMOF IS ZERO
9306
9307  032774  016002  000006                  MOV     RMDA(R0),R2     ;STORE RMDA AT R2
9308  033000  005702                          TST     R2
9309  033002  001413                          BEQ     20$                     ;BRANCH IF RMDA IS ZERO
9310
9311  033004  016002  000034                  MOV     RMDC(R0),R2     ;STORE RMDC AT R2
9312  033010  042702  176000                  BIC     #XMUDC,R2               ;CLEAR UNUSED BITS
9313  033014  001406                          BEQ     20$                     ;BRANCH IF RMDC IS ZERO
9314
9315  033016  016002  000032                  MOV     RMOF(R0),R2     ;STORE RMOF AT R2
9316  033022  042702  161577                  BIC     #XMUOF,R2               ;CLEAR UNUSED BITS
9317  033026  001401                          BEQ     20$                     ;BRANCH IF RMOF IS ZERO
9318                                  ;READ IN PRESET COMMAND DIDNT CLEAR ANY OF THE 3 REGISTERS
9319  033030  104174                          ERROR   174                     ;READ IN PRESET FAILED
9320
9321  033032                          20$:                                    ;END OF TEST
9322
9323                                  ;;*****************************************************************
9324                                  ;*TEST 57       RIP/RMOF TEST
9325
9326                                  ;;*****************************************************************
9327  033032  000004                  T57:    SCOPE
9328  033034  012737  000057  001226          MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9329
9330  033042  000240                          NOP
9331  033044  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9332  033052  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9333  033060  012737  033074  001122          MOV     #T57,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9334  033066  012737  033074  001124          MOV     #T57,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9335  033074                          T57:
9336  033074  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9337  033100  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9338  033104  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9339  033110  010037  001136                  MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS AND
9340  033114  062737  000032  001136          ADD     #RMOF,$BDADR
9341  033122  005037  001140                  CLR     $GDDAT                  ;EXPECTED RMOF
9342
9343                                  ;INITIALIZE AND SET BITS IN RMOF
9344  033126  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
9345  033134  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
9346
9347  033140  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
9348
9349  033146  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9350
9351  033154  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9352
9353  033162  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER
9354
```

```
9355  033170  012760  177777  000032          MOV     #-1,RMOF(R0)    ;LOAD RMOF
9356                                  ;EXECUTE A READ IN PRESET IN DIAGNOSTIC MODE TILL SET PULSE
9357
9358  033176  012760  000021  000000          MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
9359  033204  012702  000003                  MOV     #3,R2                   ;R2=CLOCK COUNT
9360  033210                          10$:
9361
9362  033210  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9363
9364  033216  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9365  033224  005302                          DEC     R2
9366  033226  001370                          BNE     10$                     ;ISSUE 3 CLOCKS
9367                                  ;VERIFY THAT RMOF IS ZERO
9368
9369  033230  016037  000032  001142          MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
9370  033236  042737  161577  001142          BIC     #XMUOF,$BDDAT
9371  033244  001401                          BEQ     20$                     ;BRANCH IF RMOF IS ZERO
9372  033246  104175                          ERROR   175                     ;CANT CLEAR RMOF WITH RIP
9373
9374  033250                          20$:                                    ;END OF TEST
9375
9376                                  ;;*****************************************************************
9377                                  ;*TEST 60        RMDA/RMDC/RIP TEST
9378
9379                                  ;;*****************************************************************
9380  033250  000004                  T$T60:  SCOPE
9381  033252  012737  000060  001226          MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9382
9383  033260  000240                          NOP
9384  033262  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9385  033270  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9386  033276  012737  033312  001122          MOV     #T60,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9387  033304  012737  033312  001124          MOV     #T60,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9388  033312                          T60:
9389  033312  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9390  033316  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9391  033322  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9392  033326  005037  001140                  CLR     $GDDAT
9393
9394                                  ;CLEAR, ENABLE DEBUG CLOCK, THEN PRESET RMDA AND RMDC
9395  033332  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
9396  033340  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
9397
9398  033344  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
9399
9400  033352  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9401
9402  033360  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9403
9404  033366  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9405
9406  033374  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
9407
9408  033402  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
9409                                  ;EXECUTE READ IN PRESET TILL SET PULSE
9410
```

```
9411  033410  012760  000021  000000          MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
9412  033416  012702  000003                  MOV     #3,R2
9413  033422                          10$:
9414
9415  033422  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9416
9417  033430  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9418  033436  005302                          DEC     R2
9419  033440  001370                          BNE     10$                     ;ISSUE 3 CLOCKS
9420                          ;VERIFY RMDA IS ZERO
9421
9422  033442  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
9423  033450  005737  001142                  TST     $BDDAT
9424  033454  001406                          BEQ     20$                     ;BRANCH IF RMDA RESET
9425  033456  010037  001136                  MOV     R0,$BDADR
9426  033462  062737  000006  001136          ADD     #RMDA,$BDADR
9427  033470  104176                          ERROR   176                     ;RMDA NOT RESET BY RIP
9428  033472                          20$:
9429                          ;VERIFY RMDC IS ZERO
9430
9431  033472  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
9432  033500  042737  176000  001142          BIC     #XNUDC,$BDDAT
9433  033506  001406                          BEQ     30$                     ;BRANCH IF RMDC RESET
9434  033510  010037  001136                  MOV     R0,$BDADR
9435  033514  062737  000034  001136          ADD     #RMDC,$BDADR
9436  033522  104177                          ERROR   177                     ;RMDC NOT RESET BY RIP
9437
9438  033524                          30$:                                    ;END OF TEST
9439
9440                          ;;*****************************************************************
9441                          ;*TEST 61       OFFSET COMMAND TEST
9442
9443                          ;;*****************************************************************
9444  033524  000004          T$T61:  SCOPE
9445  033526  012737  000061  001226          MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9446
9447  033534  000240                          NOP
9448  033536  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9449  033544  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9450  033552  012737  033566  001122          MOV     #T61,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9451  033560  012737  033566  001124          MOV     #T61,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9452  033566                          T61:
9453  033566  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9454  033572  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9455  033576  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9456  033602  010037  001136                  MOV     R0,$BDADR
9457  033606  062737  000012  001136          ADD     #RMDS,$BDADR
9458  033614  012737  000001  001140          MOV     #OM,$GDDAT
9459
9460                          ;SET VOLUME VALID USING SUBROUTINE
9461  033622  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
9462  033626  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
9463  033630  104000                          ERROR                   ;RETURN HERE IF ERROR
9464  033632  000433                          BR      40$
9465  033634                          10$:
9466
```

```
9467  033634  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
9468                                  ;ENABLE DEBUG CLOCK AND EXECUTE OFFSET COMMAND
9469
9470  033642  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9471
9472  033650  012760  000015  000000          MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
9473  033656  012702  000002                  MOV     #2,R2                   ;R2=CLOCK COUNT
9474  033662                          20$:
9475
9476  033662  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9477
9478  033670  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9479  033676  005302                          DEC     R2
9480  033700  001370                          BNE     20$                     ;ISSUE 2 CLOCKS
9481                                  ;VERIFY THAT OFFSET MODE IS SET
9482
9483  033702  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9484  033710  042737  177776  001142          BIC     #^COM,$BDDAT
9485  033716  001001                          BNE     40$                     ;BRANCH IF OM IS SET
9486  033720  104200                          ERROR   200                     ;CANT SET ON BY COMMAND
9487  033722                          40$:                                    ;END OF TEST
9488
9489                                  ;;**********************************************************************
9490                                  ;*TEST 62        RETURN TO CENTER TEST
9491
9492                                  ;;**********************************************************************
9493  033722  000004                  TST62:  SCOPE
9494  033724  012737  000062  001226          MOV     #62,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9495
9496  033732  000240                          NOP
9497  033734  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9498  033742  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9499  033750  012737  033764  001122          MOV     #T62,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9500  033756  012737  033764  001124          MOV     #T62,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9501  033764                          T62:
9502  033764  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9503  033770  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9504  033774  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9505  034000  010037  001136                  MOV     R0,$BDADR
9506  034004  062737  000012  001136          ADD     #RMDS,$BDADR
9507                                  ;SET OFFSET DIRECTION AND OFFSET MODE
9508                                  ;SET VOLUME VALID USING SUBROUTINE
9509  034012  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
9510  034016  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
9511  034020  104000                          ERROR                   ;RETURN HERE IF ERROR
9512  034022  000465                          BR      60$
9513  034024                          10$:
9514
9515  034024  012760  000200  000032          MOV     #OFD,RMOF(R0)   ;LOAD RMOF
9516                                  ;SET OFFSET MODE USING SUBROUTINE
9517  034032  004737  060656                  JSR     PC,SETOM        ;GO SET OFFSET MODE
9518  034036  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
9519  034040  104000                          ERROR                   ;RETURN HERE IF ERROR
9520  034042                          20$:
9521                                  ;ENABLE DEBUG CLOCK AND EXECUTE RETURN TOCENTER COMMAND
9522                                  ;SET VOLUME VALID USING SUBROUTINE
```

```
                                T62     RETURN TO CENTER TEST

 9523  034042  004737  060526                JSR     PC,SETVV        ;GO SET VOLUME VALID
 9524  034046  000402                        BR      30$             ;BRANCH TO 30$ IF NO ERROR
 9525  034050  104000                        ERROR                   ;RETURN HERE IF ERROR
 9526  034052  000451                        BR      60$
 9527  034054                        30$:
 9528
 9529  034054  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9530
 9531  034062  012760  000017  000000        MOV     #RTC!GO,RMCS1(R0)       ;LOAD RMCS1
 9532  034070  012702  000002                MOV     #2,R2
 9533  034074                        40$:
 9534
 9535  034074  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 9536
 9537  034102  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9538  034110  005302                        DEC     R2
 9539  034112  001370                        BNE     40$                     ;ISSUE 2 CLOCKS
 9540                          ;VERIFY THAT OFFSET MODE IS RESET
 9541
 9542  034114  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 9543  034122  042737  177776  001142        BIC     #^tCOM,$BDDAT
 9544  034130  001403                        BEQ     50$                     ;BRANCH IF OFFSET MODE RESET
 9545  034132  005037  001140                CLR     $GDDAT
 9546  034136  104201                        ERROR   201                     ;CANT RESET OFFSETMODE BY RTC
 9547  034140                        50$:
 9548                          ;VERIFY THAT OFFSET DIRECTION IS RESET
 9549
 9550  034140  016037  000032  001142        MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
 9551  034146  042737  177577  001142        BIC     #^tCOFD,$BDDAT
 9552  034154  001410                        BEQ     60$                     ;BRANCH IF OFD IS RESET
 9553  034156  005037  001140                CLR     $GDDAT
 9554  034162  010037  001136                MOV     R0,$BDADR
 9555  034166  062737  000032  001136        ADD     #RMOF,$BDADR
 9556  034174  104202                        ERROR   202                     ;CANT RESET OFD BY RTC
 9557  034176                        60$:                                    ;END OF TEST
 9558
 9559                          ;;*****************************************************************
 9560                          ;*TEST 63        RMDC CLEAR OFFSET TEST
 9561
 9562                          ;;*****************************************************************
 9563  034176  000004                TST63:  SCOPE
 9564  034200  012737  000063  001226        MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 9565
 9566  034206  000240                        NOP
 9567  034210  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
 9568  034216  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 9569  034224  012737  034240  001122        MOV     #T63,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 9570  034232  012737  034240  001124        MOV     #T63,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 9571  034240                        T63:
 9572  034240  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
 9573  034244  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 9574  034250  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 9575  034254  010037  001136                MOV     R0,$BDADR
 9576  034260  062737  000012  001136        ADD     #RMDS,$BDADR
 9577                          ;SET VOLUME VALID USING SUBROUTINE
 9578  034266  004737  060526                JSR     PC,SETVV        ;GO SET VOLUME VALID
```

```
 9579  034272  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
 9580  034274  104000                          ERROR                   ;RETURN HERE IF ERROR
 9581  034276  000421                          BR      40$                     ;SKIP REST OF TEST
 9582  034300                          10$:
 9583                                  ;SET OFFSET MODE USING SUBROUTINE
 9584  034300  004737  060656                  JSR     PC,SETOM        ;GO SET OFFSET MODE
 9585  034304  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
 9586  034306  104000                          ERROR                   ;RETURN HERE IF ERROR
 9587  034310                          20$:
 9588                                  ;WRITE THE DESIRED CYLINDER REGISTER AND VERIFY THAT OFFSET IS ZERO
 9589
 9590  034310  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 9591
 9592  034316  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 9593  034324  042737  177776  001142          BIC     #^tCOM,$BDDAT
 9594  034332  001403                          BEQ     40$
 9595  034334  005037  001140                  CLR     $GDDAT
 9596  034340  104203                          ERROR   203                     ;CANT RESET OFFSET BY RMDC
 9597
 9598  034342                          40$:                            ;END OF TEST
 9599
 9600                                  ;;*****************************************************************
 9601                                  ;*TEST 64        EBL CLEAR OFFSET TEST
 9602
 9603                                  ;;*****************************************************************
 9604  034342  000004                  TST64:  SCOPE
 9605  034344  012737  000064  001226          MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 9606
 9607  034352  000240                          NOP
 9608  034354  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
 9609  034362  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 9610  034370  012737  034404  001122          MOV     #T64,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 9611  034376  012737  034404  001124          MOV     #T64,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 9612  034404                          T64:
 9613  034404  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 9614  034410  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
 9615  034414  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
 9616  034420  010037  001136                  MOV     R0,$BDADR               ;SETUP REGISTER FOR ERROR MSG
 9617  034424  062737  000012  001136          ADD     #RMDS,$BDADR
 9618                                  ;SET VOLUME VALID USING SUBROUTINE
 9619  034432  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
 9620  034436  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
 9621  034440  104000                          ERROR                   ;RETURN HERE IF ERROR
 9622  034442  000440                          BR      30$                     ;SKIP REST OF TEST IF ERROR
 9623  034444                          10$:
 9624                                  ;SET OFFSET MODE USING SUBROUTINE
 9625  034444  004737  060656                  JSR     PC,SETOM        ;GO SET OFFSET MODE
 9626  034450  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
 9627  034452  104000                          ERROR                   ;RETURN HERE IF ERROR
 9628  034454                          20$:
 9629                                  ;SET 16 BIT FORMAT AND LOAD LAST SECTOR/TRACK ADDRESS
 9630
 9631  034454  012760  010000  000032          MOV     #FMT16,RMOF(R0) ;LOAD RMOF
 9632
 9633  034462  012760  002037  000006          MOV     #002037,RMDA(R0)        ;LOAD RMDA
 9634                                  ;FORCE END OF BLOCK AND VERIFY THAT OFFSET MODE IS CLEARED
```

```
9635
9636  034470  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9637
9638  034476  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
9639
9640  034504  012760  061001  000024          MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
9641
9642  034512  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9643
9644  034520  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
9645  034526  042737  177776  001142          BIC     #^C0M,$BDDAT
9646  034534  001403                          BEQ     30$                     ;BRANCH IF OFFSET IS ZERO
9647  034536  005037  001140                  CLR     $GDDAT
9648  034542  104204                          ERROR   204                     ;OFFSET NO CLEARED BY EBL
9649  034544                          30$:                                    ;END OF TEST
9650
9651                                  ;;**********************************************************************
9652                                  ;*TEST 65        RUN AND GO TEST
9653
9654                                  ;;**********************************************************************
9655  034544  000004                  TST65:  SCOPE
9656  034546  012737  000065  001226          MOV     #65,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9657  034554  000240                          NOP
9658  034556  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9659  034564  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9660  034572  012737  034606  001122          MOV     #T65,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9661  034600  012737  034606  001124          MOV     #T65,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9662  034606                          T65:
9663  034606  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9664  034612  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9665  034616  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9666  034622  005037  001140                  CLR     $GDDAT                  ;INITIALIZE EXPECTED RESULT
9667  034626  005002                          CLR     R2                      ;INTIALIZE FUNCTION CODE
9668  034630  010037  001136                  MOV     R0,$BDADR
9669  034634  062737  000024  001136          ADD     #RMMR1,$BDADR
9670  034642                          10$:
9671                                  ;CLEAR THE MASSBUS AND ENABLE DEBUG CLOCK
9672  034642  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
9673  034650  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
9674
9675  034654  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
9676
9677  034662  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
9678
9679  034670  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
9680
9681  034676  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
9682                                  ;LOAD THE FUNCTION CODE AND VERIFY RUN AND GO FLOP
9683
9684  034704  010203                          MOV     R2,R3                   ;ASSEMBLE FUNCTION CODE AND GO
9685  034706  052703  000001                  BIS     #GO,R3
9686  034712  012737  000200  001524          MOV     #200,WATCH      ;SET WATCHDOG TIMER VALUE
9687  034720  004777  144602                  JSR     PC,@CLOCK       ;START THE CLOCK
9688
9689  034724  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
9690  034730                          15$:
```

```
                                                
034730  016037  000024  001142            MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
034736  042737  137777  001142            BIC     #^CRG,$BDDAT
034744  023737  001140  001142            CMP     $GDDAT,$BDDAT
034752  001411                            BEQ     20$                     ;BRANCH IF RUN AND GO FLOP OK
034754  005737  001524                    TST     WATCH           ;TAKE ANOTHER SAMPLE IF CLOCK NOT ZERO
034760  001363                            BNE     15$
034762  004777  144542                    JSR     PC,@STOP        ;STOP THE CLOCK
034766  010237  001174                    MOV     R2,$TMP0                ;SAVE FUNCTION CODE FOR MSG
034772  104205                            ERROR   20$                     ;RUN AND GO INCORRECT
034774  000416                            BR      40$                     ;SKIP REST OF
034776                          20$:
034776  004777  144526                    JSR     PC,@STOP        ;STOP THE CLOCK
                                ;ADVNACE TO NEXT FUNCTION CODE - EXIT IF DONE
035002  062702  000002                    ADD     #2,R2
035006  022702  000076                    CMP     #ILF76,R2
035012  103407                            BLO     40$                     ;EXIT IF DONE
035014  020227  000050                    CMP     R2,#WCD                 ;CHANGE EXPECTED RESULTI IF
035020  103403                            BLO     30$                     ;DATA COMMAND
035022  012737  040000  001140            MOV     #RG,$GDDAT
035030  000704                  30$:      BR      10$                     ;REPEAT TEST
                                
035032                          40$:                                    ;END OF TEST
                                ;;*****************************************************************
                                ;*TEST 66        SET IAE TEST
                                
                                ;;*****************************************************************
035032  000004                  T66:      SCOPE
035034  012737  000066  001226            MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                
035042  000240                            NOP
035044  012737  000024  001120            MOV     #20.,$ICNT      ;20 ITERATIONS
035052  112737  000001  001131            MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
035060  012737  035074  001122            MOV     #T66,$LPADR  ;LOAD LOOP ON TEST ADDRESS
035066  012737  035074  001124            MOV     #T66,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
035074                          T66:
035074  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
035100  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
035104  013701  001456                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
035110  012702  035256                    MOV     #100$,R2                ;R2=TABLE POINTER
035114                          10$:
                                ;SET VOLUME VALID USING SUBROUTINE
035114  004737  060526                    JSR     PC,SETVV        ;GO SET VOLUME VALID
035120  000402                            BR      20$             ;BRANCH TO 20$ IF NO ERROR
035122  104000                            ERROR                   ;RETURN HERE IF ERROR
035124  000453                            BR      50$                     ;SKIP REST OF TEST
035126                          20$:
                                ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT TO 18
                                
035126  012760  177777  000006            MOV     #-1,RMDA(R0)    ;LOAD RMDA
                                
035134  012760  177777  000034            MOV     #-1,RMDC(R0)    ;LOAD RMDC
                                
035142  012760  000000  000032            MOV     #0,RMOF(R0)     ;LOAD RMOF
                                ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCSI WITH GO ON
                                
```

```
 9747  035150  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9748  035156  111203                          MOVB    (R2),R3
 9749  035160  042703  177701                  BIC     #↑CILF76,R3
 9750  035164  052703  000001                  BIS     #GO,R3
 9751
 9752  035170  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
 9753  035174  116204  000001                  MOVB    1(R2),R4                ;GET CLOCK COUNT
 9754  035200  042704  177400                  BIC     #↑C377,R4
 9755  035204                          30$:
 9756                                  ;CLOCK THE COMMAND SEQUENCER
 9757
 9758  035204  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 9759
 9760  035212  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 9761  035220  005304                          DEC     R4
 9762  035222  001370                          BNE     30$
 9763                                  ;SEE IF IAE HAS SET
 9764
 9765  035224  016004  000014                  MOV     RMER1(R0),R4    ;STORE RMER1 AT R4
 9766  035230  042704  175777                  BIC     #↑CIAE,R4
 9767  035234  001007                          BNE     50$                     ;BRANCH IF IAE SET
 9768                                  ;IAE DID NOT SET - TRY ANOTHER FUNCTION CODE
 9769  035236  062702  000002                  ADD     #2,R2
 9770  035242  105762  000001                  TSTB    1(R2)
 9771  035246  100401                          BMI     40$                     ;BRANCH IF ALL CODES TRIED
 9772  035250  000721                          BR      10$
 9773  035252                          40$:
 9774                                  ;CANNOT SET IAE WITH ANY COMBINATION OF ADDRESS AND FUNCTION CODE
 9775  035252  104206                          ERROR   206
 9776  035254                          50$:
 9777  035254  000411                          BR      200$                    ;JUMP OVER TABLE
 9778
 9779  035256                          100$:
 9780                                  ;TABLE OF FUNCTION CODES AND CLOCK COUNTS FOR TEST
 9781
 9782  035256      030                         .BYTE   SEARCH                  ;SEARCH COMMAND
 9783  035257      002                         .BYTE   2
 9784
 9785  035260      004                         .BYTE   SEEK                    ;SEEK COMMAND
 9786  035261      002                         .BYTE   2
 9787
 9788  035262      062                         .BYTE   WH                      ;WRITE HEADER COMMAND
 9789  035263      002                         .BYTE   2
 9790
 9791  035264      052                         .BYTE   WCH                     ;WRITE CHECK HEADER COMMAND
 9792  035265      002                         .BYTE   2
 9793
 9794  035266      072                         .BYTE   RH                      ;READ HEADER COMMAND
 9795  035267      002                         .BYTE   2
 9796
 9797  035270      060                         .BYTE   WD                      ;WRITE DATA COMMAND
 9798  035271      002                         .BYTE   2
 9799
 9800  035272      050                         .BYTE   WCD                     ;WRITE CHECK DATA COMMAND
 9801  035273      002                         .BYTE   2
 9802
```

```
9803  035274  070                             .BYTE   RD                      ;READ DATA COMMAND
9804  035275  002                             .BYTE   2
9805
9806  035276  000                             .BYTE                           ;END OF TABLE
9807  035277  377                             .BYTE   -1
9808
9809  035300                          200$:                                   ;END OF TEST
9810
9811                                  ;;*****************************************************************
9812                                  ;*TEST 67        SEARCH, SEEK, READ WRITE TEST
9813
9814                                  ;;*****************************************************************
9815  035300  000004                 TST67:  SCOPE
9816  035302  012737  000067  001226          MOV     #67,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9817
9818  035310  000240                          NOP
9819  035312  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
9820  035320  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
9821  035326  012737  035342  001122          MOV     #T67,$LPADR  ;LOAD LOOP ON TEST ADDRESS
9822  035334  012737  035342  001124          MOV     #T67,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
9823  035342                          T67:
9824  035342  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
9825  035346  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
9826  035352  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
9827  035356  005002                          CLR     R2                      ;INITIALIZE FUNCTION CODE
9828  035360                          10$:
9829                                  ;SET VOLUME VALID USING SUBROUTINE
9830  035360  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
9831  035364  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
9832  035366  104000                          ERROR                   ;RETURN HERE IF ERROR
9833  035370  000472                          BR      50$
9834  035372                          20$:
9835                                  ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT
9836                                  ;TO 18 BIT MODE
9837
9838  035372  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
9839
9840  035400  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
9841
9842  035406  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
9843                                  ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCS1 WITH GO ON
9844
9845  035414  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9846  035422  010203                          MOV     R2,R3                   ;ASSEMBLE CODE AND GO
9847  035424  052703  000001                  BIS     #GO,R3
9848
9849  035430  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
9850                                  ;CLOCK THE COMMAND SEQUENCER TO SET PULSE
9851  035434  012704  000002                  MOV     #2,R4
9852  035440                          30$:
9853
9854  035440  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9855
9856  035446  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9857  035454  005304                          DEC     R4
9858  035456  001370                          BNE     30$
```

```
                                        ;VERIFY IAE ACCORDING TO FUNCTION CODE TABLE
                                        
035460  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
035466  042737  175777  001142          BIC     #^CIAE,$BDDAT
035474  016237  065764  001140          MOV     FNCDTB(R2),$GDDAT       ;ASSEMBLE EXPECTED IAE
035502  042737  175777  001140          BIC     #^CIAE,$GDDAT
035510  023737  001140  001142          CMP     $GDDAT,$BDDAT
035516  001411                          BEQ     40$                     ;BRANCH IF IAE OK
035520  010037  001136                  MOV     R0,$BDADR               ;SET UP ERROR MSG
035524  062737  000014  001136          ADD     #RMER1,$BDADR
035532  010237  001174                  MOV     R2,$TMP0
035536  104207                          ERROR   207                     ;IAE IS INCORRECT
035540  000406                          BR      50$                     ;SKIP REST OF TEST
035542                                  40$:
                                        ;ADVANCE TO NEXT FUNCTIONCODE - EXIT IF DONE
035542  062702  000002                          ADD     #2,R2
035546  023702  000076                          CMP     ILF76,R2
035552  103401                          BLO     50$
035554  000701                          BR      10$
035556                                  50$:                            ;END OF TEST
                                        
                                        ;;*******************************************************
                                        ;*TEST 70        INVALID SECTOR/TRACK TEST
                                        
                                        ;;*******************************************************
035556  000004                          T$T70:  SCOPE
035560  012737  000070  001226                  MOV     #70,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                        
035566  000240                          NOP
035570  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
035576  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
035604  012737  035620  001122          MOV     #T70,$LPADR  ;LOAD LOOP ON TEST ADDRESS
035612  012737  035620  001124          MOV     #T70,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
035620                                  T70:
035620  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
035624  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
035630  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
035634  012702  036014                  MOV     #100$,R2                ;INITIALIZE TABLE POINTER
                                        
035640                                  10$:
                                        ;SET VOLUME VALID USING SUBROUTINE
035640  004737  060526                          JSR     PC,SETVV        ;GO SET VOLUME VALID
035644  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
035646  104000                          ERROR                   ;RETURN HERE IF ERROR
035650  000460                          BR      50$                     ;SKIP REST OF TEST
035652                                  20$:
                                        ;CLEAR DESIRED CYLINDER, LOAD SECTOR/TRACK ADDRESS FROM TABLE, AND SET
                                        ;18 BIT FORMAT
                                        
035652  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
                                        
035660  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
                                        
035666  011260  000006                  MOV     (R2),RMDA(R0)   ;LOAD RMDA
                                        ;EXECUTE A SEARCH COMMAND TO WHERE "SET PULSE" IS ACTIVE
```

```
9915  035672  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9916
9917  035700  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
9918  035706  012703  000002                  MOV     #2,R3
9919  035712                          30$:
9920
9921  035712  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
9922
9923  035720  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
9924  035726  005303                          DEC     R3
9925  035730  001370                          BNE     30$                     ;ISSUE 2 CLOCKS
9926                          ;VERIFY IAE IS SET
9927
9928  035732  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
9929  035740  042737  175777  001142          BIC     #^C<IAE>,$BDDAT
9930  035746  001014                          BNE     40$                     ;BRANCH IF IAE IS ON
9931  035750  012737  002000  001140          MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
9932  035756  010037  001136                  MOV     R0,$BDADR
9933  035762  062737  000014  001136          ADD     #RMER1,$BDADR
9934  035770  011237  001174                  MOV     (R2),$TMP0
9935  035774  104210                          ERROR   210                     ;IAE NOT SET BY RMDA ADDRESS
9936  035776  000405                          BR      50$
9937  036000                          40$:
9938                          ;ADVANCE TO NEXT ENTRY IN TABLE - EXIT IF DONE
9939  036000  062702  000002                  ADD     #2,R2
9940  036004  005712                          TST     (R2)
9941  036006  001401                          BEQ     50$                     ;EXIT IF END OF TABLE
9942  036010  000713                          BR      10$                     ;REPEAT TEST
9943  036012  000414                  50$:    BR      200$                    ;JUMP OVER TABLE
9944
9945  036014                          100$:
9946
9947                          ;TABLE OF SECTOR AND TRACK ADDRESSES FOR TEST
9948  036014      000                         .BYTE   0                       ;SECTOR ADDRESS = 0
9949  036015      005                         .BYTE   ^B00000101              ;TRACK ADDRESS = 5
9950
9951  036016      000                         .BYTE   0                       ;SECTOR = 0
9952  036017      006                         .BYTE   ^B00000110              ;TRACK  = 6
9953
9954  036020      000                         .BYTE   0                       ;SECTOR = 0
9955  036021      010                         .BYTE   ^B00001000              ;TRACK = 8
9956
9957  036022      000                         .BYTE   0                       ;SECTOR = 0
9958  036023      020                         .BYTE   ^B00010000              ;TRACK = 16
9959
9960  036024      000                         .BYTE   0                       ;SECTOR = 0
9961  036025      040                         .BYTE   ^B00100000              ;TRACK = 32
9962
9963  036026      000                         .BYTE   0                       ;SECTOR = 0
9964  036027      100                         .BYTE   ^B01000000              ;TRACK = 64
9965
9966  036030      000                         .BYTE   0                       ;SECTOR = 0
9967  036031      200                         .BYTE   ^B10000000              ;TRACK = 128
9968
9969  036032      036                         .BYTE   ^B00011110              ;SECTOR = 31
9970  036033      000                         .BYTE   0                       ;TRACK = 0
```

```
 9971
 9972  036034  040                             .BYTE   ^B00100000              ;SECTOR = 32
 9973  036035  000                             .BYTE   0                       ;TRACK = 0
 9974
 9975  036036  100                             .BYTE   ^B01000000              ;SECTOR = 64
 9976  036037  000                             .BYTE   0                       ;TRACK = 0
 9977
 9978  036040  200                             .BYTE   ^B10000000              ;SECTOR = 128
 9979  036041  000                             .BYTE   0                       ;TRACK = 0
 9980
 9981  036042  000                             .BYTE   0                       ;END OF TABLE
 9982  036043  000                             .BYTE   0
 9983
 9984  036044                          200$:                                   ;END OF TEST
 9985
 9986                                  ;;*************************************************************
 9987                                  ;*TEST 71        INVALID CYLINDER TEST
 9988
 9989                                  ;;*************************************************************
 9990  036044  000004                  T$T71:  SCOPE
 9991  036046  012737  000071  001226          MOV     #71,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 9992
 9993  036054  000240                          NOP
 9994  036056  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
 9995  036064  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
 9996  036072  012737  036106  001122          MOV     #T71,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 9997  036100  012737  036106  001124          MOV     #T71,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 9998  036106                          T71:
 9999  036106  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
100000  036112  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
100001  036116  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
100002  036122  012702  036274                  MOV     #10O$,R2                ;INITIALIZE TABLE POINTER
100003  036126                          10$:
100004                                  ;SET VOLUME VALID USING SUBROUTINE
100005  036126  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
100006  036132  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
100007  036134  104000                          ERROR                   ;RETURN HERE IF ERROR
100008  036136  000455                          BR      50$                     ;SKIP IF ERROR
100009  036140                          20$:
100010                                  ;CLEAR RMDA, LOAD RMDC FROM TABLE
100011
100012  036140  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
100013
100014  036146  011260  000034                  MOV     (R2),RMDC(R0)   ;LOAD RMDC
100015                                  ;ENABLE DEBUG CLOCK AND EXECUTE SEARCH COMMAND
100016
100017  036152  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
100018
100019  036160  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
100020  036166  012703  000002                  MOV     #2,R3
100021  036172                          30$:
100022
100023  036172  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
100024
100025  036200  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
100026  036206  005303                          DEC     R3
```

```
10027  036210  001370                          BNE     30$                     ;ISSUE 2 CLOCKS
10028                                  ;VERIFY IAE IS SET
10029
10030  036212  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
10031  036220  042737  175777  001142          BIC     #^C<IAE>,$BDDAT
10032  036226  001014                          BNE     40$                     ;BRANCH IF IAE IS SET
10033  036230  012737  002000  001140          MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
10034  036236  010037  001136                  MOV     R0,$BDADR
10035  036242  062737  000014  001136          ADD     #RMER1,$BDADR
10036  036250  011237  001174                  MOV     (R2),$TMP0
10037  036254  104211                          ERROR   211                     ;IAE NOT SET BY RMDC
10038  036256  000405                          BR      50$
10039  036260                         40$:
10040                                  ;ADVANCE TABBLE POINTER - EXIT IF DONE
10041  036260  062702  000002                  ADD     #2,R2
10042  036264  005712                          TST     (R2)
10043  036266  001401                          BEQ     50$
10044  036270  000716                          BR      10$
10045  036272  000405                 50$:     BR      200$                    ;JUMP OVER TABLE
10046
10047  036274                         100$:
10048
10049                                  ;TABLE OF CYLINER ADDRESSES USED DURING TEST
10050
10051  036274  001467                          .WORD   ^B1100110111            ;CYLINDER 823
10052
10053  036276  001470                          .WORD   ^B1100111000            ;CYLINDER 824
10054
10055  036300  001500                          .WORD   ^B1101000000            ;CYLINDER 832
10056
10057  036302  001600                          .WORD   ^B1110000000            ;CYLINDER 896
10058
10059  036304  000000                          .WORD   0                       ;END OF TABLE
10060
10061  036306                         200$:                                    ;END OF TEST
10062
10063                                  ;;*****************************************************************
10064                                  ;*TEST 72        SET AOE TEST
10065
10066                                  ;;*****************************************************************
10067  036306  000004                 T72:     SCOPE
10068  036310  012737  000072  001226          MOV     #72,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
10069
10070  036316  000240                          NOP
10071  036320  012737  000024  001120          MOV     #20.,$ICNT              ;20 ITERATIONS
10072  036326  112737  000001  001131          MOVB    #1,$ERMAX               ;ONE ERROR ALLOWED
10073  036334  012737  036350  001122          MOV     #T72,$LPADR     ;LOAD LOOP ON TEST ADDRESS
10074  036342  012737  036350  001124          MOV     #T72,$LPERR     ;LOAD LOOP ON ERROR ADDRESS
10075  036350                         T72:
10076  036350  012706  00' 10                  MOV     #STACK,SP               ;LOAD THE STACK POINTER
10077  036354  013700  0C  76                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS OF UUT
10078  036360  013701  001456                  MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
10079                                  ;SET VOLUME VALID USING SUBROUTINE
10080  036364  004737  060526                  JSR     PC,SETVV                ;GO SET VOLUME VALID
10081  036370  000402                          BR      10$                     ;BRANCH TO 10$ IF NO ERROR
10082  036372  104000                          ERROR                           ;RETURN HERE IF ERROR
```

```
10083  036374  000465                          BR      30$                     ;SKIP TEST IF ERROR
10084  036376                          10$:
10085                                  ;ENABLE DEBUG CLOCK AND LOAD LAST SECTOR ADDRESS, MEMORY ADDRESS AND
10086                                  ;WORD COUNT, THEN LOAD WRITE DATA COMMAND WITH GO SET
10087
10088  036376  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10089
10090  036404  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
10091
10092  036412  012760  002035  000006          MOV     #002035,RMDA(R0)        ;LOAD RMDA
10093
10094  036420  012760  001466  000034          MOV     #001466,RMDC(R0)        ;LOAD RMDC
10095
10096  036426  012760  106652  000004          MOV     #BUFFER,RMBA(R0)        ;LOAD RMBA
10097
10098  036434  012760  177244  000002          MOV     #^C<2*256>+1,RMWC(R0)   ;LOAD RMWC
10099
10100  036442  012760  000061  000000          MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
10101  036450  012702  000002                  MOV     #2,R2                   ;R2 = CLOCK COUNT
10102                                  ;CLOCK THE COMMAND SEQUENCER TO GENERATE SET PULSE
10103  036454                          20$:
10104
10105  036454  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
10106
10107  036462  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10108  036470  005302                          DEC     R2
10109  036472  001370                          BNE     20$                     ;ISSUE 2 CLOCKS
10110                                  ;FORCE EBL TO GET TO OVERFLOW ADDRESS
10111
10112  036474  012760  061001  000024          MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
10113
10114  036502  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10115                                  ;VERIFY THAT ADDRESS OVERFLOW ERROR IS SET
10116
10117  036510  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
10118  036516  042737  176777  001142          BIC     #^CAOE,$BDDAT
10119  036524  001011                          BNE     30$                     ;BRANCH IF AOE IS SET
10120  036526  010037  001136                  MOV     R0,$BDADR               ;SETUP ERROR MESSAGE
10121  036532  062737  000014  001136          ADD     #RMER1,$BDADR
10122  036540  012737  001000  001140          MOV     #AOE,$GDDAT
10123  036546  104212                          ERROR   212                     ;CANT SET AOE
10124  036550                          30$:                                    ;END OF TEST
10125
10126                                  ;;*****************************************************************
10127                                  ;*TEST 73        SET RMR TEST
10128
10129                                  ;;*****************************************************************
10130  036550  000004                  T$T73:  SCOPE
10131  036552  012737  000073  001226          MOV     #73,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10132
10133  036560  000240                          NOP
10134  036562  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10135  036570  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10136  036576  012737  036612  001122          MOV     #T73,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10137  036604  012737  036612  001124          MOV     #T73,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10138  036612                          T73:
```

```
10139  036612  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10140  036616  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10141  036622  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10142  036626  010037  001136                  MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS FOR MSG
10143  036632  062737  000014  001136          ADD     #RMER1,$BDADR
10144  036640  012702  037014                  MOV     #100$,R2                ;INITIALIZE TABLE POINTER
10145  036644                          10$:
10146                                  ;CLEAR THE DEVICE AND ENABLE DEBUG CLOCK
10147  036644  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10148  036652  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10149
10150  036656  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10151
10152  036664  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10153
10154  036672  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
10155
10156  036700  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
10157                                  ;SET GO THEN WRITE THE REGISTER SPECIFIED BY THE TABLE
10158
10159  036706  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
10160  036714  011203                          MOV     (R2),R3                 ;GENERATE REGISTER ADDRESS
10161  036716  060003                          ADD     R0,R3
10162  036720  012713  041001                  MOV     #DMD!MUR!DBEN,(R3)      ;WRITE THE REGISTER
10163                                  ;VERIFY RMR ACCORDING TO TABLE
10164
10165  036724  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
10166  036732  042737  177773  001142          BIC     #^CRMR,$BDDAT
10167  036740  016237  000002  001140          MOV     2(R2),$GDDAT            ;GET EXPECTED RESULT FROM TABLE
10168  036746  023737  001140  001142          CMP     $GDDAT,$BDDAT
10169  036754  001411                          BEQ     30$                     ;BRANCH IF RMR IS OK
10170  036756  011237  001174                  MOV     (R2),$TMP0              ;SAVE TEST REGISTER
10171  036762  032762  000004  000002          BIT     #RMR,2(R2)
10172  036770  001002                          BNE     20$                     ;BRANCH IF ERROR SHOULD BE ONE
10173  036772  104213                          ERROR   213                     ;RMR SET WHEN WRITING REGISTER
10174  036774  000401                          BR      30$
10175  036776  104214                  20$:    ERROR   214                     ;RMR DID NOT SET
10176
10177  037000                          30$:
10178                                  ;ADVANCE TABLE POINTER, EXIT IF DONE
10179  037000  062702  000004                  ADD     #4,R2
10180  037004  005712                          TST     (R2)
10181  037006  100401                          BMI     40$                     ;EXIT IF ENTRY NEGATIVE
10182  037010  000715                          BR      10$
10183  037012  000410                  40$:    BR      200$                    ;JUMP OVER TABLE
10184
10185  037014                          100$:
10186
10187                                  ;TABLE OF REGISTER ADDRESSES AND RMR VALUES
10188
10189  037014  000016                          .WORD   #RMAS                   ;ATTENTION SUMMARY REG
10190  037016  000000                          .WORD                           ;RMR = 0
10191
10192  037020  000024                          .WORD   #RMMR1                  ;MAINTENANCE REG
10193  037022  000000                          .WORD                           ;RMR = 0
10194
```

```
10195  037024  000006                          .WORD   #RMDA                   ;DISK ADDRESS REG
10196  037026  000004                          .WORD   RMR                     ;RMR = 1
10197
10198  037030  177777                          .WORD   #-1                     ;END OF TABLE
10199  037032  000000                          .WORD
10200
10201  037034                          200$:                                   ;END OF TEST
10202
10203                                  ;;*****************************************************************
10204                                  ;*TEST 74       PGM STATUS CHECK
10205
10206                                  ;;*****************************************************************
10207  037034  000004                  TST74:  SCOPE
10208  037036  012737  000074  001226          MOV     #74,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10209
10210  037044  000240                          NOP
10211  037046  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10212  037054  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10213  037062  012737  037076  001122          MOV     #T74,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10214  037070  012737  037076  001124          MOV     #T74,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10215  037076                          T74:
10216  037076  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10217  037102  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10218  037106  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10219                                  ;CLEAR AND READ DRIVE TYPE AND DRIVE STATUS
10220  037112  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10221  037120  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10222
10223  037124  016037  000026  001174          MOV     RMDT(R0),$TMP0  ;STORE RMDT AT $TMP0
10224
10225  037132  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10226                                  ;OMIT TEST IF DRQ IS ON - ELSE VERIFY THAT PGM IS OFF
10227  037140  032737  004000  001174          BIT     #DRQ,$TMP0
10228  037146  001014                          BNE     10$                     ;BRANCH IF DRQ IS ON
10229  037150  042737  176777  001142          BIC     #^CPGM,$BDDAT
10230  037156  001410                          BEQ     10$                     ;BRANCH IF PGM IS OFF
10231  037160  005037  001140                  CLR     $GDDAT
10232  037164  010037  001136                  MOV     R0,$BDADR
10233  037170  062737  000012  001134          ADD     #RMDS,$GDADR
10234  037176  104215                          ERROR   215                     ;DRQ IS OFF BUT PGM IS ON
10235  037200                          10$:                                    ;END OF TEST
10236
10237                                  ;;*****************************************************************
10238                                  ;*TEST 75       DPR STATUS CHECK
10239
10240                                  ;;*****************************************************************
10241  037200  000004                  TST75:  SCOPE
10242  037202  012737  000075  001226          MOV     #75,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10243
10244  037210  000240                          NOP
10245  037212  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10246  037220  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10247  037226  012737  037242  001122          MOV     #T75,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10248  037234  012737  037242  001124          MOV     #T75,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10249  037242                          T75:
10250  037242  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
```

```
10251  037246  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10252  037252  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10253                                  ;CLEAR AND VERIFY THAT DVA IS SET
10254  037256  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10255  037264  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10256
10257  037270  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
10258  037276  042737  173777  001142          BIC     #^CDVA,$BDDAT
10259  037304  001006                          BNE     10$                     ;BRANCH IF DVA IS ON
10260  037306  012737  004000  001140          MOV     #DVA,$GDDAT
10261  037314  010037  001136                  MOV     R0,$BDADR
10262  037320  104216                          ERROR   216                     ;DVA NOT SET
10263  037322                                  10$:
10264                                  ;VERIFY THAT DPR IS SET
10265
10266  037322  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10267  037330  042737  177377  001142          BIC     #^CDPR,$BDDAT
10268  037336  001011                          BNE     20$
10269  037340  012737  000400  001140          MOV     #DPR,$GDDAT
10270  037346  010037  001136                  MOV     R0,$BDADR
10271  037352  062737  000012  001136          ADD     #RMDS,$BDADR
10272  037360  104217                          ERROR   217                     ;DPR NOT SET
10273
10274  037362                                  20$:                            ;END OF TEST
10275
10276                                  ;;******************************************************************
10277                                  ;*TEST 76       PORT REQUEST TEST, PART 1
10278
10279                                  ;;******************************************************************
10280  037362  000004                  T$T76:  SCOPE
10281  037364  012737  000076  001226          MOV     #76,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10282
10283  037372  000240                          NOP
10284  037374  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10285  037402  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10286  037410  012737  037424  001122          MOV     #T76,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10287  037416  012737  037424  001124          MOV     #T76,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10288  037424                                  T76:
10289  037424  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10290  037430  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10291  037434  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10292                                  ;SET VOLUME VALID USING SUBROUTINE
10293  037440  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
10294  037444  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
10295  037446  104000                          ERROR                   ;RETURN HERE IF ERROR
10296  037450  000434                          BR      20$
10297  037452                                  10$:
10298                                  ;EXECUTE A RELEASE TO RESET REQUEST FLOP
10299
10300  037452  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
10301                                  ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10302
10303  037460  016002  000040                  MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
10304  037464  042702  037777                  BIC     #^C<RQA!RQB>,R2
10305  037470  001024                          BNE     20$
10306                                  ;READ RMCS1 TO SET REQUEST FLOP
```

```
10307
10308  037472  016002  000000                  MOV     RMCS1(R0),R2    ;STORE RMCS1 AT R2
10309                                  ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10310
10311  037476  016005  000012                  MOV     RMDS(R0),R5     ;STORE RMDS AT R5
10312  037502  032705  001000                  BIT     #PGM,R5         ;SEE IF PGM IS SET
10313  037506  001415                          BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
10314
10315  037510  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
10316  037516  042737  037777  001142          BIC     #^C<RQA!RQB>,$BDDAT
10317  037524  001006                          BNE     20$                     ;BRANCH IF REQUEST IS SET
10318  037526  010037  001136                  MOV     R0,$BDADR
10319  037532  062737  000040  001136          ADD     #RMMR2,$BDADR
10320  037540  104220                          ERROR   220                     ;CANT SET REQUEST BY RMCS1
10321  037542                          20$:
10322
10323                                  ;;*****************************************************************
10324                                  ;*TEST 77        PORT REQUEST TEST, PART 2
10325
10326                                  ;;*****************************************************************
10327  037542  000004                  T$T77:  SCOPE
10328  037544  012737  000077  001226          MOV     #77,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
10329
10330  037552  000240                          NOP
10331  037554  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10332  037562  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10333  037570  012737  037604  001122          MOV     #T77,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10334  037576  012737  037604  001124          MOV     #T77,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10335  037604                          T77:
10336  037604  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10337  037610  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10338  037614  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10339                                  ;SET VOLUME VALID USING SUBROUTINE
10340  037620  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
10341  037624  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
10342  037626  104000                          ERROR                   ;RETURN HERE IF ERROR
10343  037630  000435                          BR      20$
10344  037632                          10$:
10345                                  ;EXECUTE A RELEASE TO RESET REQUEST FLOP
10346
10347  037632  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
10348                                  ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10349
10350  037640  016002  000040                  MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
10351  037644  042702  037777                  BIC     #^C<RQA!RQB>,R2
10352  037650  001025                          BNE     20$
10353                                  ;WRITE THE ATTENTION SUMMARY REGISTER TO SET REQUEST FLOP
10354
10355  037652  012760  177777  000016          MOV     #-1,RMAS(R0)    ;LOAD RMAS
10356                                  ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10357
10358  037660  016005  000012                  MOV     RMDS(R0),R5     ;STORE RMDS AT R5
10359  037664  032705  001000                  BIT     #PGM,R5         ;SEE IF PGM IS SET
10360  037670  001415                          BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
10361
10362  037672  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
```

```
10363  037700  042737  037777  001142          BIC     #↑C<RQA!RQB>,$BDDAT
10364  037706  001006                          BNE     20$
10365  037710  010037  001136                  MOV     R0,$BDADR
10366  037714  062737  000040  001136          ADD     #RMMR2,$BDADR
10367  037722  104221                          ERROR   221                     ;CANT SEE REQUEST BY RMAS
10368  037724                          20$:
10369
10370                                  ;;*****************************************************************
10371                                  ;*TEST 100      PORT REQUEST TEST, PART 3
10372
10373                                  ;;*****************************************************************
10374  037724  000004                  T$T100: SCOPE
10375  037726  012737  000100  001226          MOV     #100,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10376
10377  037734  000240                          NOP
10378  037736  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10379  037744  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10380  037752  012737  037766  001122          MOV     #T100,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10381  037760  012737  037766  001124          MOV     #T100,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10382  037766                          T100:
10383  037766  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10384  037772  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10385  037776  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10386                                  ;SET VOLUME VALID USING SUBROUTINE
10387  040002  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
10388  040006  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
10389  040010  104000                          ERROR                   ;RETURN HERE IF ERROR
10390  040012  000435                          BR      20$
10391  040014                          10$:
10392                                  ;EXECUTE A RELEASE COMMAND TO RESET REQUEST FLOP
10393
10394  040014  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
10395                                  ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
10396
10397  040022  016002  000040                  MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
10398  040026  042702  037777                  BIC     #↑C<RQA!RQB>,R2
10399  040032  001025                          BNE     20$
10400                                  ;WRITE RMDA TO SET REQUEST FLOP
10401
10402  040034  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
10403                                  ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
10404
10405  040042  016005  000012                  MOV     RMDS(R0),R5     ;STORE RMDS AT R5
10406  040046  032705  001000                  BIT     #PGM,R5         ;SEE IF PGM IS SET
10407  040052  001415                          BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
10408
10409  040054  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
10410  040062  042737  037777  001142          BIC     #↑C<RQA!RQB>,$BDDAT
10411  040070  001006                          BNE     20$
10412  040072  010037  001136                  MOV     R0,$BDADR
10413  040076  062737  000040  001136          ADD     #RMMR2,$BDADR
10414  040104  104222                          ERROR   222                     ;CANT SET REQUEST BY RMDA
10415
10416  040106                          20$:
10417
10418                                  ;;*****************************************************************
```

```
10419                                   ;*TEST 101      RELEASE TEST
10420
10421                                   ;;**************************************************************
10422  040106  000004                   TST101: SCOPE
10423  040110  012737  000101  001226           MOV     #101,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10424
10425  040116  000240                           NOP
10426  040120  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
10427  040126  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10428  040134  012737  040150  001122           MOV     #T101,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10429  040142  012737  040150  001124           MOV     #T101,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10430  040150                           T101:
10431  040150  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
10432  040154  013700  001276                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10433  040160  013701  001456                   MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10434                                   ;REQUEST FLOP SHOULD SET WHEN WRITING RMCS1
10435                                   ;SET VOLUME VALID USING SUBROUTINE
10436  040164  004737  060526                   JSR     PC,SETVV        ;GO SET VOLUME VALID
10437  040170  000402                           BR      10$             ;BRANCH TO 10$ IF NO ERROR
10438  040172  104000                           ERROR                   ;RETURN HERE IF ERROR
10439  040174  000454                           BR      40$
10440  040176                           10$:
10441                                   ;EXECUTE A RELEASE COMMAND
10442
10443  040176  012760  041001  000024           MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
10444
10445  040204  012760  000013  000000           MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
10446  040212  012702  000002                   MOV     #2,R2
10447  040216                           20$:
10448
10449  040216  012760  141001  000024           MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
10450
10451  040224  012760  041001  000024           MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
10452  040232  005302                           DEC     R2
10453  040234  001370                           BNE     20$                     ;ISSUE 2 CLOCKS
10454                                   ;VERIFY REQUEST FLOPS ARE RESET
10455
10456  040236  016037  000026  001174           MOV     RMDT(R0),$TMP0  ;STORE RMDT AT $TMP0
10457
10458  040244  016037  000040  001142           MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
10459  040252  042737  037777  001142           BIC     #^C<RQA!RQB>,$BDDAT
10460  040260  001422                           BEQ     40$                     ;BRANCH IF REQUESTS ARE RESET
10461  040262  032737  004000  001174           BIT     #DRQ,$TMP0
10462  040270  001410                           BEQ     30$                     ;BRANCH IF SINGLE PORT DEVICE
10463  040272  032737  100000  001142           BIT     #RQA,$BDDAT
10464  040300  001412                           BEQ     40$                     ;BRANCH IF RQA IS RESET
10465  040302  032737  040000  001142           BIT     #RQB,$BDDAT
10466  040310  001406                           BEQ     40$                     ;BRANCH IF RQB IS RESET
10467  040312                           30$:
10468                                   ;DRQ IS ZERO AND A REQUEST FLOP IS ON, OR, DRQ IS ONE AND
10469                                   ;BOTH REQUEST FLOPS ARE ON
10470  040312  010037  001136                   MOV     R0,$BDADR
10471  040316  062737  000040  001136           ADD     #RMMR2,$BDADR
10472  040324  104223                           ERROR   223                     ;REQUEST FLOP NOT RESET
10473  040326                           40$:                                   ;END OF TEST
10474
```

```
10475                                   ;;*****************************************************************
10476                                   ;*TEST 102      WRITE ATA TEST
10477
10478                                   ;;*****************************************************************
10479  040326  000004                   TST102: SCOPE
10480  040330  012737  000102  001226           MOV     #102,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10481
10482  040336  000240                           NOP
10483  040340  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
10484  040346  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10485  040354  012737  040370  001122           MOV     #T102,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10486  040362  012737  040370  001124           MOV     #T102,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10487  040370                           T102:
10488  040370  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
10489  040374  013700  001276                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10490  040400  013701  001456                   MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10491                                   ;CLEAR THE DEVICE, SET DIAGNOSTIC MODE THEN SET UNIT READY
10492  040404  012760  000040  000010           MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10493  040412  111160  000010                   MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10494
10495  040416  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10496
10497  040424  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1
10498
10499  040432  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2
10500
10501  040440  012760  001001  000024           MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
10502                                   ;WRITE THE ATTENTION SUMMARY REGISTER
10503  040446  111102                           MOVB    (R1),R2                 ;ASSEMBLE THE ATA BIT IN R3
10504  040450  042702  177770                   BIC     #^CUNTMSK,R2
10505  040454  116203  066064                   MOVB    ATNTBL(R2),R3
10506  040460  042703  177400                   BIC     #^CATNMSK,R3
10507
10508  040464  010360  000016                   MOV     R3,RMAS(R0)     ;LOAD RMAS
10509                                   ;READ RMDS AND VERIFY THAT ATA IS RESET
10510
10511  040470  016037  000012  001142           MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10512  040476  042737  077777  001142           BIC     #^CATA,$BDDAT
10513  040504  001411                           BEQ     10$                     ;BRANCH IF ATA IS RESET
10514  040506  010037  001136                   MOV     R0,$BDADR
10515  040512  062737  000012  001136           ADD     #RMDS,$BDADR
10516  040520  005037  001140                   CLR     $GDDAT
10517  040524  104224                           ERROR   224                     ;CANT CLEAR ATA
10518  040526  000417                           BR      20$
10519  040530                           10$:
10520                                   ;READ RMAS AND VERIFY ATA IS RESET
10521
10522  040530  016037  000016  001142           MOV     RMAS(R0),$BDDAT ;STORE RMAS AT $BDDAT
10523  040536  005103                           COM     R3
10524  040540  040337  001142                   BIC     R3,$BDDAT
10525  040544  001410                           BEQ     20$                     ;BRANCH IF ATA IS RESET
10526  040546  005037  001140                   CLR     $GDDAT
10527  040552  010037  001136                   MOV     R0,$BDADR
10528  040556  062737  000016  001136           ADD     #RMAS,$BDADR
10529  040564  104225                           ERROR   225                     ;RMAS ATA NOT ZERO
10530  040566                           20$:                                    ;END OF TEST
```

```
10531
10532                                   ;;*****************************************************************
10533                                   ;*TEST 103      RESET ATA BY GO TEST
10534
10535                                   ;;*****************************************************************
10536  040566  000004                   ↑ST103: SCOPE
10537  040570  012737  000103  001226           MOV     #103,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10538
10539  040576  000240                           NOP
10540  040600  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
10541  040606  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10542  040614  012737  040630  001122           MOV     #T103,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10543  040622  012737  040630  001124           MOV     #T103,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10544  040630                           T103:
10545  040630  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
10546  040634  013700  001276                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10547  040640  013701  001456                   MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10548  040644  012760  000040  000010           MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10549  040652  111160  000010                   MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10550
10551  040656  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10552
10553  040664  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1
10554
10555  040672  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2
10556
10557  040700  012760  041001  000024           MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10558                                   ;WITH DEBUG CLOCK ENABLED, SET GO AND VERIFY THAT ATA IS RESET
10559
10560  040706  012760  000001  000000           MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
10561
10562  040714  016037  000012  001142           MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10563  040722  042737  077777  001142           BIC     #↑CATA,$BDDAT
10564  040730  001410                           BEQ     10$                     ;BRANCH IF ATA IS RESET
10565  040732  005037  001140                   CLR     $GDDAT
10566  040736  010037  001136                   MOV     R0,$BDADR
10567  040742  062737  000012  001136           ADD     #RMDS,$BDADR
10568  040750  104226                           ERROR   226                     ;CANT RESET ATA WITH GO
10569
10570  040752                           10$:                                    ;END OF TEST
10571
10572                                   ;;*****************************************************************
10573                                   ;*TEST 104      UNIT READY ATA TEST
10574
10575                                   ;;*****************************************************************
10576  040752  000004                   ↑ST104: SCOPE
10577  040754  012737  000104  001226           MOV     #104,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10578
10579  040762  000240                           NOP
10580  040764  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
10581  040772  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10582  041000  012737  041014  001122           MOV     #T104,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10583  041006  012737  041014  001124           MOV     #T104,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10584  041014                           T104:
10585  041014  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
10586  041020  013700  001276                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
```

```
10587  041024  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10588                          ;SET DIAGNOSTIC MODE AND CLEAR ATA
10589  041030  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10590  041036  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10591
10592  041042  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10593
10594  041050  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
10595
10596  041056  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
10597
10598  041064  012760  177777  000016          MOV     #-1,RMAS(R0)    ;LOAD RMAS
10599                          ;SET UNIT READY AND VERIFY ATA IS SET
10600
10601  041072  012760  001001  000024          MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
10602
10603  041100  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10604  041106  042737  077777  001142          BIC     #^CATA,$BDDAT
10605  041114  001011                          BNE     10$                     ;BRANCH IF ATA IS SET
10606  041116  010037  001136                  MOV     R0,$BDADR
10607  041122  062737  000012  001136          ADD     #RMDS,$BDADR
10608  041130  012737  100000  001140          MOV     #ATA,$GDDAT
10609  041136  104227                          ERROR   227                     ;ATA NOT SET BY 01 READY
10610  041140                          10$:
10611                          ;CLEAR ATA, RESET UNIT READY, AND VERIFY ATA IS SET
10612
10613  041140  012760  177777  000016          MOV     #-1,RMAS(R0)    ;LOAD RMAS
10614
10615  041146  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10616
10617  041154  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10618  041162  042737  077777  001142          BIC     #^CATA,$BDDAT
10619  041170  001011                          BNE     20$
10620  041172  010037  001136                  MOV     R0,$BDADR
10621  041176  062737  000012  001136          ADD     #RMDS,$BDADR
10622  041204  012737  100000  001140          MOV     #ATA,$GDDAT
10623  041212  104230                          ERROR   230                     ;ATA NOT SET BY 10 READY
10624
10625  041214                          20$:                                    ;END OF TEST
10626
10627                          ;;*****************************************************************
10628                          ;*TEST 105      ERROR ATA TEST
10629
10630                          ;;*****************************************************************
10631  041214  000004          TST105: SCOPE
10632  041216  012737  000105  001226          MOV     #105,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10633
10634  041224  000240                          NOP
10635  041226  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10636  041234  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10637  041242  012737  041256  001122          MOV     #T105,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10638  041250  012737  041256  001124          MOV     #T105,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10639  041256                          T105:
10640  041256  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10641  041262  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10642  041266  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
```

```
10643                                   ;CLEAR THE DEVICE AND RESET ATTENTION
10644  041272  012760  000040  000010           MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10645  041300  111160  000010                   MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10646
10647  041304  012760  000001  r 3024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10648
10649  041312  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1
10650
10651  041320  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2
10652
10653  041326  012760  177777  000016           MOV     #-1,RMAS(R0)    ;LOAD RMAS
10654                                   ;WRITE ONES IN ERROR REGISTER 1 AND VERIFY ATA IS SET
10655
10656  041334  012760  177777  000014           MOV     #-1,RMER1(R0)   ;LOAD RMER1
10657
10658  041342  016037  000012  001142           MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10659  041350  042737  077777  001142           BIC     #^CATA,$BDDAT
10660  041356  001011                           BNE     10$
10661  041360  012737  100000  001140           MOV     #ATA,$GDDAT
10662  041366  010037  001136                   MOV     R0,$BDADR
10663  041372  062737  000012  001136           ADD     #RMDS,$BDADR
10664  041400  104231                           ERROR   231                     ;ATA NOT SET BY COMP ERROR
10665
10666  041402                           10$:                                   ;END OF TEST
10667
10668                                   ;;*****************************************************************
10669                                   ;*TEST 106       REGISTER TRANSFER ATA TEST
10670
10671                                   ;;*****************************************************************
10672  041402  000004                   T$T106: SCOPE
10673  041404  012737  000106  001226           MOV     #106,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10674  041412  000240                           NOP
10675  041414  012737  000024  001120           MOV     #20.,$ICNT      ;20 ITERATIONS
10676  041422  112737  000001  001131           MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10677  041430  012737  041444  001122           MOV     #T106,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10678  041436  012737  041444  001124           MOV     #T106,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10679  041444                           T106:
10680  041444  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
10681  041450  013700  001276                   MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10682  041454  013701  001456                   MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10683  041460  012702  041614                   MOV     #100$,R2                ;INITIALIZE TABLE ADDRESS
10684                                   ;FORCE COMPOSITE ERROR AND RESET ATA
10685  041464                           5$:
10686  041464  012760  000040  000010           MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10687  041472  111160  000010                   MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10688
10689  041476  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
10690
10691  041504  012760  177777  000014           MOV     #-1,RMER1(R0)   ;LOAD RMER1
10692
10693  041512  012760  177777  000016           MOV     #-1,RMAS(R0)    ;LOAD RMAS
10694                                   ;WRITE THE TEST REGISTER AND VERIFY ATA
10695  041520  011203                           MOV     (R2),R3                 ;GENERATE REGISTER ADDRESS
10696  041522  060003                           ADD     R0,R3
10697  041524  005013                           CLR     (R3)
10698
```

```
10699  041526  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10700  041534  042737  077777  001142          BIC     #^CATA,$BDDAT
10701  041542  016237  000002  001140          MOV     2(R2),$GDDAT
10702  041550  023737  001140  001142          CMP     $GDDAT,$BDDAT
10703  041556  001410                          BEQ     10$                     ;BRANCH IF ATA IS OK
10704  041560  010337  001174                  MOV     R3,$TMP0
10705  041564  010037  001136                  MOV     R0,$BDADR
10706  041570  062737  000012  001136          ADD     #RMDS,$BDADR
10707  041576  104232                          ERROR   232                     ;ATA INCORRECT FOR REG WRITTEN
10708  041600                          10$:
10709                                  ;MOVE TABLE POINTER - EXIT IF DONE
10710  041600  062702  000004                  ADD     #4,R2
10711  041604  005712                          TST     (R2)
10712  041606  100401                          BMI     20$                     ;EXIT IF ENTRY MINUS
10713  041610  000725                          BR      5$
10714  041612  000410                  20$:    BR      200$                    ;JUMP OVER TABLE
10715
10716  041614                          100$:
10717                                  ;TABLE OF REGISTER ADDRESSES AND ATA BITS
10718  041614  000016                          .WORD   RMAS
10719  041616  000000                          .WORD
10720
10721  041620  000006                          .WORD   RMDA
10722  041622  100000                          .WORD   ATA
10723
10724  041624  000000                          .WORD   RMCS1
10725  041626  000000                          .WORD
10726
10727  041630  177777                          .WORD   -1                      ;END OF TABLE
10728  041632  000000                          .WORD
10729
10730  041634                          200$:                                   ;END OF TEST
10731
10732
10733                                  ;;**********************************************************************
10734                                  ;*TEST 107      P SET ATA TEST
10735
10736                                  ;;**********************************************************************
10737  041634  000004                  T$T107: SCOPE
10738  041636  012737  000107  001226          MOV     #107,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10739
10740  041644  000240                          NOP
10741  041646  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10742  041654  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10743  041662  012737  041676  001122          MOV     #T107,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10744  041670  012737  041676  001124          MOV     #T107,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10745  041676                          T107:
10746  041676  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
10747  041702  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10748  041706  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10749  041712  012702  042054                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
10750
10751  041716                          10$:
10752                                  ;SET VOLUME VALID USING SUBROUTINE
10753  041716  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
10754  041722  003402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
```

```
10755  041724  104000                          ERROR                   ;RETURN HERE IF ERROR
10756  041726  000451                          BR      50$
10757  041730                          20$:
10758                                  ;EXECUTE THE COMMAND FROM THE TABLE
10759
10760  041730  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10761  041736  011203                          MOV     (R2),R3         ;GET FUNCTION CODE
10762  041740  052703  000001                  BIS     #GO,R3
10763
10764  041744  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
10765  041750  012703  000003                  MOV     #3,R3
10766  041754                          30$:
10767
10768  041754  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
10769
10770  041762  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10771  041770  005303                          DEC     R3
10772  041772  001370                          BNE     30$
10773                                  ;VERIFY THAT ATA IS SET
10774
10775  041774  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
10776  042002  042737  077777  001142          BIC     #^CATA,$BDDAT
10777  042010  001013                          BNE     40$
10778  042012  010037  001136                  MOV     R0,$BDADR
10779  042016  062737  000012  001136          ADD     #RMDS,$BDADR
10780  042024  012737  100000  001140          MOV     #ATA,$GODAT
10781  042032  011237  001174                  MOV     (R2),$TMP0
10782  042036  104233                          ERROR   233             ;ATA NOT SET BY SEQUENCER
10783  042040                          40$:
10784                                  ;ADVANCE TABLE POINTER-EXIT IF DONE
10785  042040  062702  000002                  ADD     #2,R2
10786  042044  005712                          TST     (R2)
10787  042046  100401                          BMI     50$
10788  042050  000722                          BR      10$
10789  042052  000403                  50$:    BR      200$            ;JUMP OVER TABLE
10790
10791  042054                          100$:
10792                                  ;TABLE OF FUNCTION CODES
10793
10794  042054  000014                          .WORD   OFFSET
10795
10796  042056  000016                          .WORD   RTC
10797
10798  042060  177777                          .WORD   -1              ;END OF TABLE
10799  042062                          200$:
10800
10801                                  ;;*****************************************************************
10802                                  ;*TEST 110       SET WLE TEST
10803
10804                                  ;;*****************************************************************
10805  042062  000004                  TST110: SCOPE
10806  042064  012737  000110  001226          MOV     #110,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10807
10808  042072  000240                          NOP
10809  042074  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
10810  042102  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
```

```
10811  042110  012737  042124  001122          MOV     #T110,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10812  042116  012737  042124  001124          MOV     #T110,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10813  042124                                T110:
10814  042124  012706  001100                  MOV     #STACK,SP     ;LOAD THE STACK POINTER
10815  042130  013700  001276                  MOV     $BASE,R0      ;R0 = UNIBUS ADDRESS OF UUT
10816  042134  013701  001456                  MOV     TSTQUE,R1     ;R1 = POINTER TO DEVICE
10817  042140  012702  042334                  MOV     #10O$,R2      ;INITIALIZE TABLE POINTER
10818  042144                                10$:
10819                                         ;SET VOLUME VALID USING SUBROUTINE
10820  042144  004737  060526                  JSR     PC,SETVV      ;GO SET VOLUME VALID
10821  042150  000402                          BR      20$           ;BRANCH TO 20$ IF NO ERROR
10822  042152  104000                          ERROR                 ;RETURN HERE IF ERROR
10823  042154  000466                          BR      50$
10824  042156                                20$:
10825                                         ;ENABLE DEBUG CLOCK AND SET WRITE PROTECT ACCORDING TO TABLE
10826  042156  016203  000002                  MOV     2(R2),R3      ;GET WRITE PROTECT FROM TABLE
10827  042162  052703  041001                  BIS     #DMD!MUR!DBEN,R3      ;SET OTHER MAINT BITS
10828
10829  042166  010360  000024                  MOV     R3,RMMR1(R0)  ;LOAD RMMR1
10830                                         ;LOAD AND EXECUTE THE COMMAND FROM THE TABLE
10831  042172  011204                          MOV     (R2),R4       ;GET FUNCTION CODE FROM TABLE
10832  042174  052704  000001                  BIS     #GO,R4        ;SET GO
10833
10834  042200  010460  000000                  MOV     R4,RMCS1(R0)  ;LOAD RMCS1
10835  042204  012705  000002                  MOV     #2,R5         ;R5=CLOCK COUNT
10836  042210  052703  100000          30$:    BIS     #DBCK,R3      ;SET CLOCK
10837
10838  042214  010360  000024                  MOV     R3,RMMR1(R0)  ;LOAD RMMR1
10839  042220  042703  100000                  BIC     #DBCK,R3      ;RESET CLOCK
10840
10841  042224  010360  000024                  MOV     R3,RMMR1(R0)  ;LOAD RMMR1
10842  042230  005305                          DEC     R5
10843  042232  001366                          BNE     30$           ;ISSUE 2 CLOCKS
10844                                         ;VERIFY THAT WRITE LOCK ERROR IS ACCORDING TO TABLE
10845
10846  042234  016037  000014  001142          MOV     RMER1(R0),$BDDAT      ;STORE RMER1 AT $BDDAT
10847  042242  042737  173777  001142          BIC     #^tCWLE,$BDDAT
10848  042250  026237  000004  001142          CMP     4(R2),$BDDAT
10849  042256  001417                          BEQ     40$           ;BRANCH IF WLE IS OK
10850  042260  016237  000004  001140          MOV     4(R2),$GDDAT  ;SAVE DATA FOR ERROR MSG
10851  042266  010037  001136                  MOV     R0,$BDADR
10852  042272  062737  000014  001136          ADD     #RMER1,$BDADR
```

```
10853  042300  011237  001174                 MOV     (R2),$TMP0      ;$TMP0=FUNCTION CODE
10854  042304  016237  000002  001176         MOV     2(R2),$TMP1     ;$TMP1=WRITE PROTECT
10855  042312  104234                         ERROR   234             ;WLE INCORRECT
10856  042314  000406                         BR      50$
10857  042316                             40$:
10858                                     ;ADVANCE TABLE POINTER TO NEXT FUNCTION CODE-EXIT IF DONE
10859  042316  062702  000006                 ADD     #6,R2
10860  042322  005762  000002                 TST     2(R2)
10861  042326  100401                         BMI     50$
10862  042330  000705                         BR      10$             ;REPEAT TEST
10863
10864  042332  000416                 50$:    BR      200$            ;JUMP OVER TABLE
10865
10866  042334                             100$:
10867
10868                                     ;TABLE OF FUNCTION CODES, WRITE PROTECT AND WRITE LOCK ERRORS
10869
10870  042334  000060                         .WORD   WD              ;WRITE DATA COMMAND
10871  042336  000010                         .WORD   MWP             ;WRITE PROTECT ON
10872  042340  004000                         .WORD   WLE             ;WRITE LOCK ERROR ONE
10873
10874  042342  000060                         .WORD   WD              ;WRITE DATA COMMAND
10875  042344  000000                         .WORD                   ;MWP OFF
10876  042346  000000                         .WORD                   ;WLE OFF
10877
10878  042350  000070                         .WORD   RD              ;READ DATA COMMAND
10879  042352  000010                         .WORD   MWP             ;MWP ON
10880  042354  000000                         .WORD                   ;WLE OFF
10881
10882  042356  000020                         .WORD   RIP             ;READ IN PRESET COMMAND
10883  042360  000010                         .WORD   MWP             ;MWP ON
10884  042362  000000                         .WORD                   ;WLE OFF
10885
10886  042364  000000                         .WORD                   ;END OF TABLE
10887  042366  177777                         .WORD   -1
10888
10889  042370                             200$:                       ;END OF TEST
10890
10891                                     ;;*****************************************************************
10892                                     ;*TEST 111      EXCEPTION TEST
10893
10894                                     ;;*****************************************************************
10895  042370  000004                     TST111: SCOPE
10896  042372  012737  000111  001226             MOV     #111,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
10897
10898                                     ;WITH OCCUPIED SET, EACH OF THE FOLLOWING ERRORS SHOULD CAUSE AN
10899                                     ;EXCEPTION:
10900                                     ;
10901                                     ;       PAR, IF RUN AND GO IS SET
10902                                     ;       RMR, IF RUN AND GO IS SET
10903                                     ;       ILR, IF RUN AND GO IS SET
10904                                     ;       DCK
10905                                     ;       HCE
10906                                     ;       HCRC
10907                                     ;       FER
10908                                     ;       OPI
```

```
10909
10910  042400  000240                      NOP
10911  042402  012737  000024  001120      MOV     #20.,$ICNT      ;20 ITERATIONS
10912  042410  112737  000001  001131      MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
10913  042416  012737  042432  001122      MOV     #T111,$LPADR  ;LOAD LOOP ON TEST ADDRESS
10914  042424  012737  042432  001124      MOV     #T111,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
10915  042432                          T111:
10916  042432  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
10917  042436  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
10918  042442  013701  001456              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
10919  042446  012702  042726              MOV     #100$,R2        ;INITIALIZE TABLE POINTER
10920  042452                          10$:
10921                                   ;INITIALIZE AND VERIFY THAT EXCEPTION IS RESET
10922  042452  012760  000040  000010      MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
10923  042460  111160  000010              MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
10924
10925  042464  016037  000024  001142      MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
10926  042472  042737  167777  001142      BIC     #^CREX,$BDDAT
10927  042500  001410                      BEQ     20$             ;BRANCH IF EXCEPTION IS 0
10928  042502  010037  001136              MOV     R0,$BDADR
10929  042506  062737  000024  001136      ADD     #R'R1,$BDADR
10930  042514  005037  001140              CLR     $GDDAT
10931  042520  104235                      ERROR   235             ;CANT CLEAR EXCEPTION
10932  042522                          20$:
10933                                   ;SET VOLUME VALID USING SUBROUTINE
10934  042522  004737  060526              JSR     PC,SETVV        ;GO SET VOLUME VALID
10935  042526  000402                      BR      30$             ;BRANCH TO 30$ IF NO ERROR
10936  042530  104000                      ERROR                   ;RETURN HERE IF ERROR
10937  042532  000474                      BR      60$
10938  042534                          30$:
10939                                   ;EXECUTE A WRITE DATA COMMAND TO SET OCCUPIED, RUN AND GO
10940
10941  042534  012760  000000  000006      MOV     #0,RMDA(R0)     ;LOAD RMDA
10942
10943  042542  012760  000000  000034      MOV     #0,RMDC(R0)     ;LOAD RMDC
10944
10945  042550  012760  041001  000024      MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10946
10947  042556  012760  000061  000000      MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
10948  042564  012703  000002              MOV     #2,R3
10949  042570                          40$:
10950
10951  042570  012760  141001  000024      MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
10952
10953  042576  012760  041001  000024      MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
10954  042604  005303                      DEC     R3
10955  042606  001370                      BNE     40$             ;ISSUE 2 CLOCKS
10956                                   ;LOAD ERROR REGISTER WITH ENTRY FROM TABLE AND VERIFY THAT EXCEPTION IS SET
10957
10958  042610  011260  000014              MOV     (R2),RMER1(R0)  ;LOAD RMER1
10959  042614  012737  000200  001524      MOV     #200,WATCH      ;SET WATCHDOG TIMER VALUE
10960  042622  004777  136700              JSR     PC,@CLOCK       ;START THE CLOCK
10961  042626                          45$:
10962
10963  042626  016037  000024  001142      MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
10964  042634  042737  167777  001142      BIC     #^CREX,$BDDAT
```

```
10965  042642  001021                          BNE     50$
10966  042644  005737  001524                  TST     WATCH           ;HAS CLOCK EXPIRED ??
10967  042650  001366                          BNE     45$             ;NO - TAKE ANOTHER SAMPLE
10968  042652  004777  136652                  JSR     PC,@STOP        ;STOP THE CLOCK
10969  042656  012737  010000  001140          MOV     #REX,$GDDAT
10970  042664  010037  001136                  MOV     R0,$BDADR
10971  042670  062737  000024  001136          ADD     #RMR1,$BDADR
10972  042676  011237  001174                  MOV     (R2),$TMP0
10973  042702  104236                          ERROR   236             ;CANT SET EXCEPTION
10974  042704  000407                          BR      60$
10975  042706                          50$:
10976                                  ;ADVANCE TABLE POINTER-EXIT IF DONE
10977  042706  004777  136616                  JSR     PC,@STOP        ;STOP THE CLOCK
10978  042712  062702  000002                  ADD     #2,R2
10979  042716  005712                          TST     (R2)
10980  042720  001401                          BEQ     60$
10981  042722  000653                          BR      10$
10982  042724  000402                  60$:    BR      200$            ;JUMP OVER TABLE
10983
10984  042726                          100$:
10985                                  ;PRESENTLY, THE TABLE HAS ONLY ONE ENTRY, THE TEST USING RMR.  THE
10986                                  ;TABLE SHOULD BE EXPANDED TO TEST ALL THE CONDITIONS LISTED ABOVE IF
10987                                  ;A HARDWARE CHANGE IS MADE SUCH THAT IT IS POSSIBLE TO WRITE ERROR
10988                                  ;REGISTER 1 WITH GO SET.
10989
10990  042726  000004                          .WORD   RMR             ;RMR IS CAUSED BY HARDWARE
10991
10992  042730  000000                          .WORD                   ;END OF TABLE
10993  042732                          200$:                           ;END OF TEST
10994
10995                                  ;;*****************************************************************
10996                                  ;*TEST 112       SET OPI TEST
10997
10998                                  ;;*****************************************************************
10999  042732  000004                  T112:   SCOPE
11000  042734  012737  000112  001226          MOV     #112,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
11001
11002  042742  000240                          NOP
11003  042744  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
11004  042752  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
11005  042760  012737  042774  001122          MOV     #T112,$LPADR    ;LOAD LOOP ON TEST ADDRESS
11006  042766  012737  042774  001124          MOV     #T112,$LPERR    ;LOAD LOOP ON ERROR ADDRESS
11007  042774                          T112:
11008  042774  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
11009  043000  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
11010  043004  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
11011  043010  005002                          CLR     R2              ;INITIALIZE FUNCTION CODE
11012  043012  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MSG
11013  043016  062737  000014  001136          ADD     #RMER1,$BDADR
11014  043024                          10$:
11015                                  ;CLEAR AND VERIFY OPI IS RESET
11016  043024  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
11017  043032  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
11018
11019  043036  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11020  043044  042737  157777  001142          BIC     #^COPI,$BDDAT
```

```
11021  043052  001403                        BEQ     20$             ;BRANCH IF OPI IS ZERO
11022  043054  005037  001140                CLR     $GDDAT
11023  043060  104237                        ERROR   237             ;CANT CLEAR OPI
11024  043062                          20$:
11025                                  ;EXECUTE THE FUNCTION CODE AND VERIFY OPI IS SET
11026
11027  043062  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
11028
11029  043070  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11030  043076  010203                        MOV     R2,R3           ;SET GO
11031  043100  052703  000001                BIS     #GO,R3
11032
11033  043104  010360  000000                MOV     R3,RMCS1(R0)    ;LOAD RMCS1
11034  043110  012704  000004                MOV     #4,R4           ;R4=CLOCK COUNT
11035  043114                          30$:
11036
11037  043114  012760  140001  000024        MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
11038
11039  043122  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11040  043130  005304                        DEC     R4
11041  043132  001370                        BNE     30$             ;ISSUE 4 CLOCKS
11042
11043  043134  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11044  043142  042737  157777  001142        BIC     #^COPI,$BDDAT
11045  043150  001007                        BNE     40$
11046  043152  012737  020000  001140        MOV     #OPI,$GDDAT
11047  043160  010237  001174                MOV     R2,$TMP0
11048  043164  104240                        ERROR   240             ;CANT SET OPI
11049  043166  000406                        BR      50$
11050  043170                          40$:
11051                                  ;ADVANCE FUNCTION CODE-EXIT IF DONE
11052  043170  062702  000002                ADD     #2,R2
11053  043174  022702  000076                CMP     #ILF76,R2
11054  043200  103401                        BLO     50$
11055  043202  000710                        BR      10$             ;REPEAT FOR NEXT CODE
11056
11057  043204                          50$:                            ;END OF TEST
11058
11059                                  ;;*****************************************************************
11060                                  ;*TEST 113       RECALIBRATE TEST
11061
11062                                  ;;*****************************************************************
11063  043204  000004                  ^ST113: SCOPE
11064  043206  012737  000113  001226        MOV     #113,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
11065
11066  043214  000240                        NOP
11067  043216  012737  000024  001120        MOV     #20.,$ICNT      ;20 ITERATIONS
11068  043224  112737  000001  001131        MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
11069  043232  012737  043246  001122        MOV     #T113,$LPADR  ;LOAD LOOP ON TEST ADDRESS
11070  043240  012737  043246  001124        MOV     #T113,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
11071  043246                          T113:
11072  043246  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
11073  043252  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
11074  043256  013701  001456                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
11075
11076                                  ;;*****************************************************************
```

```
11077                                   ;VERIFY THAT OPI SETS IF UNIT READY DROPS AFTER DECODE
11078
11079                                   ;SET VOLUME VALID USING SUBROUTINE
11080  043262  004737  060526                 JSR     PC,SETVV        ;GO SET VOLUME VALID
11081  043266  000403                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
11082  043270  104000                          ERROR                   ;RETURN HERE IF ERROR
11083  043272  000137  044570                 JMP     330$
11084  043276                          10$:
11085                                   ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11086
11087  043276  012760  041001  000024         MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11088
11089  043304  012760  000000  000014         MOV     #0,RMER1(R0)    ;LOAD RMER1
11090
11091  043312  012760  000000  000042         MOV     #0,RMER2(R0)    ;LOAD RMER2
11092
11093  043320  012760  000007  000000         MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11094                                   ;       STEP COMMAND SEQUENCER TO RECAL COM (2 CLOCKS)
11095  043326  012702  000002                 MOV     #2,R2
11096  043332                          20$:
11097
11098  043332  012760  141001  000024         MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11099
11100  043340  012760  041001  000024         MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11101  043346  005302                          DEC     R2
11102  043350  001370                          BNE     20$
11103                                   ;       DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11104
11105  043352  012760  040001  000024         MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11106  043360  012702  000002                 MOV     #2,R2
11107  043364                          30$:
11108
11109  043364  012760  140001  000024         MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
11110
11111  043372  012760  040001  000024         MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11112  043400  005302                          DEC     R2
11113  043402  001370                          BNE     30$
11114                                   ;       VERIFY THAT OPI IS SET
11115
11116  043404  016037  000014  001142         MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11117  043412  042737  157777  001142         BIC     #^C<OPI>,$BDDAT
11118  043420  001011                          BNE     40$
11119  043422  012737  020000  001140         MOV     #OPI,$GDDAT
11120  043430  010037  001136                 MOV     R0,$BDADR
11121  043434  062737  000014  001136         ADD     #RMER1,$BDADR
11122  043442  104241                          ERROR   241             ;OPI NOT SET DURING RECAL
11123
11124  043444  012737  043452  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
11125
11126                                   ;;*****************************************************************
11127                                   ;VERIFY THAT RECALIBRATE ABORTS DURING EXECUTION
11128
11129  043452                          50$:
11130                                   ;SET VOLUME VALID USING SUBROUTINE
11131  043452  004737  060526                 JSR     PC,SETVV        ;GO SET VOLUME VALID
11132  043456  000403                          BR      60$             ;BRANCH TO 60$ IF NO ERROR
```

```
11133  043460  104000                          ERROR                   ;RETURN HERE IF ERROR
11134  043462  000137  044570                  JMP     330$
11135
11136  043466                          60$:
11137                                  ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11138
11139  043466  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11140
11141  043474  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11142
11143  043502  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11144
11145  043510  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11146                                  ;       STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11147  043516  012702  000003                  MOV     #3,R2
11148  043522                          70$:
11149
11150  043522  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11151
11152  043530  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11153  043536  005302                          DEC     R2
11154  043540  001370                          BNE     70$
11155                                  ;       SET DRIVE FAULT TO CAUSE ABORT CONDITION
11156
11157  043542  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
11158                                  ;       STEP 2 CLOCKS AND VERIFY GO IS RESET
11159  043550  012702  000002                  MOV     #2,R2
11160  043554                          80$:
11161
11162  043554  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
11163
11164  043562  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
11165  043570  005302                          DEC     R2
11166  043572  001370                          BNE     80$
11167
11168  043574  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11169  043602  042737  177776  001142          BIC     #^CGO,$BDDAT
11170  043610  001405                          BEQ     90$
11171  043612  010037  001136                  MOV     R0,$BDADR
11172  043616  005037  001140                  CLR     $GODAT
11173  043622  104242                          ERROR   242             ;GO NOT RESET DUE TO ABORT
11174
11175  043624  012737  043632  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11176
11177                                  ;;*****************************************************************
11178                                  ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11179
11180  043632                          100$:
11181                                  ;SET VOLUME VALID USING SUBROUTINE
11182  043632  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11183  043636  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
11184  043640  104000                          ERROR                   ;RETURN HERE IF ERROR
11185  043642  000137  044570                  JMP     330$
11186  043646                          110$:
11187                                  ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11188
```

```
11189  043646  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11190
11191  043654  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11192
11193  043662  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11194
11195  043670  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11196                                  ;       STEP THE COMMAND SEQUENCER
11197  043676  012702  000017                  MOV     #15.,R2
11198  043702                          120$:
11199
11200  043702  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11201
11202  043710  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11203  043716  005302                          DEC     R2
11204  043720  001370                          BNE     120$
11205                                  ;       VERIFY THAT OPI IS SET
11206
11207  043722  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11208  043730  042737  157777  001142          BIC     #^COPI,$BDDAT
11209  043736  001011                          BNE     130$
11210  043740  010037  001136                  MOV     R0,$BDADR
11211  043744  062737  000014  001136          ADD     #RMER1,$BDADR
11212  043752  012737  020000  001140          MOV     #OPI,$GDDAT
11213  043760  104243                          ERROR   243             ;OPI NOT SET DUE TO ON LATCH
11214
11215  043762  012737  043770  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11216
11217                                  ;;**************************************************************
11218                                  ;VERIFY ATA SETS IF DRIVE COMPLETES RECALIBRATE (ON CYLINDER SETS)
11219
11220  043770                          150$:
11221                                  ;SET VOLUME VALID USING SUBROUTINE
11222  043770  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11223  043774  000403                          BR      160$            ;BRANCH TO 160$ IF NO ERROR
11224  043776  104000                          ERROR                   ;RETURN HERE IF ERROR
11225  044000  000137  044570                  JMP     330$
11226  044004                          160$:
11227                                  ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11228
11229  044004  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11230
11231  044012  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11232
11233  044020  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11234
11235  044026  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11236                                  ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11237  044034  012702  000015                  MOV     #13.,R2
11238  044040                          170$:
11239
11240  044040  012760  141401  000024          MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
11241
11242  044046  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11243  044054  005302                          DEC     R2
11244  044056  001370                          BNE     170$
```

```
11245                                          ;       DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
11246
11247  044060  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11248
11249  044066  012760  041401  000024                  MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11250                                          ;       STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
11251  044074  012702  000003                          MOV     #3,R2
11252  044100                                  180$:
11253
11254  044100  012760  141401  000024                  MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
11255
11256  044106  012760  041401  000024                  MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11257  044114  005302                                  DEC     R2
11258  044116  001370                                  BNE     180$
11259                                          ;       VERIFY ATA IS SET
11260
11261  044120  016037  000012  001142                  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
11262  044126  042737  077777  001142                  BIC     #^tCATA,$BDDAT
11263  044134  001011                                  BNE     190$
11264  044136  012737  100000  001140                  MOV     #ATA,$GDDAT
11265  044144  010037  001136                          MOV     R0,$BDADR
11266  044150  062737  000012  001136                  ADD     #RMDS,$BDADR
11267  044156  104244                                  ERROR   244             ;ATA NOT SET BY RECAL
11268
11269  044160  012737  044166  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11270
11271                                          ;;*****************************************************************
11272                                          ;VERIFY THAT RECALIBRATE ABORTS AFTER EXECUTION DURING WAIT LOOP
11273
11274  044166                                  200$:
11275                                          ;SET VOLUME VALID USING SUBROUTINE
11276  044166  004737  060526                          JSR     PC,SETVV        ;GO SET VOLUME VALID
11277  044172  000402                                  BR      210$            ;BRANCH TO 210$ IF NO ERROR
11278  044174  104000                                  ERROR                   ;RETURN HERE IF ERROR
11279  044176  000574                                  BR      330$
11280  044200                                  210$:
11281                                          ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11282
11283  044200  012760  041401  000024                  MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11284
11285  044206  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
11286
11287  044214  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
11288
11289  044222  012760  000007  000000                  MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11290                                          ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11291  044230  012702  000015                          MOV     #13.,R2
11292  044234                                  220$:
11293
11294  044234  012760  141401  000024                  MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
11295
11296  044242  012760  041401  000024                  MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11297  044250  005302                                  DEC     R2
11298  044252  001370                                  BNE     220$
11299                                          ;       DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
11300
```

```
11301  044254  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11302                                  ;       STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
11303  044262  012702  000007                  MOV     #7,R2
11304  044266                          230$:
11305
11306  044266  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11307
11308  044274  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11309  044302  005302                          DEC     R2
11310  044304  001370                          BNE     230$
11311                                  ;       VERIFY THAT GO IS STILL SET
11312
11313  044306  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11314  044314  042737  177776  001142          BIC     #^CGO,$BDDAT
11315  044322  001006                          BNE     240$
11316  044324  012737  000001  001140          MOV     #GO,$GDDAT
11317  044332  010037  001136                  MOV     R0,$BDADR
11318  044336  104245                          ERROR   245             ;GO RESET EARLY DURING RECAL
11319                                  ;       SET SEEK INCOMPLETE ERROR
11320  044340                          240$:
11321
11322  044340  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
11323                                  ;       STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
11324  044346  012702  000003                  MOV     #3,R2
11325  044352                          250$:
11326
11327  044352  012760  141201  000024          MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
11328
11329  044360  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
11330  044366  005302                          DEC     R2
11331  044370  001370                          BNE     250$
11332
11333  044372  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11334  044400  042737  177776  001142          BIC     #^CGO,$BDDAT
11335  044406  001405                          BEQ     260$
11336  044410  010037  001136                  MOV     R0,$BDADR
11337  044414  005037  001140                  CLR     $GDDAT
11338  044420  104246                          ERROR   246             ;GO NOT RESET DUR TO WAIT ABORT
11339
11340  044422  012737  044430  001124  260$:   MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11341
11342                                  ;;**************************************************************
11343                                  ;VERIFY THE TAG BUS DURING RECALIBRATE
11344
11345  044430                          300$:
11346                                  ;SET VOLUME VALID USING SUBROUTINE
11347  044430  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11348  044434  000402                          BR      310$            ;BRANCH TO 310$ IF NO ERROR
11349  044436  104000                          ERROR                   ;RETURN HERE IF ERROR
11350  044440  000453                          BR      330$
11351  044442                          310$:
11352                                  ;       ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
11353
11354  044442  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11355
11356  044450  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
```

```
11357
11358  044456  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11359
11360  044464  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
11361  044472  012702  044572                  MOV     #400$,R2        ;INITIALIZE TABLE POINTER
11362                                  ;       VERIFY TAG BUS ACCORDING TO TABLE
11363  044476                          315$:
11364
11365  044476  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
11366  044504  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
11367  044512  021237  001142                  CMP     (R2),$BDDAT
11368  044516  001411                          BEQ     320$
11369  044520  011237  001140                  MOV     (R2),$GDDAT
11370  044524  010037  001136                  MOV     R0,$BDADR
11371  044530  062737  000040  001136          ADD     #RMMR2,$BDADR
11372  044536  104247                          ERROR   247             ;INCORRECT TAG BUS DURING RECAL
11373  044540  000413                          BR      330$
11374  044542                          320$:
11375                                  ;       ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
11376  044542  062702  000002                  ADD     #2,R2
11377  044546  005712                          TST     (R2)
11378  044550  100407                          BMI     330$            ;EXIT IF ENTRY NEGATIVE
11379                                  ;       STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
11380
11381  044552  012760  151001  000024          MOV     #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0)        ;LOAD RMMR1
11382
11383  044560  012760  041401  000024          MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
11384  044566  000743                          BR      315$
11385  044570  000416                  330$:   BR      500$            ;JUMP OVER TABLE
11386
11387  044572                          400$:
11388
11389                                  ;TABLE OF TAG BUS CONTROL AND BIT VALUES
11390
11391  044572  001777                          .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
11392  044574  001777                          .WORD   1777
11393  044576  006100                          .WORD   CC!CH!BBO6      ;CONTROL BITS ENABLED, BIT 6 ON
11394  044600  006100                          .WORD   CC!CH!BBO6
11395  044602  026100                          .WORD   TAG!CC!CH!BBO6  ;TAG COMES ON
11396  044604  026100                          .WORD   TAG!CC!CH!BBO6
11397  044606  026100                          .WORD   TAG!CC!CH!BBO6
11398  044610  026100                          .WORD   TAG!CC!CH!BBO6
11399  044612  026100                          .WORD   TAG!CC!CH!BBC
11400  044614  026100                          .WORD   TAG!CC!CH!BBO6
11401  044616  006100                          .WORD   CC!CH!BBO6      ;TAG GOES OFF
11402  044620  006100                          .WORD   CC!CH!BBO6
11403  044622  001777                          .WORD   1777            ;CONTROL BITS DISABLED
11404
11405  044624  177777                          .WORD   -1              ;END OF TABLE
11406
11407  044626                          500$:                           ;END OF TEST
11408
11409                                  ;;*****************************************************************
11410                                  ;*TEST 114       SEEK TEST
11411
11412                                  ;;*****************************************************************
```

```
11413  044626  000004                          TST114: SCOPE
11414  044630  012737  000114  001226                  MOV     #114,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
11415
11416  044636  000240                                  NOP
11417  044640  012737  000024  001120                  MOV     #20.,$ICNT      ;20 ITERATIONS
11418  044646  112737  000001  001131                  MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
11419  044654  012737  044670  001122                  MOV     #T114,$LPADR  ;LOAD LOOP ON TEST ADDRESS
11420  044662  012737  044670  001124                  MOV     #T114,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
11421  044670                                  T114:
11422  044670  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
11423  044674  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
11424  044700  013701  001456                          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
11425
11426                                          ;;**************************************************************
11427                                          ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
11428
11429                                          ;SET VOLUME VALID USING SUBROUTINE
11430  044704  004737  060526                          JSR     PC,SETVV        ;GO SET VOLUME VALID
11431  044710  000403                                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
11432  044712  104000                                  ERROR                   ;RETURN HERE IF ERROR
11433  044714  000137  046334                          JMP     330$
11434  044720                                  10$:
11435                                          ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11436                                          ;       ADDRESS,  AND LOAD SEEK COMMAND
11437
11438  044720  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11439
11440  044726  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
11441
11442  044734  012760  000000  000034                  MOV     #0,RMDC(R0)     ;LOAD RMDC
11443
11444  044742  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
11445
11446  044750  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
11447
11448  044756  012760  000005  000000                  MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11449                                          ;       STEP COMMAND SEQUENCER TO SEEK COM (2 CLOCKS)
11450  044764  012702  000002                          MOV     #2,R2
11451  044770                                  20$:
11452
11453  044770  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11454
11455  044776  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11456  045004  005302                                  DEC     R2
11457  045006  001370                                  BNE     20$
11458                                          ;       DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11459
11460  045010  012760  040001  000024                  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11461  045016  012702  000002                          MOV     #2,R2
11462  045022                                  30$:
11463
11464  045022  012760  140001  000024                  MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
11465
11466  045030  012760  040001  000024                  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11467  045036  005302                                  DEC     R2
11468  045040  001370                                  BNE     30$
```

```
11469                                   ;       VERIFY THAT OPI IS SET
11470
11471  045042  016037  000014  001142           MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11472  045050  042737  157777  001142           BIC     #^COPI,$BDDAT
11473  045056  001011                           BNE     40$
11474  045060  012737  020000  001140           MOV     #OPI,$GDDAT
11475  045066  010037  001136                   MOV     R0,$BDADR
11476  045072  062737  000014  001136           ADD     #RMER1,$BDADR
11477  045100  104250                           ERROR   250             ;OPI NOT SET DURING SEEK
11478
11479  045102  012737  045110  001124  40$:     MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
11480
11481                                   ;;*****************************************************************
11482                                   ;VERIFY THAT SEEK ABORTS DURING EXECUTION
11483
11484  045110                           50$:
11485                                   ;SET VOLUME VALID USING SUBROUTINE
11486  045110  004737  060526                   JSR     PC,SETVV        ;GO SET VOLUME VALID
11487  045114  000403                           BR      60$             ;BRANCH TO 60$ IF NO ERROR
11488  045116  104000                           ERROR                   ;RETURN HERE IF ERROR
11489  045120  000137  046334                   JMP     330$
11490
11491  045124                           60$:
11492                                   ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11493                                   ;       ADDRESS,  AND LOAD SEEK COMMAND
11494
11495  045124  012760  041001  000024           MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11496
11497  045132  012760  000000  000006           MOV     #0,RMDA(R0)     ;LOAD RMDA
11498
11499  045140  012760  000000  000034           MOV     #0,RMDC(R0)     ;LOAD RMDC
11500
11501  045146  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1
11502
11503  045154  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2
11504
11505  045162  012760  000005  000000           MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11506                           ;       STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11507  045170  012702  000003                   MOV     #3,R2
11508  045174                           70$:
11509
11510  045174  012760  141001  000024           MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11511
11512  045202  012760  041001  000024           MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11513  045210  005302                           DEC     R2
11514  045212  001370                           BNE     70$
11515                           ;       SET DRIVE FAULT TO CAUSE ABORT CONDITION
11516
11517  045214  012760  041101  000024           MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
11518                           ;       STEP 2 CLOCKS AND VERIFY GO IS RESET
11519  045222  012702  000002                   MOV     #2,R2
11520  045226                           80$:
11521
11522  045226  012760  141101  000024           MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
11523
11524  045234  012760  041101  000024           MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
```

```
11525  045242  005302                          DEC     R2
11526  045244  001370                          BNE     80$
11527
11528  045246  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11529  045254  042737  177776  001142          BIC     #^CGO,$BDDAT
11530  045262  001405                          BEQ     90$
11531  045264  010037  001136                  MOV     R0,$BDADR
11532  045270  005037  001140                  CLR     $GDDAT
11533  045274  104251                          ERROR   251             ;GO NOT RESET DUE TO ABORT
11534
11535  045276  012737  045304  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11536
11537                                  ;;*****************************************************************
11538                                  ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11539
11540  045304                          100$:
11541                                  ;SET VOLUME VALID USING SUBROUTINE
11542  045304  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11543  045310  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
11544  045312  104000                          ERROR                   ;RETURN HERE IF ERROR
11545  045314  000137  046334                  JMP     330$
11546  045320                          110$:
11547                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11548                                  ;       ADDRESS,  AND LOAD SEEK COMMAND
11549
11550  045320  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11551
11552  045326  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11553
11554  045334  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11555
11556  045342  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11557
11558  045350  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11559
11560  045356  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11561                                  ;       STEP THE COMMAND SEQUENCER
11562  045364  012702  000017                  MOV     #15.,R2
11563  045370                          120$:
11564
11565  045370  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11566
11567  045376  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11568  045404  005302                          DEC     R2
11569  045406  001370                          BNE     120$
11570                                  ;       VERIFY THAT OPI IS SET
11571
11572  045410  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11573  045416  042737  157777  001142          BIC     #^COPI,$BDDAT
11574  045424  001011                          BNE     130$
11575  045426  010037  001136                  MOV     R0,$BDADR
11576  045432  062737  000014  001136          ADD     #RMER1,$BDADR
11577  045440  012737  020000  001140          MOV     #OPI,$GDDAT
11578  045446  104252                          ERROR   252             ;OPI NOT SET DUE TO ON LATCH
11579
11580  045450  012737  045456  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
```

```
11581
11582                                   ;;*****************************************************************
11583                                   ;VERIFY ATA SETS IF DRIVE COMPLETES SEEK (ON CYLINDER SETS)
11584
11585  045456                           150$:
11586                                   ;SET VOLUME VALID USING SUBROUTINE
11587  045456  004737  060526                 JSR     PC,SETVV        ;GO SET VOLUME VALID
11588  045462  000403                         BR      160$            ;BRANCH TO 160$ IF NO ERROR
11589  045464  104000                         ERROR                   ;RETURN HERE IF ERROR
11590  045466  000137  046334                 JMP     330$
11591  045472                           160$:
11592                                   ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11593                                   ;       ADDRESS,  AND LOAD SEEK COMMAND
11594
11595  045472  012760  041401  000024         MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11596
11597  045500  012760  000000  000006         MOV     #0,RMDA(R0)     ;LOAD RMDA
11598
11599  045506  012760  000000  000034         MOV     #0,RMDC(R0)     ;LOAD RMDC
11600
11601  045514  012760  000000  000014         MOV     #0,RMER1(R0)    ;LOAD RMER1
11602
11603  045522  012760  000000  000042         MOV     #0,RMER2(R0)    ;LOAD RMER2
11604
11605  045530  012760  000005  000000         MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11606                                   ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11607  045536  012702  000015                 MOV     #13.,R2
11608  045542                           170$:
11609
11610  045542  012760  141401  000024         MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
11611
11612  045550  012760  041401  000024         MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11613  045556  005302                         DEC     R2
11614  045560  001370                         BNE     170$
11615                                   ;       DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
11616
11617  045562  012760  041001  000024         MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11618
11619  045570  012760  041401  000024         MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11620                                   ;       STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
11621  045576  012702  000003                 MOV     #3,R2
11622  045602                           180$:
11623
11624  045602  012760  141401  000024         MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
11625
11626  045610  012760  041401  000024         MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11627  045616  005302                         DEC     R2
11628  045620  001370                         BNE     180$
11629                                   ;       VERIFY ATA IS SET
11630
11631  045622  016037  000012  001142         MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
11632  045630  042737  077777  001142         BIC     #^CATA,$BDDAT
11633  045636  001011                         BNE     190$
11634  045640  012737  100000  001140         MOV     #ATA,$GDDAT
11635  045646  010037  001136                 MOV     R0,$BDADR
11636  045652  062737  000012  001136         ADD     #RMDS,$BDADR
```

```
11637  045660  104253                          ERROR   253             ;ATA NOT SET BY SEEK
11638
11639  045662  012737  045670  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11640
11641                                  ;;*****************************************************************
11642                                  ;VERIFY THAT SEEK ABORTS AFTER EXECUTION DURING WAIT LOOP
11643
11644  045670                          200$:
11645                                  ;SET VOLUME VALID USING SUBROUTINE
11646  045670  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11647  045674  000403                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
11648  045676  104000                          ERROR                   ;RETURN HERE IF ERROR
11649  045700  000137  046334                  JMP     330$
11650  045704                          210$:
11651                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11652                                  ;       ADDRESS  AND LOAD SEEK COMMAND
11653
11654  045704  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11655
11656  045712  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11657
11658  045720  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11659
11660  045726  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11661
11662  045734  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11663
11664  045742  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11665                                  ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
11666  045750  012702  000015                  MOV     #13.,R2
11667  045754                          220$:
11668
11669  045754  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
11670
11671  045762  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11672  045770  005302                          DEC     R2
11673  045772  001370                          BNE     220$
11674                                  ;       DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
11675
11676  045774  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11677                                  ;       STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
11678  046002  012702  000007                  MOV     #7,R2
11679  046006                          230$:
11680
11681  046006  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11682
11683  046014  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11684  046022  005302                          DEC     R2
11685  046024  001370                          BNE     230$
11686                                  ;       VERIFY THAT GO IS STILL SET
11687
11688  046026  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11689  046034  042737  177776  001142          BIC     #^CGO,$BDDAT
11690  046042  001006                          BNE     240$
11691  046044  012737  000001  001140          MOV     #GO,$GDDAT
11692  046052  010037  001136                  MOV     R0,$BDALR
```

```
11693  046056  104254                          ERROR   254             ;GO RESET EARLY DURING SEEK
11694                                          ;SET SEEK INCOMPLETE ERROR
11695  046060                          240$:
11696
11697  046060  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
11698                                          ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
11699  046066  012702  000003                  MOV     #3,R2
11700  046072                          250$:
11701
11702  046072  012760  141201  000024          MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
11703
11704  046100  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
11705  046106  005302                          DEC     R2
11706  046110  001370                          BNE     250$
11707
11708  046112  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11709  046120  042737  177776  001142          BIC     #^CGO,$BDDAT
11710  046126  001405                          BEQ     260$
11711  046130  010037  001136                  MOV     R0,$BDADR
11712  046134  005037  001140                  CLR     $GDDAT
11713  046140  104255                          ERROR   255             ;GO NOT RESET DUE TO WAIT ABORT
11714
11715  046142  012737  046154  001124  260$:   MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11716  046150  012703  000001                  MOV     #1,R3           ;INITIALIZE CYLINDER ADDRESS
11717
11718                                          ;;*****************************************************************
11719                                          ;VERIFY THE TAG BUS DURING SEEK
11720
11721  046154                          300$:
11722                                          ;SET VOLUME VALID USING SUBROUTINE
11723  046154  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11724  046160  000402                          BR      310$            ;BRANCH TO 310$ IF NO ERROR
11725  046162  104000                          ERROR                   ;RETURN HERE IF ERROR
11726  046164  000463                          BR      330$
11727  046166                          310$:
11728                                          ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11729                                          ;       ADDRESS  AND LOAD SEEK COMMAND
11730
11731  046166  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11732
11733  046174  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11734
11735  046202  010360  000034                  MOV     R3,RMDC(R0)     ;LOAD RMDC
11736
11737  046206  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11738
11739  046214  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11740
11741  046222  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
11742  046230  012702  046350                  MOV     #400$,R2        ;INITIALIZE TABLE POINTER
11743                                          ;       VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
11744  046234                          315$:
11745
11746  046234  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
11747  046242  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
11748  046250  011237  001140                  MOV     (R2),$GDDAT
```

```
11749  046254  050337  001140                  BIS     R3,$GDDAT       ;OR CYLINDER ADDRESS IN
11750  046260  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11751  046266  001407                          BEQ     320$            ;BRANCH IF TAG BUS OK
11752  046270  010037  001136                  MOV     R0,$BDADR
11753  046274  062737  000040  001136          ADD     #RMMR2,$BDADR
11754  046302  104256                          ERROR   256             ;INCORRECT TAG BUS DURING SEEK
11755  046304  000413                          BR      330$
11756  046306                          320$:
11757                                  ;       ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
11758  046306  062702  000002                  ADD     #2,R2
11759  046312  005712                          TST     (R2)
11760  046314  100407                          BMI     330$            ;EXIT IF ENTRY NEGATIVE
11761                                  ;       STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
11762
11763  046316  012760  151001  000024          MOV     #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0)        ;LOAD RMMR1
11764
11765  046324  012760  041401  000024          MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
11766  046332  000740                          BR      315$
11767  046334                          330$:
11768                                  ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
11769
11770  046334  006303                          ASL     R3              ;SHIFT TO NEXT CYLINDER
11771  046336  020327  000512                  CMP     R3,#512
11772  046342  101001                          BHI     340$            ;EXIT IF 512 WAS DONE
11773  046344  000703                          BR      300$            ;TEST NEXT CYLINDER
11774  046346  000416                  340$:   BR      500$            ;JUMP OVER TABLE
11775
11776  046350                          400$:
11777
11778                                  ;TABLE OF TAG BUS CONTROL AND BIT VALUES
11779
11780  046350  001777                          .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
11781  046352  001777                          .WORD   1777
11782  046354  004000                          .WORD   CC      ;CONTROL BITS ENABLED, BIT 6 ON
11783  046356  004000                          .WORD   CC
11784  046360  024000                          .WORD   TAG!CC  ;TAG COMES ON
11785  046362  024000                          .WORD   TAG!CC
11786  046364  024000                          .WORD   TAG!CC
11787  046366  024000                          .WORD   TAG!CC
11788  046370  024000                          .WORD   TAG!CC
11789  046372  024000                          .WORD   TAG!CC
11790  046374  004000                          .WORD   CC      ;TAG GOES OFF
11791  046376  004000                          .WORD   CC
11792  046400  001777                          .WORD   1777            ;CONTROL BITS DISABLED
11793
11794  046402  177777                          .WORD   -1              ;END OF TABLE
11795
11796  046404                          500$:                           ;END OF TEST
11797
11798                                  ;;*****************************************************************
11799                                  ;*TEST 115       SEARCH TEST
11800
11801                                  ;;*****************************************************************
11802  046404  000004                  TST115: SCOPE
11803  046406  012737  000115  001226          MOV     #115,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
11804
```

```
11805  046414  000240                          NOP
11806  046416  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
11807  046424  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
11808  046432  012737  046446  001122          MOV     #T115,$LPADR  ;LOAD LOOP ON TEST ADDRESS
11809  046440  012737  046446  001124          MOV     #T115,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
11810  046446                          T115:
11811  046446  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
11812  046452  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
11813  046456  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
11814
11815                                  ;;*****************************************************************
11816                                  ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
11817
11818                                  ;SET VOLUME VALID USING SUBROUTINE
11819  046462  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11820  046466  000403                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
11821  046470  104000                          ERROR                   ;RETURN HERE IF ERROR
11822  046472  000137  050504                  JMP     330$
11823  046476                          10$:
11824                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11825                                  ;       ADDRESS,  AND LOAD SEARCH COMMAND
11826
11827  046476  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11828
11829  046504  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11830
11831  046512  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11832
11833  046520  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11834
11835  046526  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11836
11837  046534  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
11838                                  ;       STEP COMMAND SEQUENCER TO SEARCH COM (2 CLOCKS)
11839  046542  012702  000002                  MOV     #2,R2
11840  046546                          20$:
11841
11842  046546  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11843
11844  046554  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11845  046562  005302                          DEC     R2
11846  046564  001370                          BNE     20$
11847                                  ;       DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
11848
11849  046566  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11850  046574  012702  000002                  MOV     #2,R2
11851  046600                          30$:
11852
11853  046600  012760  140001  000024          MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
11854
11855  046606  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
11856  046614  005302                          DEC     R2
11857  046616  001370                          BNE     30$
11858                                  ;       VERIFY THAT OPI IS SET
11859
11860  046620  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
```

```
11861  046626  042737  157777  001142          BIC     #^COPI,$BDDAT
11862  046634  001011                          BNE     40$
11863  046636  012737  020000  001140          MOV     #OPI,$GDDAT
11864  046644  010037  001136                  MOV     R0,$BDADR
11865  046650  062737  000014  001136          ADD     #RMER1,$BDADR
11866  046656  104257                          ERROR   257             ;OPI NOT SET DURING SEARCH
11867
11868  046660  012737  046666  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
11869
11870                                  ;;*****************************************************************
11871                                  ;VERIFY THAT SEARCH ABORTS DURING EXECUTION
11872
11873  046666                          50$:
11874                                  ;SET VOLUME VALID USING SUBROUTINE
11875  046666  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11876  046672  000403                          BR      60$             ;BRANCH TO 60$ IF NO ERROR
11877  046674  104000                          ERROR                   ;RETURN HERE IF ERROR
11878  046676  000137  050504                  JMP     330$
11879
11880  046702                          60$:
11881                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11882                                  ;       ADDRESS,  AND LOAD SEARCH COMMAND
11883
11884  046702  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11885
11886  046710  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11887
11888  046716  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11889
11890  046724  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11891
11892  046732  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11893
11894  046740  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
11895                                  ;       STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
11896  046746  012702  000003                  MOV     #3,R2
11897  046752                          70$:
11898
11899  046752  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11900
11901  046760  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11902  046766  005302                          DEC     R2
11903  046770  001370                          BNE     70$
11904                                  ;       SET DRIVE FAULT TO CAUSE ABORT CONDITION
11905
11906  046772  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
11907                                  ;       STEP 2 CLOCKS AND VERIFY GO IS RESET
11908  047000  012702  000002                  MOV     #2,R2
11909  047004                          80$:
11910
11911  047004  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
11912
11913  047012  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
11914  047020  005302                          DEC     R2
11915  047022  001370                          BNE     80$
11916
```

```
11917  047024  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
11918  047032  042737  177776  001142          BIC     #↑CGO,$BDDAT
11919  047040  001405                          BEQ     90$
11920  047042  010037  001136                  MOV     R0,$BDADR
11921  047046  005037  001140                  CLR     $GDDAT
11922  047052  104260                          ERROR   260             ;GO NOT RESET DUE TO ABORT
11923
11924                                  ;(THE OTHER TWO ABORT TESTS IN THE COMMAND SEQUENCER ARE TESTED
11925                                  ;DURING DATA COMMAND TESTS)
11926
11927  047054  012737  047062  001124  90$:     MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
11928
11929                                  ;;*****************************************************************
11930                                  ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
11931
11932  047062                          100$:
11933                                  ;SET VOLUME VALID USING SUBROUTINE
11934  047062  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11935  047066  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
11936  047070  104000                          ERROR                   ;RETURN HERE IF ERROR
11937  047072  000137  050504                  JMP     330$
11938  047076                          110$:
11939                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11940                                  ;       ADDRESS,  AND LOAD SEARCH COMMAND
11941
11942  047076  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11943
11944  047104  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11945
11946  047112  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11947
11948  047120  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11949
11950  047126  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11951
11952  047134  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
11953                                  ;       STEP THE COMMAND SEQUENCER
11954  047142  012702  000023                  MOV     #19.,R2
11955  047146                          120$:
11956
11957  047146  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
11958
11959  047154  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
11960  047162  005302                          DEC     R2
11961  047164  001370                          BNE     120$
11962                                  ;       VERIFY THAT OPI IS SET
11963
11964  047166  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
11965  047174  042737  157777  001142          BIC     #↑COPI,$BDDAT
11966  047202  001011                          BNE     130$
11967  047204  010037  001136                  MOV     R0,$BDADR
11968  047210  062737  000014  001136          ADD     #RMER1,$BDADR
11969  047216  012737  020000  001140          MOV     #OPI,$GDDAT
11970  047224  104261                          ERROR   261             ;OPI NOT SET DUE TO ON LATCH
11971
11972  047226  012737  047234  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
```

```
11973
11974                                   ;;*****************************************************************
11975                                   ;VERIFY ATA SETS IF DRIVE COMPLETES SEARCH (ON CYLINDER AND
11976                                   ;SECTOR COMPARE SETS)
11977
11978  047234                           150$:
11979                                   ;SET VOLUME VALID USING SUBROUTINE
11980  047234  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
11981  047240  000403                          BR      160$            ;BRANCH TO 160$ IF NO ERROR
11982  047242  104000                          ERROR                   ;RETURN HERE IF ERROR
11983  047244  000137  050504                  JMP     330$
11984  047250                           160$:
11985                                   ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
11986                                   ;       ADDRESS,  AND LOAD SEARCH COMMAND
11987
11988  047250  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
11989
11990  047256  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
11991
11992  047264  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
11993
11994  047272  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
11995
11996  047300  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
11997
11998  047306  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
11999                                   ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
12000  047314  012702  000021                  MOV     #17.,R2
12001  047320                           170$:
12002
12003  047320  012760  141401  000024          MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
12004
12005  047326  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12006  047334  005302                          DEC     R2
12007  047336  001370                          BNE     170$
12008                                   ;       DROP ON CYLINDER TO RESET LATCH,AND RAISE INDEX PULSE
12009                                   ;TO SET FORMAT CHANGE FLOP
12010
12011
12012  047340  012760  041005  000024          MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
12013                                   ;       RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
12014
12015  047346  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12016                                   ;       STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
12017  047354  012702  000002                  MOV     #2,R2
12018  047360                           180$:
12019
12020  047360  012760  151401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
12021
12022  047366  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12023  047374  005302                          DEC     R2
12024  047376  001370                          BNE     180$
12025                                   ;FORCE SECTOR COMPARE BY CLOCKING SECTOR PULSE WITH SECTOR COMPARE
12026                                   ;ACTIVE
12027
12028  047400  012760  051403  000024          MOV     #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(R0)     ;LOAD RMMR1
```

```
12029
12030  047406  012760  051443  000024          MOV     #DMD!MUR!MOC!DBEN!DTO!MSC!MS,RMMR1(R0)  ;LOAD RMMR1
12031
12032  047414  012760  051403  000024          MOV     #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(R0)     ;LOAD RMMR1
12033                                  ;CLOCK SEQUENCER TO SET ATA (3 CLOCKS)
12034  047422  012702  000003                  MOV     #3,R2           ;R2 = CLOCK COUNT
12035  047426                          185$:
12036
12037  047426  012760  151401  000024          MOV     #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
12038
12039  047434  012760  051401  000024          MOV     #DMD!MUR!MOC!DBEN!DTO,RMMR1(R0) ;LOAD RMMR1
12040  047442  005302                          DEC     R2
12041  047444  001370                          BNE     185$
12042                                  ;       VERIFY ATA IS SET
12043
12044  047446  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
12045  047454  042737  077777  001142          BIC     #^CATA,$BDDAT
12046  047462  001011                          BNE     190$
12047  047464  012737  100000  001140          MOV     #ATA,$GDDAT
12048  047472  010037  001136                  MOV     R0,$BDADR
12049  047476  062737  000012  001136          ADD     #RMDS,$BDADR
12050  047504  104262                          ERROR   26              ;ATA NOT SET BY SEARCH
12051
12052  047506  012737  047514  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12053
12054                                  ;;*****************************************************************
12055                                  ;VERIFY THAT SEARCH ABORTS AFTER EXECUTION DURING SEARCH SEEK LOOP
12056
12057  047514                          200$:
12058                                  ;SET VOLUME VALID USING SUBROUTINE
12059  047514  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12060  047520  000403                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
12061  047522  104000                          ERROR                   ;RETURN HERE IF ERROR
12062  047524  000137  050504                  JMP     330$
12063  047530                          210$:
12064                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
12065                                  ;       ADDRESS  AND LOAD SEARCH COMMAND
12066
12067  047530  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12068
12069  047536  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12070
12071  047544  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12072
12073  047552  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12074
12075  047560  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12076
12077  047566  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
12078                                  ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
12079  047574  012702  000021                  MOV     #17 ,R2
12080  047600                          220$:
12081
12082  047600  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
12083
12084  047606  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC RMMR1(R0)     ;LOAD RMMR1
```

```
12085  047614  005302                          DEC     R2
12086  047616  001370                          BNE     220$
12087                                  ;       DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
12088
12089  047620  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
12090                                  ;       STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
12091  047626  012702  000007                  MOV     #7,R2
12092  047632                          230$:
12093
12094  047632  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
12095
12096  047640  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
12097  047646  005302                          DEC     R2
12098  047650  001370                          BNE     230$
12099                                  ;       VERIFY THAT GO IS STILL SET
12100
12101  047652  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
12102  047660  042737  177776  001142          BIC     #^CGO,$BDDAT
12103  047666  001006                          BNE     240$
12104  047670  012737  000001  001140          MOV     #GO,$GDDAT
12105  047676  010037  001136                  MOV     R0,$BDADR
12106  047702  104263                          ERROR   263             ;GO RESET EARLY DURING SEARCH
12107                                  ;       SET SEEK INCOMPLETE ERROR
12108  047704                          240$:
12109
12110  047704  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
12111                                  ;       STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
12112  047712  012702  000003                  MOV     #3,R2
12113  047716                          250$:
12114
12115  047716  012760  141201  000024          MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
12116
12117  047724  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
12118  047732  005302                          DEC     R2
12119  047734  001370                          BNE     250$
12120
12121  047736  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
12122  047744  042737  177776  001142          BIC     #^CGO,$BDDAT
12123  047752  001405                          BEQ     260$
12124  047754  010037  001136                  MOV     R0,$BDADR
12125  047760  005037  001140                  CLR     $GDDAT
12126  047764  104264                          ERROR   264             ;GO NOT RESET DUE TO WAIT ABORT
12127
12128  047766  012737  047774  001124  260$:   MOV     #265$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12129
12130                                  ;;*****************************************************************
12131                                  ;VERIFY THAT SEARCH ABORTS DURING SECTOR COMPARE LOOP
12132
12133  047774                          265$:
12134                                  ;SET VOLUME VALID USING SUBROUTINE
12135  047774  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12136  050000  000403                          BR      270$            ;BRANCH TO 270$ IF NO ERROR
12137  050002  104000                          ERROR                   ;RETURN HERE IF ERROR
12138  050004  000137  050504                  JMP     330$
12139  050010                          270$:
12140                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
```

```
12141                                   ;       ADDRESS,  AND LOAD SEARCH COMMAND
12142
12143  050010  012760  041401  000024           MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12144
12145  050016  012760  000000  000006           MOV     #0,RMDA(R0)     ;LOAD RMDA
12146
12147  050024  012760  000000  000034           MOV     #0,RMDC(R0)     ;LOAD RMDC
12148
12149  050032  012760  000000  000014           MOV     #0,RMER1(R0)    ;LOAD RMER1
12150
12151  050040  012760  000000  000042           MOV     #0,RMER2(R0)    ;LOAD RMER2
12152
12153  050046  012760  000031  000000           MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
12154                                   ;       STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
12155  050054  012702  000021                   MOV     #17.,R2
12156  050060                          275$:
12157
12158  050060  012760  141401  000024           MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
12159
12160  050066  012760  041401  000024           MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12161  050074  005302                           DEC     R2
12162  050076  001370                           BNE     275$
12163                                   ;       DROP ON CYLINDER TO RESET LATCH, AND RAISE INDEX PULSE
12164                                   ;TO SET FORMAT CHANGE FLOP
12165
12166
12167  050100  012760  041005  000024           MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
12168                                   ;       RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
12169
12170  050106  012760  051401  000024           MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12171                                   ;       STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
12172  050114  012702  000002                   MOV     #2,R2
12173  050120                          280$:
12174
12175  050120  012760  151401  000024           MOV     #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
12176
12177  050126  012760  051401  000024           MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12178  050134  005302                           DEC     R2
12179  050136  001370                           BNE     280$
12180                                   ;       VERIFY THAT SEARCH ENABLE IS ON DURING SECTOR COMPARE LOOP
12181  050140  012702  000004                   MOV     #4,R2           ;R2 = CLOCK COUNT
12182  050144                          281$:
12183
12184  050144  016037  000024  001142           MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12185  050152  042737  173777  001142           BIC     #^tCESRC,$BDDAT
12186  050160  001411                           BEQ     282$            ;BRANCH IF SEARCH NOT ENABLED
12187
12188  050162  012760  151401  000024           MOV     #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
12189
12190  050170  012760  051401  000024           MOV     #DMD!MUR!MOC!DBEN!DTO,RMMR1(R0) ;LOAD RMMR1
12191  050176  005302                           DEC     R2
12192  050200  001361                           BNE     281$
12193  050202  000411                           BR      283$
12194  050204  012737  004000  001140  282$:    MOV     #ESRC,$GDDAT
12195  050212  010037  001136                   MOV     R0,$BDADR
12196  050216  062737  000024  001136           ADD     #RMMR1,$BDADR
```

```
12197  050224  104265                          ERROR   265             ;SEARCH NOT ENABLED
12198  050226                          283$:
12199                                  ;       SET DRIVE FAULT TO CAUSE ABORT CONDITION
12200
12201  050226  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!MDF,RMMR1(R0)     ;LOAD RMMR1
12202                                  ;       STEP 2 CLOCKS AND VERIFY GO IS RESET
12203  050234  012702  000002                  MOV     #2,R2           ;R2 = CLOCK COUNT
12204  050240                          285$:
12205
12206  050240  012760  151501  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!MDF!DBCk,RMMR1(R0)        ;LOAD RMMR1
12207
12208  050246  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!MDF,RMMR1(R0)     ;LOAD RMMR1
12209  050254  005302                          DEC     R2
12210  050256  001370                          BNE     285$
12211
12212  050260  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
12213  050266  042737  177776  001142          BIC     #^CGO,$BDDAT
12214  050274  001406                          BEQ     290$
12215  050276  012737  000000  001140          MOV     #0,$GDDAT
12216  050304  010037  001136                  MOV     R0,$BDADR
12217  050310  104260                          ERROR   260             ;GO NOT RESET DURING SECTOR COMPARE
12218
12219  050312  012737  050324  001124  290$:   MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12220  050320  012703  000001                  MOV     #1,R3           ;INITIALIZE CYLINDER ADDRESS
12221
12222                                  ;;**************************************************************
12223                                  ;VERIFY THE TAG BUS DURING SEARCH
12224
12225  050324                          300$:
12226                                  ;SET VOLUME VALID USING SUBROUTINE
12227  050324  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12228  050330  000402                          BR      310$            ;BRANCH TO 310$ IF NO ERROR
12229  050332  104000                          ERROR                   ;RETURN HERE IF ERROR
12230  050334  000463                          BR      330$
12231  050336                          310$:
12232                                  ;       ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
12233                                  ;       ADDRESS  AND LOAD SEARCH COMMAND
12234
12235  050336  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12236
12237  050344  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12238
12239  050352  010360  000034                  MOV     R3,RMDC(R0)     ;LOAD RMDC
```

```
12240
12241  050356  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12242
12243  050364  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12244
12245  050372  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
12246  050400  012702  050520                  MOV     #400$,R2        ;INITIALIZE TABLE POINTER
12247                                          ;
                                               VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
12248  050404                          315$:
12249
12250  050404  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
12251  050412  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
12252  050420  011237  001140                  MOV     (R2),$GDDAT
12253  050424  050337  001140                  BIS     R3,$GDDAT       ;OR CYLINDER ADDRESS IN
12254  050430  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
12255  050436  001407                          BEQ     320$            ;BRANCH IF TAG BUS OK
12256  050440  010037  001136                  MOV     R0,$BDADR
12257  050444  062737  000040  001136          ADD     #RMMR2,$BDADR
12258  050452  104266                          ERROR   266             ;INCORRECT TAG BUS DURING SEARCH
12259  050454  000420                          BR      340$
12260  050456                          320$:
12261                                          ;       ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
12262  050456  062702  000002                  ADD     #2,R2
12263  050462  005712                          TST     (R2)
12264  050464  100407                          BMI     330$            ;EXIT IF ENTRY NEGATIVE
12265                                          ;       STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
12266
12267  050466  012760  141401  000024          MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
12268
12269  050474  012760  041401  000024          MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
12270  050502  000740                          BR      315$
12271  050504                          330$:
12272                                          ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
12273
12274  050504  006303                          ASL     R3              ;SHIFT TO NEXT CYLINDER
12275  050506  020327  000512                  CMP     R3,#512
12276  050512  101001                          BHI     340$            ;EXIT IF 512 WAS DONE
12277  050514  000703                          BR      300$            ;TEST NEXT CYLINDER
12278  050516  000424                  340$:   BR      500$            ;JUMP OVER TABLE
12279
12280  050520                          400$:
12281
12282                                          ;TABLE OF TAG BUS CONTROL AND BIT VALUES
12283
12284  050520  001777                          .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
12285  050522  001777                          .WORD   1777
12286  050524  001777                          .WORD   1777
12287  050526  001777                          .WORD   1777
12288  050530  001777                          .WORD   1777
12289  050532  004000                          .WORD   CC      ;CONTROL BITS ENABLED, BIT 6 ON
12290  050534  004000                          .WORD   CC
12291  050536  024000                          .WORD   TAG!CC  ;TAG COMES ON
12292  050540  024000                          .WORD   TAG!CC
12293  050542  024000                          .WORD   TAG!CC
12294  050544  024000                          .WORD   TAG!CC
12295  050546  024000                          .WORD   TAG!CC
```

```
12296  050550  024000                          .WORD   TAG!CC
12297  050552  004000                          .WORD   CC      ;TAG GOES OFF
12298  050554  004000                          .WORD   CC
12299  050556  001777                          .WORD   1777            ;CONTROL BITS DISABLED
12300  050560  001777                          .WORD   1777
12301  050562  001777                          .WORD   1777
12302  050564  001777                          .WORD   1777
12303
12304  050566  177777                          .WORD   -1              ;END OF TABLE
12305
12306  050570                          500$:                           ;END OF TEST
12307
12308                                  ;;**********************************************************************
12309                                  ;*TEST 116       SEARCH TIMEOUT TEST
12310
12311                                  ;;**********************************************************************
12312  050570  000004                  TST116: SCOPE
12313  050572  012737  000116  001226          MOV     #116,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
12314
12315  050600  000240                          NOP
12316  050602  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
12317  050610  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
12318  050616  012737  050632  001122          MOV     #T116,$LPADR  ;LOAD LOOP ON TEST ADDRESS
12319  050624  012737  050632  001124          MOV     #T116,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
12320  050632                          T116:
12321  050632  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
12322  050636  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
12323  050642  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
12324                                  ;SET VOLUME VALID USING SUBROUTINE
12325  050646  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12326  050652  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
12327  050654  104000                          ERROR                   ;RETURN HERE IF ERROR
12328  050656  000550                          BR      90$
12329  050660                          10$:
12330                                  ;ENABLE DEBUG CLOCK AND LOAD SEARCH COMMAND
12331
12332  050660  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12333
12334  050666  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12335
12336  050674  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12337
12338  050702  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12339
12340  050710  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12341
12342  050716  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
12343                                  ;EXECUTE SEARCH TO TEST FOR ON LATCH RESET (17 CLOCKS)
12344  050724  012702  000021                  MOV     #17.,R2
12345  050730                          20$:
12346
12347  050730  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
12348
12349  050736  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12350  050744  005302                          DEC     R2
12351  050746  001370                          BNE     20$
```

```
12352                                   ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER
12353
12354  050750  012760  041001  000024           MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
12355
12356  050756  012760  041401  000024           MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
12357                                   ;STEP COMMAND SEQUENCER TO SECTOR COMPARE LOOP (2 CLOCKS)
12358  050764  012702  000002                   MOV     #2,R2
12359  050770                          30$:
12360
12361  050770  012760  151401  000024           MOV     #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(R0)    ;LOAD RMMR1
12362
12363  050776  012760  051401  000024           MOV     #DMD!DBEN!MUR!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12364  051004  005302                           DEC     R2
12365  051006  001370                           BNE     30$
12366                                   ;THE COMMAND SEQUENCER IS NOW IN THE SECTOR COMPARE LOOP.  STEP
12367                                   ;THROUGH THE LOOP AND VERIFY SEARCH IS ENABLED.
12368  051010  012702  000005                   MOV     #5,R2
12369  051014                          40$:
12370
12371  051014  012760  151401  000024           MOV     #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(R0)    ;LOAD RMMR1
12372
12373  051022  012760  051401  000024           MOV     #DMD!DBEN!MUR!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
12374
12375  051030  016037  000024  001142           MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12376  051036  042737  173777  001142           BIC     #^CESRC,$BDDAT
12377  051044  001403                           BEQ     50$             ;BRANCH IF SEARCH NOT ENABLED
12378  051046  005302                           DEC     R2
12379  051050  001361                           BNE     40$
12380  051052  000412                           BR      60$
12381  051054  012737  004000  001140  50$:     MOV     #ESRC,$GDDAT    ;SETUP ERROR MSG
12382  051062  010037  001136                   MOV     R0,$BDADR
12383  051066  062737  000024  001136           ADD     #RMMR1,$BDADR
12384  051074  104265                           ERROR   265             ;SEARCH NOT ENABLED
12385  051076  000440                           BR      90$
12386  051100                          60$:
12387                                   ;DROP DTO TO ENABLE SEARCH TIMEOUT AND WAIT FOR OPI TO SET.
12388
12389  051100  012760  041401  000024           MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12390  051106  012737  000074  001524           MOV     #60.,WATCH      ;SET WATCHDOG TIMER VALUE
12391  051114  004777  130406                   JSR     PC,@CLOCK       ;START THE CLOCK
12392  051120                          70$:
12393
12394  051120  016037  000014  001142           MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
12395  051126  042737  157777  001142           BIC     #^COPI,$BDDAT
12396  051134  001017                           BNE     80$
12397  051136  005737  001524                   TST     WATCH
12398  051142  001366                           BNE     70$
12399  051144  004777  130360                   JSR     PC,@STOP        ;STOP THE CLOCK
12400  051150  012737  020000  001140           MOV     #OPI,$GDDAT     ;SETUP ERROR MSG
12401  051156  010037  001136                   MOV     R0,$BDADR
12402  051162  062737  000014  001136           ADD     #RMER1,$BDADR
12403  051170  104267                           ERROR   267             ;OPI NOT SET BY SEARCH TIMEOUT
12404  051172  000402                           BR      90$
12405  051174                          80$:
12406  051174  004777  130330                   JSR     PC,@STOP        ;STOP THE CLOCK
12407
```

```
12408  051200                                  90$:                            ;END OF TEST
12409
12410                                          ;;*****************************************************************
12411                                          ;*TEST 117      DATA COMMAND TESTS (1)
12412
12413                                          ;;*****************************************************************
12414  051200  000004                         TST117: SCOPE
12415  051202  012737  000117  001226                 MOV     #117,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
12416
12417  051210  000240                                 NOP
12418  051212  012737  000024  001120                 MOV     #20.,$ICNT      ;20 ITERATIONS
12419  051220  112737  000001  001131                 MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
12420  051226  012737  051242  001122                 MOV     #T117,$LPADR  ;LOAD LOOP ON TEST ADDRESS
12421  051234  012737  051242  001124                 MOV     #T117,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
12422  051242                                 T117:
12423  051242  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
12424  051246  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
12425  051252  013701  001456                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
12426                                          ;;*****************************************************************
12427                                          ;VERIFY DATA COMMAND SETS OPI IF DRIVE NOT READY
12428                                          ;SET VOLUME VALID USING SUBROUTINE
12429  051256  004737  060526                        JSR     PC,SETVV        ;GO SET VOLUME VALID
12430  051262  000402                                BR      10$             ;BRANCH TO 10$ IF NO ERROR
12431  051264  104000                                ERROR                   ;RETURN HERE IF ERROR
12432  051266  000471                                BR      40$
12433  051270                                 10$:
12434                                          ;       ENABLE DEBUG CLOCK AND LOAD READ COMMAND
12435
12436  051270  012760  041401  000024                 MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12437
12438  051276  012760  000000  000014                 MOV     #0,RMER1(R0)    ;LOAD RMER1
12439
12440  051304  012760  000000  000042                 MOV     #0,RMER2(R0)    ;LOAD RMER2
12441
12442  051312  012760  000000  000006                 MOV     #0,RMDA(R0)     ;LOAD RMDA
12443
12444  051320  012760  000000  000034                 MOV     #0,RMDC(R0)     ;LOAD RMDC
12445
12446  051326  012760  000071  000000                 MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12447                                          ;       STEP COMMAND SEQUENCER TO UNIT READY TEST (3 CLOCKS)
12448  051334  012702  000003                        MOV     #3,R2
12449  051340                                 20$:
12450
12451  051340  012760  141401  000024                 MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12452
12453  051346  012760  041401  000024                 MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12454  051354  005302                                DEC     R2
12455  051356  001370                                BNE     20$
12456                                          ;       DROP UNIT READY
12457
12458  051360  012760  040401  000024                 MOV     #DMD!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12459                                          ;       STEP SEQUENCER AND VERIFY OPI SETS (2 CLOCKS)
12460  051366  012702  000002                        MOV     #2,R2
12461  051372                                 30$:
12462
12463  051372  012760  140401  000024                 MOV     #DMD!MOC!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
```

```
12464
12465  051400  012760  040401  000024          MOV     #DMD!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
12466  051406  005302                          DEC     R2
12467  051410  001370                          BNE     30$
12468
12469  051412  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
12470  051420  042737  157777  001142          BIC     #^C<OPI>,$BDDAT
12471  051426  001011                          BNE     40$
12472  051430  012737  020000  001140          MOV     #OPI,$GDDAT
12473  051436  010037  001136                  MOV     R0,$BDADR
12474  051442  062737  000014  001136          ADD     #RMER1,$BDADR
12475  051450  104270                          ERROR   270             ;OPI NOT SET DURING DATA
12476
12477  051452  012737  051460  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR TEST
12478
12479                                  ;;*****************************************************************
12480                                  ;VERIFY DATA COMMAND ABORTS AT LOCATION 129
12481
12482  051460                          50$:
12483                                  ;SET VOLUME VALID USING SUBROUTINE
12484  051460  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12485  051464  000402                          BR      60$             ;BRANCH TO 60$ IF NO ERROR
12486  051466  104000                          ERROR                   ;RETURN HERE IF ERROR
12487  051470  000576                          BR      150$
12488  051472                          60$:
12489                                  ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
12490
12491  051472  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12492
12493  051500  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12494
12495  051506  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12496
12497  051514  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12498
12499  051522  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12500
12501  051530  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12502                                  ;STEP COMMAND SEQUENCER TO ABORT TEST AT LOCATION 129 (4 CLOCKS)
12503  051536  012702  000004                  MOV     #4,R2
12504  051542                          70$:
12505
12506  051542  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12507
12508  051550  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12509  051556  005302                          DEC     R2
12510  051560  001370                          BNE     70$
12511                                  ;SET DEVICE FAULT TO CAUSE ABORT CONDITION
12512
12513  051562  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12514                                  ;STEP THE SEQUENCER THROUGH THE TEST FOR ABORT (1 CLOCK)
12515  051570  012702  000001                  MOV     #1,R2
12516  051574                          80$:
12517
12518  051574  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
12519
```

```
12520  051602  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12521  051610  005302                          DEC     R2
12522  051612  001370                          BNE     80$
12523
12524                                  ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
12525                                  ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
12526  051614  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
12527  051620                          85$:
12528
12529  051620  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
12530
12531  051626  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12532
12533  051634  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12534  051642  042737  157777  001142          BIC     #^CEBL,$BDDAT
12535  051650  001014                          BNE     90$             ;BRANCH IF EBL IS SET
12536
12537  051652  005302                          DEC     R2
12538  051654  001361                          BNE     85$             ;CONTINUE BIT CLOCKS IF COUNT NOT 0
12539  051656  012737  020000  001140          MOV     #EBL,$GDDAT
12540  051664  010037  001136                  MOV     R0,$BDADR
12541  051670  062737  000024  001136          ADD     #RMMR1,$BDADR
12542  051676  104271                          ERROR   271             ;EBL NOT SET AT DATA ABORT
12543  051700  000472                          BR      150$
12544  051702                          90$:
12545                                  ;STEP THE SEQUENCER THROUGH ITS TEST FOR EBL (2 CLOCKS)
12546  051702  012702  000002                  MOV     #2,R2
12547  051706                          100$:
12548
12549  051706  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)     ;LOAD RMMR1
12550
12551  051714  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12552  051722  005302                          DEC     R2
12553  051724  001370                          BNE     100$
12554
12555                                  ;ABORT EBL SHOULD NOW BE INACTIVE - FORCE BIT CLOCK USING THE
12556                                  ;MAINTENANCE REGISTER TO RESET EBL (16 BIT CLOCKS)
12557  051726  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
12558  051732                          110$:
12559
12560  051732  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)     ;LOAD RMMR1
12561
12562  051740  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12563  051746  005302                          DEC     R2
12564  051750  001370                          BNE     110$            ;ISSUE 16 BIT CLOCKS THEN TEST
12565  051752                          120$:
12566                                  ;VERIFY EBL IS NOW RESET
12567
12568  051752  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12569  051760  042737  157777  001142          BIC     #^CEBL,$BDDAT
12570  051766  001411                          BEQ     130$            ;BRANCH IF EBL IS RESET
12571  051770  005037  001140                  CLR     $GDDAT
12572  051774  010037  001136                  MOV     R0,$BDADR
12573  052000  062737  000024  001136          ADD     #RMMR1,$BDADR
12574  052006  104273                          ERROR   273             ;EBL DIDNT RESET ON TIME
12575  052010  000426                          BR      150$
```

```
12576  052012                                130$:
12577                                        ;VERIFY GO RESETS WITHIN 4 CLOCK CYCLES
12578  052012  012702  000004                MOV     #4,R2
12579  052016                                140$:
12580
12581  052016  012760  141501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
12582
12583  052024  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12584  052032  005302                        DEC     R2
12585  052034  001370                        BNE     140$
12586
12587  052036  016037  000000  001142        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
12588  052044  042737  177776  001142        BIC     #^CGO,$BDDAT
12589  052052  001405                        BEQ     150$
12590  052054  005037  001140                CLR     $GDDAT
12591  052060  010037  001136                MOV     R0,$BDADR
12592  052064  104274                        ERROR   274             ;GO NOT RESET DURING DATA ABORT
12593
12594  052066  012737  052074  001124  150$:  MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12595
12596                                        ;;*****************************************************************
12597                                        ;VERIFY SEQUENCER BRANCHES TO SEEK WHEN RUN AND GO FLOP SETS
12598
12599  052074                                200$:
12600                                        ;SET VOLUME VALID USING SUBROUTINE
12601  052074  004737  060526                JSR     PC,SETV.        ;GO SET VOLUME VALID
12602  052100  000402                        BR      210$            ;BRANCH TO 210$ IF NO ERROR
12603  052102  104000                        ERROR                   ;RETURN HERE IF ERROR
12604  052104  000512                        BR      250$
12605  052106                                210$:
12606                                        ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12607
12608  052106  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12609
12610  052114  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
12611
12612  052122  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
12613
12614  052130  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
12615
12616  052136  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
12617
12618  052144  012760  000071  000000        MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12619                                        ;MOVE SEQUENCER TO TEST FOR RUN AND GO AT LOCATION 130 (5 CLOCKS)
12620  052152  012702  000005                MOV     #5,R2
12621  052156                                220$:
12622
12623  052156  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12624
12625  052164  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12626  052172  005302                        DEC     R2
12627  052174  001370                        BNE     220$
12628                                        ;VERIFY RUN AND GO IS SET
12629
12630  052176  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12631  052204  042737  137777  001142        BIC     #^CRG,$BDDAT
```

```
12632  052212  001012                          BNE     230$
12633  052214  012737  040000  001140          MOV     #RG,$GDDAT
12634  052222  010037  001136                  MOV     R0,$BDADR
12635  052226  062737  000024  001136          ADD     #RMMR1,$BDADR
12636  052234  104275                          ERROR   275             ;RUN AND GO NOT SET
12637  052236  000435                          BR      250$
12638  052240                          230$:
12639                                  ;VERIFY THAT CYLINDER TAG COMES UP IN ONE CLOCK CYCLE
12640  052240  012702  000001                  MOV     #1,R2
12641  052244                          240$:
12642
12643  052244  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12644
12645  052252  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12646  052260  005302                          DEC     R2
12647  052262  001370                          BNE     240$
12648
12649  052264  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
12650  052272  042737  160000  001142          BIC     #RQA!RQB!TAG,$BDDAT
12651  052300  012737  004000  001140          MOV     #CC,$GDDAT
12652  052306  023737  001140  001142          CMP     $GDDAT,$BDDAT
12653  052314  001406                          BEQ     250$
12654  052316  010037  001136                  MOV     R0,$BDADR
12655  052322  062737  000040  001136          ADD     #RMMR2,$BDADR
12656  052330  104276                          ERROR   276             ;CYLINDER TAG NOT SET DURING DATA
12657
12658  052332  012737  052344  001124  250$:   MOV     #260$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
12659
12660                                  ;;*****************************************************************
12661                                  ;VERIFY DATA COMMAND ABORTS AT COMMAND SEQUENCER LOCATIONS 144, 145
12662  052340  012702  000144                  MOV     #144,R2         ;INITIALIZE TEST LOCATION
12663  052344                          260$:
12664                                  ;SET VOLUME VALID USING SUBROUTINE
12665  052344  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12666  052350  000402                          BR      270$            ;BRANCH TO 270$ IF NO ERROR
12667  052352  104000                          ERROR                   ;RETURN HERE IF ERROR
12668  052354  000553                          BR      320$
12669  052356                          270$:
12670                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12671
12672  052356  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12673
12674  052364  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12675
12676  052372  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12677
12678  052400  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12679
12680  052406  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12681
12682  052414  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12683                                  ;WAIT FOR RUN AND GO TO SET
12684  052422  012737  000310  001524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
12685  052430  004777  127072                  JSR     PC,@CLOCK       ;START THE CLOCK
12686  052434                          280$:
12687
```

```
12688  052434  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12689  052442  042737  137777  001142          BIC     #↑CRG,$BDDAT
12690  052450  001017                          BNE     290$
12691  052452  005737  001524                  TST     WATCH
12692  052456  001366                          BNE     280$
12693  052460  004777  127044                  JSR     PC,@STOP        ;STOP THE CLOCK
12694  052464  012737  040000  001140          MOV     #RG,$GDDAT
12695  052472  010037  001136                  MOV     R0,$BDADR
12696  052476  062737  000024  001136          ADD     #RMMR1,$BDADR
12697  052504  104275                          ERROR   275             ;RUN AND GO NOT SET
12698  052506  000476                          BR      320$
12699  052510                          290$:
12700  052510  004777  127014                  JSR     PC,@STOP        ;STOP THE CLOCK
12701                                  ;MOVE COMMAND SEQUENCER TO ABORT TEST (LOCATION 144 OR 145)
12702  052514  012703  000006                  MOV     #6,R3           ;SETUP CLOCK COUNT
12703  052520  022702  000144                  CMP     #144,R2
12704  052524  001402                          BEQ     300$
12705  052526  012703  000007                  MOV     #7,R3
12706  052532                          300$:
12707
12708  052532  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12709
12710  052540  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12711  052546  005303                          DEC     R3
12712  052550  001370                          BNE     300$
12713                                  ;SET DRIVE FAULT TO FORCE ABORT CONDITION
12714
12715  052552  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12716                                  ;CLOCK SEQUENCER THROUGH ITS TEST FOR ABORT (1 CLOCK)
12717  052560  012703  000001                  MOV     #1,R3
12718  052564                          305$:
12719
12720  052564  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
12721
12722  052572  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12723  052600  005303                          DEC     R3
12724  052602  001370                          BNE     305$            ;ISSUE 2 CLOCKS
12725
12726                                  ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
12727                                  ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
12728  052604  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
12729  052610                          306$:
12730
12731  052610  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
12732
12733  052616  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
12734
12735  052624  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12736  052632  042737  157777  001142          BIC     #↑CEBL,$BDDAT
12737  052640  001013                          BNE     310$            ;BRANCH IF EBL IS SET
12738
12739  052642  005302                          DEC     R2
12740  052644  001361                          BNE     306$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
12741  052646  012737  020000  001140          MOV     #EBL,$GDDAT
12742  052654  010037  001136                  MOV     R0,$BDADR
12743  052660  062737  000024  001136          ADD     #RMMR1,$BDADR
```

```
12744  052666  104271                        ERROR   271             ;SEQ DID NOT ABORT AT DATA SEEK
12745  052670  022702  000144        310$:   CMP     #144,R2
12746  052674  001003                        BNE     320$
12747  052676  012702  000145                MOV     #145,R2
12748  052702  000620                        BR      260$
12749
12750  052704  012737  052716  001124 320$:  MOV     #330$,$LPERR            ;CHANGE LOOP ON ERROR ADDRESS
12751
12752                                ;;*********************************************************************
12753                                ;VERIFY HEAD TAG DURING DATA COMMAND
12754  052712  012702  000400                MOV     #400,R2         ;INITIALIZE TRACK ADDRESS
12755  052716                        330$:
12756                                ;SET VOLUME VALID USING SUBROUTINE
12757  052716  004737  060526                JSR     PC,SETVV        ;GO SET VOLUME VALID
12758  052722  000402                        BR      340$            ;BRANCH TO 340$ IF NO ERROR
12759  052724  104000                        ERROR                   ;RETURN HERE IF ERROR
12760  052726  000550                        BR      400$
12761  052730                        340$:
12762                                ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12763
12764  052730  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12765
12766  052736  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
12767
12768  052744  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
12769
12770  052752  010260  000006                MOV     R2,RMDA(R0)     ;LOAD RMDA
12771
12772  052756  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
12773
12774  052764  012760  000071  000000        MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12775                                ;WAIT FOR RUN AND GO TO SET
12776  052772  012737  000310  001524        MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
12777  053000  004777  126522                JSR     PC,@CLOCK       ;START THE CLOCK
12778  053004                        350$:
12779
12780  053004  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12781  053012  042737  137777  001142        BIC     #^CRG,$BDDAT
12782  053020  001017                        BNE     360$
12783  053022  005737  001524                TST     WATCH
12784  053026  001366                        BNE     350$
12785  053030  004777  126474                JSR     PC,@STOP        ;STOP THE CLOCK
12786  053034  012737  040000  001140        MOV     #RG,$GDDAT
12787  053042  010037  001136                MOV     R0,$BDADR
12788  053046  062737  000024  001136        ADD     #RMMR1,$BDADR
12789  053054  104275                        ERROR   275             ;RUN AND GO NOT SET
12790  053056  000474                        BR      400$
12791  053060                        360$:
12792  053060  004777  126444                JSR     PC,@STOP        ;STOP THE CLOCK
12793                                ;STEP COMMAND SEQUENCER TO HEAD SEQUENCE, LOCATION 156 (17 CLOCKS)
12794  053064  012703  000021                MOV     #17.,R3
12795  053070                        370$:
12796
12797  053070  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12798
12799  053076  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
```

```
12800  053104  005303                          DEC     R3
12801  053106  001370                          BNE     370$
12802                                ;DROP AND RAISE ON CYLINDER TO RESET ON LATCH
12803
12804  053110  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
12805
12806  053116  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12807  053124  012703  053252                  MOV     #450$,R3        ;INITIALIZE TABLE POINTER
12808                                ;VERIFY TAG BUS ACCORDING TO TABLE AND TRACK ADDRESS IN R2
12809  053130                       375$:
12810
12811  053130  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
12812  053136  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
12813  053144  011337  001140                  MOV     (R3),$GDDAT
12814  053150  010204                          MOV     R2,R4           ;GENERATE EXPECTED TAG BUS
12815  053152  000304                          SWAB    R4
12816  053154  042704  177770                  BIC     #^C7,R4
12817  053160  050437  001140                  BIS     R4,$GDDAT
12818  053164  023737  001140  001142          CMP     $GDDAT,$BDDAT
12819  053172  001020                          BNE     390$
12820
12821  053174  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12822
12823  053202  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR!
12824                                ;ADVANCE TO NEXT TABLE ENTRY
12825  053210  062703  000002                  ADD     #2,R3
12826  053214  005713                          TST     (R3)
12827  053216  100401                          BMI     380$
12828  053220  000743                          BR      375$
12829  053222                       380$:
12830                                ;SHIFT TO NEXT TRACK ADDRESS-EXIT LOOP IF DONE
12831  053222  006302                          ASL     R2
12832  053224  020227  002000                  CMP     R2,#2000
12833  053230  101007                          BHI     400$
12834  053232  000631                          BR      330$
12835  053234                       390$:
12836                                ;ERROR ON TAG BUS DURING HEAD SEQUENCE
12837  053234  010037  001136                  MOV     R0,$BDADR
12838  053240  062737  000040  001136          ADD     #RMMR2,$BDADR
12839  053246  104276                          ERROR   276             ;INCORRECT TAG BUS
12840
12841  053250  000440               400$:      BR      500$            ;JUMP OVER TABLE
12842
12843  053252                       450$:
12844                                ;TABLE OF TAG BUS DURING HEAD SEQUENCE
12845  053252  002000                          .WORD   CH
12846  053254  002000                          .WORD   CH
12847  053256  022000                          .WORD   CH!TAG
12848  053260  022000                          .WORD   CH!TAG
12849  053262  022000                          .WORD   CH!TAG
12850  053264  022000                          .WORD   CH!TAG
12851  053266  022000                          .WORD   CH!TAG
12852  053270  022000                          .WORD   CH!TAG
12853  053272  002000                          .WORD   CH
12854  053274  002000                          .WORD   CH
12855  053276  001777                          .WORD   1777
```

```
12856  053300  001777                          .WORD   1777
12857  053302  001777                          .WORD   1777
12858  053304  001777                          .WORD   1777
12859  053306  001777                          .WORD   1777
12860  053310  001777                          .WORD   1777
12861  053312  001777                          .WORD   1777
12862  053314  001777                          .WORD   1777
12863  053316  001777                          .WORD   1777
12864  053320  001777                          .WORD   1777
12865  053322  001777                          .WORD   1777
12866  053324  001777                          .WORD   1777
12867  053326  001777                          .WORD   1777
12868  053330  001777                          .WORD   1777
12869  053332  001777                          .WORD   1777
12870  053334  001777                          .WORD   1777
12871  053336  001777                          .WORD   1777
12872  053340  001777                          .WORD   1777
12873  053342  001777                          .WORD   1777
12874  053344  001777                          .WORD   1777
12875  053346  001777                          .WORD   1777
12876
12877  053350  177777                          .WORD   -1              ;END OF TABLE
12878
12879  053352                          500$:                           ;END OF TEST
12880
12881                                  ;;**********************************************************************
12882                                  ;*TEST 120      DATA COMMAND TESTS (2)
12883
12884                                  ;;**********************************************************************
12885  053352  000004                  TST120: SCOPE
12886  053354  012737  000120  001226          MOV     #120,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
12887
12888  053362  000240                          NOP
12889  053364  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
12890  053372  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
12891  053400  012737  053414  001122          MOV     #T120,$LPADR  ;LOAD LOOP ON TEST ADDRESS
12892  053406  012737  053414  001124          MOV     #T120,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
12893  053414                          T120:
12894  053414  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
12895  053420  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
12896  053424  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
12897                                  ;;**********************************************************************
12898                                  ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT RESET
12899                                  ;SET VOLUME VALID USING SUBROUTINE
12900  053430  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12901  053434  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
12902  053436  104000                          ERROR                   ;RETURN HERE IF ERROR
12903  053440  000514                          BR      60$
12904  053442                          10$:
12905                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND DURING CYLINDER SEQUENCE
12906
12907  053442  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12908
12909  053450  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
12910
12911  053456  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
```

```
12912
12913  053464  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12914
12915  053472  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12916
12917  053500  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12918                                  ;WAIT FOR RUN AND GO TO SET
12919  053506  012737  000310  001524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
12920  053514  004777  126006                  JSR     PC,@CLOCK       ;START THE CLOCK
12921  053520                          20$:
12922
12923  053520  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12924  053526  042737  137777  001142          BIC     #^CRG,$BDDAT
12925  053534  001017                          BNE     30$
12926  053536  005737  001524                  TST     WATCH
12927  053542  001366                          BNE     20$
12928  053544  004777  125760                  JSR     PC,@STOP        ;STOP THE CLOCK
12929  053550  012737  040000  001140          MOV     #RG,$GDDAT
12930  053556  010037  001136                  MOV     R0,$BDADR
12931  053562  062737  000024  001136          ADD     #RMMR1,$BDADR
12932  053570  104275                          ERROR   275             ;RUN AND GO DIDNT SET
12933  053572  000437                          BR      60$
12934  053574                          30$:
12935  053574  004777  125730                  JSR     PC,@STOP        ;STOP THE CLOCK
12936                                  ;STEP COMMAND SEQUENCER AND VERIFY OPI SETS (19 CLOCKS)
12937  053600  012702  000023                  MOV     #19.,R2
12938  053604                          40$:
12939
12940  053604  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12941
12942  053612  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12943  053620  005302                          DEC     R2
12944  053622  001370                          BNE     40$
12945
12946  053624  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
12947  053632  042737  157777  001142          BIC     #^COPI,$BDDAT
12948  053640  001011                          BNE     50$             ;BRANCH IF OPI SET
12949  053642  012737  020000  001140          MOV     #OPE,$GDDAT
12950  053650  010037  001136                  MOV     R0,$BDADR
12951  053654  062737  000014  001136          ADD     #RMER1,$BDADR
12952  053662  104277                          ERROR   277             ;OPI NOT SET BY ON LATCH SET
12953  053664  012737  053672  001124  50$:    MOV     #60$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
12954                                  ;;*****************************************************************
12955                                  ;VERIFY DATA COMMAND ABORTS DURING SEEK WAIT LOOP
12956  053672                          60$:
12957                                  ;SET VOLUME VALID USING SUBROUTINE
12958  053672  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
12959  053676  000402                          BR      70$             ;BRANCH TO 70$ IF NO ERROR
12960  053700  104000                          ERROR                   ;RETURN HERE IF ERROR
12961  053702  000567                          BR      140$
12962  053704                          70$:
12963                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
12964
12965  053704  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
12966
12967  053712  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
```

```
12968
12969  053720  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
12970
12971  053726  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
12972
12973  053734  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
12974
12975  053742  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
12976                                  ;WAIT FOR RUN & GO TO SET
12977  053750  012737  000310  001524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
12978  053756  004777  125544                  JSR     PC,@CLOCK       ;START THE CLOCK
12979  053762                          80$:
12980
12981  053762  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
12982  053770  042737  137777  001142          BIC     #^CRG,$BDDAT
12983  053776  001017                          BNE     90$
12984  054000  005737  001524                  TST     WATCH
12985  054004  001366                          BNE     80$
12986  054006  004777  125516                  JSR     PC,@STOP        ;STOP THE CLOCK
12987  054012  012737  040000  001140          MOV     #RG,$GDDAT
12988  054020  010037  001136                  MOV     R0,$BDADR
12989  054024  062737  000024  001136          ADD     #RMMR1,$BDADR
12990  054032  104275                          ERROR   275             ;RUN AND GO DIDNT SET
12991  054034  000512                          BR      140$
12992  054036                          90$:
12993  054036  004777  125466                  JSR     PC,@STOP        ;STOP THE CLOCK
12994                                  ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
12995  054042  012702  000021                  MOV     #17.,R2
12996  054046                          100$:
12997
12998  054046  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
12999
13000  054054  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13001  054062  005302                          DEC     R2
13002  054064  001370                          BNE     100$
13003                                  ;DROP ON CYLINDER TO RESET LATCH
13004
13005  054066  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13006                                  ;MOVE COMMAND SEQUENCER TO SEEK WAIT LOOP (31 CLOCKS)
13007  054074  012702  000037                  MOV     #31.,R2
13008  054100                          110$:
13009
13010  054100  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
13011
13012  054106  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13013  054114  005302                          DEC     R2
13014  054116  001370                          BNE     110$
13015                                  ;STEP THROUGH SEEK WAIT LOOP (6 CLOCKS) 2 TIMES
13016  054120  012702  000006                  MOV     #6.,R2
13017  054124                          120$:
13018
13019  054124  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
13020
13021  054132  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13022  054140  005302                          DEC     R2
13023  054142  001370                          BNE     120$
```

```
13024                                   ;SET SEEK INCOMPLETE ERROR TO CAUSE ABORT
13025
13026  054144  012760  041201  000024           MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
13027                                   ;CLOCK THE SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13028  054152  012702  000002                   MOV     #2,R2
13029  054156                          130$:
13030
13031  054156  012760  141201  000024           MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
13032
13033  054164  012760  041201  000024           MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
13034  054172  005302                           DEC     R2
13035  054174  001370                           BNE     130$
13036
13037                                   ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13038                                   ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
13039  054176  012702  000020                   MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
13040  054202                          135$:
13041
13042  054202  012760  045501  000024           MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
13043
13044  054210  012760  041501  000024           MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
13045
13046  054216  016037  000024  001142           MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13047  054224  042737  157777  001142           BIC     #^CEBL,$BDDAT
13048  054232  001013                           BNE     140$
13049
13050  054234  005302                           DEC     R2
13051  054236  001361                           BNE     135$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13052  054240  012737  020000  001140           MOV     #EBL,$GDDAT
13053  054246  010037  001136                   MOV     R0,$BDADR
13054  054252  062737  000024  001136           ADD     #RMMR1,$BDADR
13055  054260  104300                           ERROR   300             ;EBL NOT SET DUE TO ABORT
13056  054262  012737  054270  001124  140$:    MOV     #150$,$LPERR            ;CHANGE LOOP ON ERROR ADDRESS
13057
13058                                   ;;*****************************************************************
13059                                   ;VERIFY DATA COMMAND ABORTS DURING OFFSET IF ON CYLINDER LATCH
13060                                   ;DOESNT RESET
13061  054270                          150$:
13062                                   ;SET VOLUME VALID USING SUBROUTINE
13063  054270  004737  060526                   JSR     PC,SETVV        ;GO SET VOLUME VALID
13064  054274  000402                           BR      160$            ;BRANCH TO 160$ IF NO ERROR
13065  054276  104000                           ERROR                   ;RETURN HERE IF ERROR
13066  054300  000536                           BR      220$
13067  054302                          160$:
13069                                   ;LOAD TRACK,SETCOR,CYLINDER ADDRESS
13069
13070  054302  012760  000000  000006           MOV     #0,RMDA(R0)     ;LOAD RMDA
13071
13072  054310  012760  000000  000034           MOV     #0,RMDC(R0)     ;LOAD RMDC
13073                                   ;SET OFFSET MODE USING SUBROUTINE
13074  054316  004737  060656                   JSR     PC,SETOM        ;GO SET OFFSET MODE
13075  054322  000402                           BR      170$            ;BRANCH TO 170$ IF NO ERROR
13076  054324  104000                           ERROR                   ;RETURN HERE IF ERROR
13077  054326  000523                           BR      220$
13078  054330                          170$:
13079                                   ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
```

```
13080
13081  054330  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13082
13083  054336  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
13084
13085  054344  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
13086
13087  054352  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
13088                                  ;WAIT FOR RUN AND GO TO SET
13089  054360  012737  000310  001524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
13090  054366  004777  125134                  JSR     PC,@CLOCK       ;START THE CLOCK
13091  054372                          180$:
13092
13093  054372  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13094  054400  042737  137777  001142          BIC     #^CRG,$BDDAT
13095  054406  001017                          BNE     190$
13096  054410  005737  001524                  TST     WATCH
13097  054414  001366                          BNE     180$
13098  054416  004777  125106                  JSR     PC,@STOP        ;STOP THE CLOCK
13099  054422  012737  040000  001140          MOV     #RG,$GDDAT
13100  054430  010037  001140                  MOV     R0,$GDDAT
13101  054434  062737  000024  001136          ADD     #RMMR1,$BDADR
13102  054442  104275                          ERROR   275             ;RUN AND GO NOT SET
13103  054444  000454                          BR      220$
13104  054446                          190$:
13105  054446  004777  125056                  JSR     PC,@STOP        ;STOP THE CLOCK
13106                                  ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
13107  054452  012702  000021                  MOV     #17.,R2
13108  054456                          200$:
13109
13110  054456  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13111
13112  054464  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13113  054472  005302                          DEC     R2
13114  054474  001370                          BNE     200$
13115                                  ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13116                                  ;AT LOCATION 166
13117
13118  054476  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13119
13120  054504  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
13121                                  ;MOVE SEQUENCER TO SET OPI AND EBL (39 CLOCKS)
13122  054512  012702  000047                  MOV     #39.,R2
13123  054516                          210$:
13124
13125  054516  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
13126
13127  054524  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
13128  054532  005302                          DEC     R2
13129  054534  001370                          BNE     210$
13130                                  ;VERIFY OPI IS SET
13131
13132  054536  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
13133  054544  042737  157777  001142          BIC     #^COPI,$BDDAT
13134  054552  001011                          BNE     220$
13135  054554  012737  020000  001140          MOV     #OPI,$GDDAT
```

```
13136  054562  010037  001136                MOV     R0,$BDADR
13137  054566  062737  000014  001136        ADD     #RMER1,$BDADR
13138  054574  104277                        ERROR   277             ;OPI NOT SET DURING OFFSET
13139  054576  012737  054604  001124  220$: MOV     #230$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
13140
13141                                  ;;*****************************************************************
13142                                  ;VERIFY DATA COMMAND ABORTS DURING OFFSET WAIT LOOP
13143
13144  054604                          230$:
13145                                  ;SET VOLUME VALID USING SUBROUTINE
13146  054604  004737  060526                JSR     PC,SETVV        ;GO SET VOLUME VALID
13147  054610  000403                        BR      240$            ;BRANCH TO 240$ IF NO ERROR
13148  054612  104000                        ERROR                   ;RETURN HERE IF ERROR
13149  054614  000137  055224                JMP     310$
13150  054620                          240$:
13151                                  ;LOAD SECTOR,TRACK AND CYLINDER ADDRESS
13152
13153  054620  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
13154
13155  054626  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
13156                                  ;SET OFFSET MODE USING SUBROUTINE
13157  054634  004737  060656                JSR     PC,SETOM        ;GO SET OFFSET MODE
13158  054640  000402                        BR      245$            ;BRANCH TO 245$ IF NO ERROR
13159  054642  104000                        ERROR                   ;RETURN HERE IF ERROR
13160  054644  000567                        BR      310$
13161                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13162  054646                          245$:
13163
13164  054646  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13165
13166  054654  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
13167
13168  054662  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
13169
13170  054670  012760  000071  000000        MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
13171                                  ;WAIT FOR RUN AND GO TO SET
13172  054676  012737  000310  001524        MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
13173  054704  004777  124616                JSR     PC,@CLOCK       ;START THE CLOCK
13174  054710                          250$:
13175
13176  054710  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13177  054716  042737  137777  001142        BIC     #^CRG,$BDDAT
13178  054724  001017                        BNE     260$
13179  054726  005737  001524                TST     WATCH
13180  054732  001366                        BNE     250$
13181  054734  004777  124570                JSR     PC,@STOP        ;STOP THE CLOCK
13182  054740  012737  040000  001140        MOV     #RG,$GDDAT
13183  054746  010037  001136                MOV     R0,$BDADR
13184  054752  062737  000024  001136        ADD     #RMMR1,$BDADR
13185  054760  104275                        ERROR   275             ;RUN AND GO NOT SET
13186  054762  000520                        BR      310$
13187  054764                          260$:
13188  054764  004777  124540                JSR     PC,@STOP        ;STOP THE CLOCK
13189                                  ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
13190  054770  012702  000021                MOV     #17.,R2
13191  054774                          270$:
```

```
13192
13193  054774  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13194
13195  055002  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13196  055010  005302                          DEC     R2
13197  055012  001370                          BNE     270$
13198                                  ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13199                                  ;AT LOCATION 166
13200
13201  055014  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13202
13203  055022  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13204                                  ;MOVE SEQUENCER TO LOCATION  174 (37 CLOCKS)
13205  055030  012702  000045                  MOV     #37.,R2
13206  055034                          280$:
13207
13208  055034  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ,LOAD RMMR1
13209
13210  055042  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13211  055050  005302                          DEC     R2
13212  055052  001370                          BNE     280$
13213                                  ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER RESET
13214
13215  055054  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13216                                  ;STEP SEQUENCER THROUGH OFFSET WAIT LOOP TWICE (7 CLOCKS)
13217  055062  012702  000007                  MOV     #7.,R2
13218  055066                          290$:
13219
13220  055066  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
13221
13222  055074  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13223  055102  005302                          DEC     R2
13224  055104  001370                          BNE     290$
13225                                  ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
13226
13227  055106  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
13228                                  ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13229  055114  012702  000002                  MOV     #2,R2
13230  055120                          300$:
13231
13232  055120  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
13233
13234  055126  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
13235  055134  005302                          DEC     R2
13236  055136  001370                          BNE     300$
13237
13238                                  ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13239                                  ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
13240  055140  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
13241  055144                          305$:
13242
13243  055144  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
13244
13245  055152  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
13246
13247  055160  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
```

```
13248  055166  042737  157777  001142          BIC     #↑CEBL,$BDDAT
13249  055174  001013                          BNE     310$
13250
13251  055176  005302                          DEC     R2
13252  055200  001361                          BNE     305$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13253  055202  012737  020000  001140          MOV     #EBL,$GDDAT
13254  055210  010037  001136                  MOV     R0,$BDADR
13255  055214  062737  000024  001136          ADD     #RMMR1,$BDADR
13256  055222  104271                          ERROR   271             ;EBL NOT SET DUE TO OFFSET WAIT ABORT
13257
13258  055224  012737  055232  001124  310$:   MOV     #320$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
13259
13260                                  ;;*****************************************************************
13261                                  ;VERIFY THAT DATA COMMAND ABORTS DURING SECTOR WAIT LOOP AT LOCATION 179
13262
13263  055232                          320$:
13264                                  ;SET VOLUME VALID USING SUBROUTINE
13265  055232  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
13266  055236  000403                          BR      330$            ;BRANCH TO 330$ IF NO ERROR
13267  055240  104000                          ERROR                   ;RETURN HERE IF ERROR
13268  055242  000137  055670                  JMP     420$
13269  055246                          330$:
13270                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13271
13272  055246  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13273
13274  055254  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
13275
13276  055262  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
13277
13278  055270  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
13279
13280  055276  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
13281
13282  055304  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
13283                                  ;WAIT FOR RUN AND GO TO SET
13284  055312  012737  000310  001524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
13285  055320  004777  124202                  JSR     PC,@CLOCK       ;START THE CLOCK
13286  055324                          340$:
13287
13288  055324  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13289  055332  042737  137777  001142          BIC     #↑CRG,$BDDAT
13290  055340  001017                          BNE     350$
13291  055342  005737  001524                  TST     WATCH
13292  055346  001366                          BNE     340$
13293  055350  004777  124154                  JSR     PC,@STOP        ;STOP THE CLOCK
13294  055354  012737  040000  001140          MOV     #RG,$GDDAT
13295  055362  010037  001136                  MOV     R0,$BDADR
13296  055366  062737  000024  001136          ADD     #RMMR1,$BDADR
13297  055374  104275                          ERROR   275             ;RUN AND GO NOT SET
13298  055376  000534                          BR      420$
13299  055400                          350$:
13300  055400  004777  124124                  JSR     PC,@STOP        ;STOP THE CLOCK
13301                                  ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
13302  055404  012702  000021                  MOV     #17.,R2
13303  055410                          360$:
```

```
13304
13305  055410  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13306
13307  055416  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13308  055424  005302                          DEC     R2
13309  055426  001370                          BNE     360$
13310                                  ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
13311                                  ;AT LOCATION 166
13312
13313  055430  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13314                                  ;MOVE SEQUENCER TO SECTOR WAIT (34 CLOCKS)
13315  055436  012702  000042                  MOV     #34.,R2
13316  055442                          370$:
13317
13318  055442  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13319
13320  055450  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13321  055456  005302                          DEC     R2
13322  055460  001370                          BNE     370$
13323                                  ;STEP THROUGH SECTOR WAIT LOOP TWICE AND VERIFY SEARCH IS ENABLED
13324                                  ;DURING THE LOOP (6 CLOCKS)
13325  055462  012702  000006                  MOV     #6.,R2
13326  055466                          380$:
13327
13328  055466  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13329
13330  055474  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13331
13332  055502  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13333  055510  042737  173777  001142          BIC     #^CESRC,$BDDAT
13334  055516  001403                          BEQ     390$
13335  055520  005302                          DEC     R2
13336  055522  001361                          BNE     380$
13337  055524  000412                          BR      400$
13338  055526  012737  004000  001140  390$:   MOV     #ESRC,$GDDAT
13339  055534  010037  001136                  MOV     R0,$BDADR
13340  055540  062737  000024  001136          ADD     #RMMR1,$BDADR
13341  055546  104301                          ERROR   301             ;SEARCH NOT ENABLED DURING DATA
13342  055550  000447                          BR      420$
13343  055552                          400$:
13344                                  ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
13345
13346  055552  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
13347                                  ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
13348  055560  012702  000002                  MOV     #2,R2
13349  055564                          410$:
13350
13351  055564  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)     ;LOAD RMMR1
13352
13353  055572  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
13354  055600  005302                          DEC     R2
13355  055602  001370                          BNE     410$
13356
13357                                  ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
13358                                  ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
13359  055604  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
```

```
13360  055610                          415$:
13361
13362  055610  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
13363
13364  055616  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
13365                                  ;VERIFY EBL IS SET
13366
13367  055624  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13368  055632  042737  157777  001142          BIC     #^CEBL,$BDDAT
13369  055640  001013                          BNE     420$
13370
13371  055642  005302                          DEC     R2
13372  055644  001361                          BNE     415$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
13373  055646  012737  020000  001140          MOV     #EBL,$GDDAT
13374  055654  010037  001136                  MOV     R0,$BDADR
13375  055660  062737  000024  001136          ADD     #RMMR1,$BDADR
13376  055666  104271                          ERROR   271             ;NO ABORT AT SECTOR WAIT
13377
13378  055670                          420$:                           ;END OF TEST
13379
13380                                  ;;*****************************************************************
13381                                  ;*TEST 121       DATA COMMAND TESTS(3)
13382
13383                                  ;;*****************************************************************
13384  055670  000004                  TST121: SCOPE
13385  055672  012737  000121  001226          MOV     #121,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
13386
13387  055700  000240                          NOP
13388  055702  012737  000024  001120          MOV     #20.,$ICNT      ;20 ITERATIONS
13389  055710  112737  000001  001131          MOVB    #1,$ERMAX       ;ONE ERROR ALLOWED
13390  055716  012737  055732  001122          MOV     #T121,$LPADR  ;LOAD LOOP ON TEST ADDRESS
13391  055724  012737  055732  001124          MOV     #T121,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
13392  055732                          T121:
13393  055732  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
13394  055736  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS OF UUT
13395  055742  013701  001456                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
13396                                  ;;*****************************************************************
13397                                  ;VERIFY THE TAG BUS DURING DATA COMMAND
13398
13399                                  ;;*****************************************************************
13400                                  ;FIRST PART USES OFFSET FORWARD
13401
13402                                  ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
13403  055746  112737  000035  001410          MOVB    #29.,RMDA0      ;SECTOR 29
13404  055754  112737  000004  001411          MOVB    #4,RMDA0+1      ;TRACK 4
13405  055762  012737  001466  001436          MOV     #822.,RMDC0     ;CYLINDER 822
13406  055770  012737  000200  001434          MOV     #OFD,RMOF0      ;FORWARD OFFSET
13407  055776  012737  000071  001402          MOV     #RD!GO,RMCS10   ;READ DATA
13408  056004  012737  177400  001404          MOV     #^C256.+1,RMWC0 ;WORD COUNT
13409  056012  012737  106652  001406          MOV     #BUFFER,RMBA0   ;BUFFER ADDRESS
13410
13411                                  ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
13412  056020  004737  056042                  JSR     PC,10$
13413
13414                                  ;;*****************************************************************
13415                                  ;SECOND PART USES OFFSET REVERSE
```

```
13416                                          ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
13417  056024  112737  000000  001434          MOVB    #0,RMOFO        ;REVERSE OFFSET
13418
13419                                          ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
13420  056032  004737  056042                  JSR     PC,10$
13421
13422  056036  000137  057050                  JMP     300$
13423
13424                                          ;;*****************************************************************
13425                                          ;SUBROUTINE USED DURING TEST
13426
13427  056042                                  10$:
13428                                          ;SET VOLUME VALID USING SUBROUTINE
13429  056042  004737  060526                  JSR     PC,SETVV        ;GO SET VOLUME VALID
13430  056046  000403                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
13431  056050  104000                          ERROR                   ;RETURN HERE IF ERROR
13432  056052  000137  056666                  JMP     160$
13433  056056                                  20$:
13434                                          ;LOAD TRACK, SECTOR AND CYLINDER ADDRESS, LOAD OFFSET
13435
13436  056056  013760  001410  000006          MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
13437
13438  056064  013760  001436  000034          MOV     RMDC0,RMDC(R0)  ;LOAD RMDC
13439
13440  056072  013760  001434  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
13441                                          ;SET OFFSET MODE USING SUBROUTINE
13442  056100  004737  060656                  JSR     PC,SETOM        ;GO SET OFFSET MODE
13443  056104  000403                          BR      30$             ;BRANCH TO 30$ IF NO ERROR
13444  056106  104000                          ERROR                   ;RETURN HERE IF ERROR
13445  056110  000137  056666                  JMP     160$
13446  056114                                  30$:
13447                                          ;LOAD BUFFER ADDRESS AND WORD COUNT
13448
13449  056114  013760  001404  000002          MOV     RMWC0,RMWC(R0)  ;LOAD RMWC
13450
13451  056122  013760  001406  000004          MOV     RMBA0,RMBA(R0)  ;LOAD RMBA
13452                                          ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
13453
13454  056130  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13455
13456  056136  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
13457
13458  056144  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
13459
13460  056152  013760  001402  000000          MOV     RMCS10,RMCS1(R0)        ;LOAD RMCS1
13461                                          ;WAIT FOR RUN AND GO TO SET
13462  056160  012737  000310  C01524          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
13463  056166  004777  123334                  JSR     PC,@CLOCK       ;START THE CLOCK
13464  056172                                  40$:
13465
13466  056172  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
13467  056200  042737  137777  001142          BIC     #^CRG,$BDDAT
13468  056206  001020                          BNE     50$
13469  056210  005737  001524                  TST     WATCH
13470  056214  001366                          BNE     40$
13471  056216  004777  123306                  JSR     PC,@STOP        ;STOP THE CLOCK
```

```
13472  056222  012737  040000  001140          MOV     #RG,$GDDAT
13473  056230  010037  001136                  MOV     R0,$BDADR
13474  056234  062737  000024  001136          ADD     #RMMR1,$BDADR
13475  056242  104275                          ERROR   275             ;RUN AND GO NOT SET
13476  056244  000137  056666                  JMP     160$
13477  056250                          50$:
13478  056250  004777  123254                  JSR     PC,@STOP        ;STOP THE CLOCK
13479  056254  012704  056670                  MOV     #200$,R4        ;R4 = TABLE POINTER
13480
13481                                  ;STEP SEQUENCER TO HEAD SEQUENCE AT LOCATION 156 (17 CLOCKS)
13482  056260  012705  000021                  MOV     #17.,R5         ;R5 = CLOCK COUNT
13483  056264                          60$:
13484
13485  056264  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
13486  056272  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
13487  056300  012437  001140                  MOV     (R4)+,$GDDAT
13488  056304  053737  001436  001140          BIS     RMDC0,$GDDAT    ;OR CYLINDER ADDRESS
13489  056312  023737  001140  001142          CMP     $GDDAT,$BDDAT
13490  056320  001011                          BNE     70$             ;BRANCH IF TAG BUS WRONG
13491
13492  056322  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13493
13494  056330  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13495  056336  005305                          DEC     R5
13496  056340  001351                          BNE     60$
13497  056342  000407                          BR      80$
13498  056344  010037  001136          70$:    MOV     R0,$BDADR
13499  056350  062737  000040  001136          ADD     #RMMR2,$BDADR
13500  056356  104276                          ERROR   276             ;INCORRECT TAG BUS DURING DATA
13501  056360  000542                          BR      160$
13502  056362                          80$:
                                       ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST AT
                                       ;LOCATION 166

13506  056362  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13507
13508  056370  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13509                                  ;STEP SEQUENCER TO END OF OFFSET AT LOCATION 174 (37 CLOCKS)
13510  056376  012705  000045                  MOV     #37.,R5         ;RELOAD CLOCK COUNT
13511  056402                          90$:
13512
13513  056402  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
13514  056410  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
13515  056416  011437  001140                  MOV     (R4),$GDDAT
13516  056422  032714  002000                  BIT     #CH,(R4)
13517  056426  001430                          BEQ     110$            ;BRANCH IF CONTROL/HEADER NOT ON
13518  056430  032714  004000                  BIT     #CC,(R4)
13519  056434  001416                          BEQ     100$            ;BRANCH IF HEADER TAG
                                       ;CONTROL TAG SHOULD BE ON-SETUP EXPECTED OFFSET
13521  056436  052737  000010  001140          BIS     #BB03,$GDDAT    ;ASSUME OFD IS NOT SET
13522  056444  032737  000200  001434          BIT     #OFD,RMOF0
13523  056452  001406                          BEQ     95$
13524  056454  042737  000010  001140          BIC     #BB03,$GDDAT    ;RESET BUS BIT 3 - DIRECTION IS REV
13525  056462  052737  000004  001140          BIS     #BB02,$GDDAT
13526  056470  000407                  95$:    BR      110$
13527
```

```
13528  056472                                  100$:
13529                                          ;HEADER TAG SHOULD BE ON - SETUP EXPECTED TRACK ADDRESS
13530  056472  013703  001410                  MOV     RMDA0,R3        ;GET TRACK
13531  056476  000303                          SWAB    R3
13532  056500  042703  177770                  BIC     #^C7,R3
13533  056504  050337  001140                  BIS     R3,$GDDAT
13534  056510                                  110$:
13535                                          ;COMPARE EXPECTED AND RECEIVED TAG BUS DATA
13536  056510  023737  001140  001142          CMP     $GDDAT,$BDDAT
13537  056516  001013                          BNE     120$
13538
13539  056520  012760  141401  000024          MOV     #L D!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13540
13541  056526  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13542  056534  062704  000002                  ADD     #2,R4           ;MOVE TABLE POINTER
13543  056540  005305                          DEC     R5              ;DECREMENT CLOCK COUNT
13544  056542  001317                          BNE     90$
13545  056544  000407                          BR      130$
13546  056546  010037  001136          120$:   MOV     R0,$BDADR
13547  056552  062737  000040  001136          ADD     #RMMR2,$BDADR
13548  056560  104276                          ERROR   276             ;INCORRECT TAG BUS DURING DATA
13549  056562  000441                          BR      160$
13550  056564                                  130$:
13551                                          ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER TO PASS TEST AT
13552                                          ;SEQUENCER LOCATION 175
13553
13554  056564  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
13555
13556  056572  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13557                                          ;STEP SEQUENCER TO SECTOR WAIT LOOP (8 CLOCKS)
13558  056600  012705  000010                  MOV     #8.,R5
13559  056604                                  140$:
13560
13561  056604  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
13562  056612  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
13563  056620  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;GOOD DATA SAME AS LAST CMP
13564  056626  001011                          BNE     150$            ;BRANCH IF ERROR
13565
13566  056630  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
13567
13568  056636  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
13569  056644  005305                          DEC     R5
13570  056646  001356                          BNE     140$
13571  056650  000406                          BR      160$
13572  056652  010037  001136          150$:   MOV     R0,$BDADR
13573  056656  062737  000040  001136          ADD     #RMMR2,$BDADR
13574  056664  104276                          ERROR   276             ;INCORRECT TAG BUS DURING DATA
13575
13576  056666  000207                  160$:   RTS     PC
13577
13578  056670                                  200$:
13579                                          ;TABLE OF TAG BUS CONTROL AND DATA VALUES
13580  056670  001777                          .WORD   1777            ;LOCATION 0
13581  056672  001777                          .WORD   1777            ;LOCATION 25
13582  056674  001777                          .WORD   1777            ;LOCATION 26
13583  056676  001777                          .WORD   1777            ;LOCATION 128
```

```
13584  056700  001777                    .WORD   1777           ;LOCATION 129
13585  056702  001777                    .WORD   1777           ;LOCATION 130
13586  056704  004000                    .WORD   CC             ;LOCATION 144
13587  056706  004000                    .WORD   CC             ;LOCATION 145
13588  056710  024000                    .WORD   CC!TAG         ;LOCATION 146
13589  056712  024000                    .WORD   CC!TAG         ;LOCATION 147
13590  056714  024000                    .WORD   CC!TAG         ;LOCATION 148
13591  056716  024000                    .WORD   CC!TAG         ;LOCATION 149
13592  056720  024000                    .WORD   CC!TAG         ;LOCATION 150
13593  056722  024000                    .WORD   CC!TAG         ;LOCATION 151
13594  056724  004000                    .WORD   CC             ;LOCATION 152
13595  056726  004000                    .WORD   CC             ;LOCATION 153
13596  056730  001777                    .WORD   1777           ;LOCATION 154
13597  056732  002000                    .WORD   CH             ;LOCATION 156
13598  056734  002000                    .WORD   CH             ;LOCATION 157
13599  056736  022000                    .WORD   CH!TAG         ;LOCATION 158
13600  056740  022000                    .WORD   CH!TAG         ;LOCATION 159
13601  056742  022000                    .WORD   CH!TAG         ;LOCATION 160
13602  056744  022000                    .WORD   CH!TAG         ;LOCATION 161
13603  056746  022000                    .WORD   CH!TAG         ;LOCATION 162
13604  056750  022000                    .WORD   CH!TAG         ;LOCATION 163
13605  056752  002000                    .WORD   CH             ;LOCATION 164
13606  056754  002000                    .WORD   CH             ;LOCATION 165
13607  056756  001777                    .WORD   1777           ;LOCATION 232
13608  056760  001777                    .WORD   1777           ;LOCATION 233
13609  056762  001777                    .WORD   1777           ;LOCATION 234
13610  056764  001777                    .WORD   1777           ;LOCATION 235
13611  056766  001777                    .WORD   1777           ;LOCATION 236
13612  056770  001777                    .WORD   1777           ;LOCATION 237
13613  056772  001777                    .WORD   1777           ;LOCATION 238
13614  056774  001777                    .WORD   1777           ;LOCATION 239
13615  056776  001777                    .WORD   1777           ;LOCATION 240
13616  057000  001777                    .WORD   1777           ;LOCATION 241
13617  057002  001777                    .WORD   1777           ;LOCATION 242
13618  057004  001777                    .WORD   1777           ;LOCATION 243
13619  057006  001777                    .WORD   1777           ;LOCATION 244
13620  057010  001777                    .WORD   1777           ;LOCATION 245
13621  057012  001777                    .WORD   1777           ;LOCATION 246
13622  057014  001777                    .WORD   1777           ;LOCATION 247
13623  057016  001777                    .WORD   1777           ;LOCATION 248
13624  057020  001777                    .WORD   1777           ;LOCATION 249
13625  057022  001777                    .WORD   1777           ;LOCATION 250
13626  057024  001777                    .WORD   1777           ;LOCATION 251
13627  057026  001777                    .WORD   1777           ;LOCATION 252
13628  057030  001777                    .WORD   1777           ;LOCATION 166
13629  057032  006000                    .WORD   CC!CH          ;LOCATION 169
13630  057034  006000                    .WORD   CC!CH          ;LOCATION 170
13631  057036  026000                    .WORD   CC!CH!TAG      ;LOCATION 171
13632  057040  026000                    .WORD   CC!CH!TAG      ;LOCATION 172
13633  057042  026000                    .WORD   CC!CH!TAG      ;LOCATION 173
13634  057044  026000                    .WORD   CC!CH!TAG      ;LOCATION 174
13635  057046  026000                    .WORD   CC!CH!TAG      ;LOCATION 175,ETC
13636
13637  057050                  300$:                            ;END OF TEST
```

```
13638  057050                          $EOSP:
13639  057050  000240                          NOP
13640  057052  013700  001456                  MOV     TSTQUE,R0
13641  057056  062700  000002                  ADD     #2,R0
13642  057062  010037  001456                  MOV     R0,TSTQUE
13643  057066  005710                          TST     (R0)
13644  057070  001402                          BEQ     1$
13645  057072  000137  006212                  JMP     READY
13646  057076  012737  001460  001456  1$:     MOV     #TSTQUE+2,TSTQUE
13647                                  .SBTTL  END OF PASS ROUTINE
13648
13649                                  ;;**********************************************************************
13650                                  ;*INCREMENT THE PASS NUMBER ($PASS)
13651                                  ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
13652                                  ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
13653                                  ;*IF THERES A MONITOR GO TO IT
13654                                  ;*IF THERE ISN'T JUMP TO READY
13655
13656  057104                          $EOP:
13657  057104  000240                          NOP
13658  057106  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
13659  057112  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
13660  057116  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
13661  057122  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
13662  057130  005327                          DEC     (PC)+           ;;LOOP?
13663  057132  000001                  $EOPCT: .WORD   1
13664  057134  003063                          BGT     $DOAGN          ;;YES
13665  057136  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
13666  057140  000001                  $ENDCT: .WORD   1
13667  057142  057132                          $EOPCT
13668  057144  104401  057152                  TYPE    ,65$            ;;TYPE ASCIZ STRING
13669  057150  000407                          BR      64$             ;;GET OVER THE ASCIZ
13670                                  ;;65$:  .ASCIZ  <12><15>/END PASS #/
13671  057170                          64$:
13672  057170  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
13673                                                                  ;;TYPE PASS NUMBER
13674  057174  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
13675  057176  104401  057204                  TYPE    ,67$            ;;TYPE ASCIZ STRING
13676  057202  000421                          BR      66$             ;;GET OVER THE ASCIZ
13677                                  ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
13678  057246                          66$:
13679  057246  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
13680                                                                  ;;TOTAL NUMBER OF ERRORS
13681  057252  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
13682  057254  104401  001217                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
13683  057260  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
13684  057264  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
13685  057270  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
13686  057272  000005                          RESET                   ;;CLEAR THE WORLD
13687  057274  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
13688  057276  000240                          NOP                     ;;SAVE ROOM
13689  057300  000240                          NOP                     ;;FOR
13690  057302  000240                          NOP                     ;;ACT11
13691  057304                          $DOAGN:
13692  057304  000137                          JMP     @(PC)+          ;;RETURN
13693  057306  006212                  $RTNAD: .WORD   READY
```

```
13694  057310     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
13695  057314                                 .EVEN
```

```
13696                                   .SBTTL  SUBROUTINES
13697
13698                                   ;;*****************************************************************
13699                                   .SBTTL  ERROR TYPEOUT ROUTINE
13700
13701                                   ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
13702                                   ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
13703                                   ;*
13704                                   ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
13705                                   ;*PRINTED ON THE FIRST LINE;
13706                                   ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
13707                                   ;*ONE OR MORE SUCCEEDING LINES;
13708                                   ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
13709                                   ;*ARE PRINTED AFTER THE ERROR MESSAGE.
13710
13711  057314                           ERRTYP:
13712  057314  104414                           SAVREG
13713  057316  032777  020000  121630            BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
13714  057324  001402                           BEQ     1$              ;NO!!
13715  057326  000137  060044                   JMP     21$             ;YES!!
13716                                   ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
13717  057332  104401  001217           1$:     TYPE    ,$CRLF
13718  057336  104401  060060                   TYPE    ,ERTY00         ;TYPE "UNT#"
13719  057342  013746  001234                   MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
13720                                                                   ;;TYPE UNIT NUMBER
13721  057346  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
13722  057350     003                           .BYTE   3               ;;TYPE 3 DIGIT(S)
13723  057351     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
13724  057352  005037  060050                   CLR     TSTNMB          ;LOAD TEST NUMBER FOR
13725  057356  013737  001226  060050           MOV     $TESTN,TSTNMB
13726  057364  104401  060066                   TYPE    ,ERTY01         ;TYPE "TST#"
13727  057370  013746  060050                   MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
13728                                                                   ;;TYPE TEST NUMBER
13729  057374  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
13730  057376     003                           .BYTE   3               ;;TYPE 3 DIGIT(S)
13731  057377     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
13732  057400  005037  060052                   CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
13733  057404  113737  001130  060052           MOVB    $ITEMB,ERRNMB   ;TYPEOUT
13734  057412  001406                           BEQ     2$              ;SKIP IF NO ERROR CALLED
13735  057414  104401  060076                   TYPE    ,ERTY02         ;TYPE "ERR#"
13736  057420  013746  060052                   MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
13737                                                                   ;;TYPE ERROR NUMBER
13738  057424  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
13739  057426     003                           .BYTE   3               ;;TYPE 3 DIGIT(S)
13740  057427     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
13741  057430  104401  060105           2$:     TYPE    ,ERTY03         ;TYPE "PC="
13742  057434  013746  001132                   MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
13743                                                                   ;;TYPE PROGRAM COUNTER
13744  057440  104403                           TYPOS                   ;;GO TYPE--OCTAL ASCII
13745  057442     006                           .BYTE   6               ;;TYPE 6 DIGIT(S)
13746  057443     001                           .BYTE   1               ;;TYPE LEADING ZEROS
13747                                   ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
13748  057444  005737  060052           3$:     TST     ERRNMB          ;WAS AN ERROR CALLED?
13749  057450  001575                           BEQ     21$             ;NO!!
13750  057452  104401  001217                   TYPE    ,$CRLF          ;YES-TYPE CRLF
13751  057456  105037  060056                   CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
13752  057462  105037  060057                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
13753  057466  013700  060052                  MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
13754  057472  006300                          ASL     R0              ;FOUR ENTRIES IN ERROR
13755  057474  006300                          ASL     R0              ;TABLE
13756  057476  006300                          ASL     R0
13757  057500  062700  001522                  ADD     #$ERRTB-8.,R0
13758  057504  011001                          MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
13759                                                                  ;TABLE
13760  057506  001507                          BEQ     12$             ;BRANCH IF NO ERROR MESSAGE
13761                                  ;TYPE THE ERROR MESSAGE
13762  057510  012102                  4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
13763  057512  001505                          BEQ     12$             ;BRANCH IF END OF MESSAGE
13764  057514  010237  057662                  MOV     R2,11$          ;LOAD ADDRESS OF STRING
13765  057520  005037  060054                  CLR     BOTADR          ;CLEAR BOT ADDRESS
13766  057524  112203                  5$:     MOVB    (R2)+,R3        ;END OF STRING??
13767  057526  001454                          BEQ     10$             ;YES!!
13768  057530  122703  000015                  CMPB    #CR,R3          ;CARRIAGE RETURN??
13769  057534  001003                          BNE     6$              ;NO!!
13770  057536  105037  060057                  CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
13771  057542  000770                          BR      5$              ;GET NEXT CHARACTER
13772  057544  122703  000012          6$:     CMPB    #LF,R3          ;LINE FEED??
13773  057550  001765                          BEQ     5$              ;YES-GET NEXT CHARACTER
13774  057552  122703  000011                  CMPB    #HT,R3          ;HORIZONTAL TAB??
13775  057556  001007                          BNE     8$              ;NO!!
13776  057560  105237  060057          7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
13777  057564  132737  000007  060057          BITB    #7,CHRCNT
13778  057572  001372                          BNE     7$
13779  057574  000407                          BR      9$
13780  057576  105237  060057          8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
13781  057602  122703  000040                  CMPB    #' ,R3          ;SPACE??
13782  057606  001002                          BNE     9$              ;NO!!
13783  057610  010237  060054                  MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
13784  057614  122737  000100  060057  9$:     CMPB    #64.,CHRCNT     ;END OF LINE??
13785  057622  103340                          BHIS    5$              ;NO!!
13786  057624  013704  060054                  MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
13787  057630  001007                          BNE     90$             ;BRANCH IF SPACE DETECTED
13788  057632  104401  001217                  TYPE    ,$CRLF          ;TYPE CRLF
13789  057636  105037  060057                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
13790  057642  013702  057662                  MOV     11$,R2          ;SET UP R2 FOR TESTING
13791  057646  000726                          BR      5$
13792  057650  105044                  90$:    CLRB    -(R4)           ;REPLACE SPACE
13793  057652  112737  177777  060056          MOVB    #-1,BOTFLG      ;SET BOT FLAG
13794  057660  104401                  10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
13795  057662  000000                  11$:    .WORD                   ;STRING ADDRESS GOES HERE
13796  057664  105737  060056                  TSTB    BOTFLG          ;WAS STRING TRUNCATED??
13797  057670  001707                          BEQ     4$              ;NO!!
13798  057672  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
13799  057676  105037  060056                  CLRB    BOTFLG          ;CLEAR BOT FLAG
13800  057702  105037  060057                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
13801  057706  013702  060054                  MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
13802  057712  010237  057662                  MOV     R2,11$          ;SETUP 11$ FOR TYPING
13803  057716  112742  000040                  MOVB    #' ,-(R2)       ;RESTORE SPACE
13804  057722  105722                          TSTB    (R2)+           ;RESTORE R2
13805  057724  000677                          BR      5$              ;TYPE REST OF STRING
13806  057726                          12$:
13807                                  ;TYPE ERROR HEADER AND ERROR DATA
```

```
13808  057726                          13$:
13809  057726  016001  000002                  MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
13810  057732  001444                          BEQ     21$             ;BRANCH IF NO HEADER
13811  057734  104401  001217                  TYPE    ,$CRLF          ;(ASSUME NO DATA)
13812  057740  016002  000004                  MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
13813  057744  016003  000006                  MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
13814  057750  012137  057760          14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
13815  057754  001433                          BEQ     21$             ;BRANCH IF END OF HEADERS
13816                                                                  ;(ASSUME END OF DATA)
13817  057756  104401                          TYPE
13818  057760  000000                  15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
13819  057762  104401  001217                  TYPE    ,$CRLF
13820  057766  005702                          TST     R2              ;DATA WITH HEADER??
13821  057770  001767                          BEQ     14$             ;NO!!
13822  057772  012204                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
13823  057774  012305                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
13824  057776  105725                  16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
13825  060000  100407                          BMI     18$             ;BINARY
13826  060002  001403                          BEQ     17$             ;OCTAL
13827  060004  013446                          MOV     @(R4)+,-(SP)    ;DECIMAL
13828  060006  104405                          TYPDS
13829  060010  000405                          BR      19$
13830  060012  013446                  17$:    MOV     @(R4)+,-(SP)
13831  060014  104402                          TYPOC
13832  060016  000402                          BR      19$
13833  060020  013446                  18$:    MOV     @(R4)+,-(SP)
13834  060022  104406                          TYPBN
13835  060024  005714                  19$:    TST     (R4)            ;MORE DATA??
13836  060026  001403                          BEQ     20$             ;NO!!
13837  060030  104401  060113                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
13838  060034  000760                          BR      16$             ;AND CONTINUE
13839  060036  104401  001217          20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
13840  060042  000742                          BR      14$             ;BEFORE NEXT HEADER
13841  060044  104415                  21$:    RESREG
13842  060046  000207                          RTS     PC
13843
13844  060050  000000                  TSTNMB: .WORD   0               ;TEST NUMBER
13845  060052  000000                  ERRNMB: .WORD   0               ;ERROR NUMBER
13846  060054  000000                  BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
13847  060056     000                  BOTFLG: .BYTE   0               ;BOT FLAG
13848  060057     000                  CHRCNT: .BYTE   0               ;CHARACTER COUNT
13849
13850  060060  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
13851  060066  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
13852  060074  000043
13853  060076  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
13854  060104     000
13855  060105     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
13856  060112     000
13857  060113     040  000040          ERTY04: .ASCIZ  @ @
13858                                  .EVEN
13859
```

```
13860                                   .SBTTL  CLOCK SUBROUTINES
13861
13862                                   ;ROUTINE TO SIZE FOR CLOCKS (KW11-L OR KW11-P)
13863                                   ;;*****************************************************************
13864  060116  000240                   SIZCLK: NOP
13865  060120  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
13866  060124  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
13867  060130  012737  060214  000004           MOV     #10$,ERRVEC     ;LOAD 04 TRAP VECTORS
13868  060136  012737  000300  000006           MOV     #PR6,ERRVEC+2
13869
13870                                   ;SEE IF A KW11-P CLOCK IS PRESENT - GO TO 10$ IF NOT PRESENT
13871  060144  005777  121332                   TST     @$LPCSR         ;TEST FOR P CLOCK
13872  060150  012737  060356  001526           MOV     #PCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
13873  060156  012737  060500  001530           MOV     #PSTOP,STOP     ;LOAD STOP ADDRESS
13874  060164  012777  060444  121314           MOV     #PCOUNT,@$LPVEC ;LOAD P CLOCK INTERRUPT VECTOR
13875  060172  012777  000300  121310           MOV     #PR6,@$LPVEC+2
13876  060200  013777  001516  121306           MOV     $LLVEC+2,@$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
13877  060206  005077  121304                   CLR     @$LLVEC+2
13878  060212  000454                           BR      30$
13879  060214  012716  060222           10$:    MOV     #15$,(SP)       ;DUMMY RTI ADDRESS
13880  060220  000002                           RTI                     ;RESTORE PRIORITY
13881  060222                           15$:
13882                                   ;NO P CLOCK-SEE IF L CLOCK IS PRESENT-GO TO 20$ IF NOT PRESET
13883  060222  012737  060300  000004           MOV     #20$,ERRVEC     ;CHANGE 04 TRAP VECTOR
13884  060230  005777  121256                   TST     @$LLCSR         ;TEST FOR L CLOCK
13885  060234  012737  060374  001526           MOV     #LCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
13886  060242  012737  060506  001530           MOV     #LSTOP,STOP     ;LOAD STOP ADDRESS
13887  060250  012777  060444  121236           MOV     #LCOUNT,@$LLVEC ;LOAD L CLOCK INTERRUPT VECTOR
13888  060256  012777  000300  121232           MOV     #PR6,@$LLVEC+2
13889  060264  013777  001510  121214           MOV     $LPVEC+2,@$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
13890  060272  005077  121212                   CLR     @$LPVEC+2
13891  060276  000422                           BR      30$
13892  060300  012716  060306           20$:    MOV     #25$,(SP)       ;DUMMY RTI ADDRESS
13893  060304  000002                           RTI                     ;RESTORE PRIORITY
13894  060306                           25$:
13895                                   ;NO CLOCK AVAILABLE - AUGMENT RETURN ADDRESS
13896  060306  005037  001526                   CLR     CLOCK           ;CLEAR SUBROUTINE ADDRESS
13897  060312  012737  001510  001506           MOV     #$LPVEC+2,$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
13898  060320  005037  001510                   CLR     $LPVEC+2
13899  060324  012737  001516  001514           MOV     #$LLVEC+2,$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
13900  060332  005037  001516                   CLR     $LLVEC+2
13901  060336  062766  000002  000004           ADD     #2,4(SP)        ;CHANGE RETURN ADDRESS
13902
13903  060344                           30$:
13904  060344  012637  000006                   MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
13905  060350  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
13906  060354  000207                           RTS     PC
13907                                   ;ROUTINES TO START THE CLOCK (KW11-L OR KW11-P)
13908                                   ;;*****************************************************************
13909
13910  060356  012777  177777  121120   PCLOCK: MOV     #-1,@$LPCSB     ;LOAD COUNT SET BUFFER
13911  060364  012777  000135  121110           MOV     #135,@$LPCSR    ;LOAD CONTROL REGISTER
13912  060372  000403                           BR      PLCLK           ;GO TO COMMON CODE
13913
13914  060374  012777  000100  121110   LCLOCK: MOV     #100,@$LLCSR    ;LOAD CONTROL REGISTER
13915
```

```
13916  060402  005037  001522          PLCLK:  CLR     TIME            ;CLEAR TIMER COUNT
13917  060406  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
13918  060410  012605                          MOV     (SP)+,R5        ;;SAVE THE PSW IN R5
13919  060412  010537  001520                  MOV     R5,$PSW         ;SAVE PRIORITY
13920  060416  042705  177437                  BIC     #^CPR7,R5       ;MASK X
13921  060422  022705  000300                  CMP     #PR6,R5         ;IS PRIORITY TOO HIGH??
13922  060426  101005                          BHI     40$             ;NO!!
13923  060430  012746  000240                  MOV     #PR5,-(SP)      ;;PUT NEW PS ON STACK
13924  060434  012746  060442                  MOV     #30$,-(SP)      ;;PUT NEW PC ON  STACK
13925  060440  000002                          RTI                     ;;POP NEW PC AND PS
13926  060442                          30$:
13927  060442  000207                  40$:    RTS     PC
13928
13929                                  ;ROUTINES TO HANDLE CLOCK INTERRUPTS (KW11-L OR KW11-P)
13930
13931  060444                          PCOUNT:
13932  060444                          LCOUNT:
13933  060444  062737  000021  001522          ADD     #17.,TIME       ;ADD 17MS TO ELAPSED TIME
13934  060452  103003                          BCC     10$             ;BRANCH IF NO OVERFLOW
13935  060454  012737  177777  001522          MOV     #-1,TIME        ;RESTORE MAXIMUM COUNT
13936  060462  162737  000021  001524  10$:    SUB     #17.,WATCH      ;DECREMENT REMAINING TIME
13937  060470  100002                          BPL     20$             ;BRANCH IF POSITIVE
13938  060472  005037  001524                  CLR     WATCH           ;CLEAR REMAINING TIME
13939  060476  000002                  20$:    RTI                     ;RETURN TO USER
13940
13941                                  ;ROUTINES TO STOP THE CLOCK (KW11-L OR KW11-P)
13942
13943  060500  005077  120776          PSTOP:  CLR     @$LPCSR         ;STOP P CLOCK
13944  060504  000402                          BR      PLSTP           ;GO TO COMMON STOP CODE
13945
13946  060506  005077  121000          LSTOP:  CLR     @$LLCSR         ;STOP L CLOCK
13947
13948  060512                          PLSTP:
13949  060512  013746  001520                  MOV     $PSW,-(SP)      ;;PUT NEW PS ON STACK
13950  060516  012746  060524                  MOV     #10$,-(SP)      ;;PUT NEW PC ON  STACK
13951  060522  000002                          RTI                     ;;POP NEW PC AND PS
13952  060524                          10$:
13953  060524  000207                          RTS     PC
13954
13955                                  .SBTTL  SET VOLUME VALID SUBROUTINE
13956
13957                                  ;THIS SUBROUTINE INITIALIZES THE SUBSYSTEM AND SETS VOLUME VALID,
13958                                  ;RETURNING WITH THE DRIVE STILL IN DIAGNOSTIC MODE.  THE SUBROUTINE
13959                                  ;RETURNS TO THE WORD FOLLOWING THE CALL, EXCEPT WHEN AN ERROR IS
13960                                  ;DETECTED, IN WHICH CASE IT RETURNS TO THE SECOND WORD FOLLOWING THE
13961                                  ;CALL.
13962                                  ;
13963                                  ;CALL:  JSR     PC,SETVV        JUMP TO SUBROUTINE
13964                                  ;       BR      ??              RETURN HERE IF NO ERROR
13965                                  ;       ERROR                   RETURN HERE IF ERROR
13966
13967  060526                          SETVV:
13968  060526  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR THE MASSBUS
13969  060534  111160  000010                  MOVB    (R1),RMCS2(R0)  ;SELECT UNIT
13970
13971  060540  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
```

```
13972
13973  060546  012760  001001  000024          MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
13974
13975  060554  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
13976
13977  060562  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
13978
13979  060570  012760  000023  000000          MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
13980
13981  060576  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
13982  060604  042737  177677  001142          BIC     #^CVV,$BDDAT
13983  060612  001020                          BNE     10$             ;BRANCH IF VOLUME VALID SET
13984  060614  010037  001136                  MOV     R0,$BDADR       ;SETUP FOR ERROR MSG
13985  060620  062737  000012  001136          ADD     #RMDS,$BDADR
13986  060626  012737  000100  001140          MOV     #VV,$GDDAT
13987  060634  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
13988  060640  112776  000170  000000          MOVB    #170,@(SP)      ;WRITE ERROR NUMBER
13989  060646  012737  000022  001174          MOV     #PACACK,$TMP0
13990  060654  000207                  10$:    RTS     PC              ;RETURN
13991
13992                                  .SBTTL  SET OFFSET MODE SUBROUTINE
13993
13994                                  ;THIS SUBROUTINE EXECUTES AN OFFSET COMMAND AND VERIFIES THAT OFFSET
13995                                  ;MODE SETS.  THE DRIVE SHOULD BE IN DIAGNOSTIC MODE WHEN CALLING THE
13996                                  ;SUBROUTINE, WHICH WILL LEAVE DMD ON.  THE SUBROUTINE RETURNS TO THE
13997                                  ;WORD FOLLOWING THE CALL UNLESS THERE IS AN ERROR, IN WHICH CASE IT
13998                                  ;RETURNS TO THE SECOND WORD FOLLOWING THE CALL
13999                                  ;
14000                                  ;CALL:  JSR     PC,SETOM        JUMP TO SUBROUTINE
14001                                  ;       BR      ??              RETURN HERE IF NO ERROR
14002                                  ;       ERROR                   RETURN HERE IF ERROR
14003
14004  060656                  SETOM:
14005
14006  060656  012760  001001  000024          MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
14007
14008  060664  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
14009
14010  060672  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
14011
14012  060700  012760  000015  000000          MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
14013
14014  060706  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
14015  060714  042737  177776  001142          BIC     #^COM,$BDDAT
14016  060722  001015                          BNE     10$             ;BRANCH IF OFFSET ON
14017  060724  012737  000001  001140          MOV     #OM,$GDDAT
14018  060732  010037  001136                  MOV     R0,$BDADR
14019  060736  062737  000012  001136          ADD     #RMDS,$BDADR
14020  060744  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
14021  060750  112776  000200  000000          MOVB    #200,@(SP)      ;WRITE ERROR NUMBER
14022  060756                  10$:
14023  060756  000207                  RTS     PC              ;RETURN TO USER
14024
```

```
                                .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

060760                          $SAVREG:
060760  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
060762  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
060764  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
060766  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
060770  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
060772  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
060774  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
061000  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
061004  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
061010  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
061014  000002                          RTI
                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
061016                          $RESREG:
061016  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
061022  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
061026  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
061032  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
061036  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
061040  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
061042  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
061044  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
061046  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
061050  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
061052  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

061054  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
061056  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
14081  061062  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
14082  061064  112737  000060  061126  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
14083  061072  006101                          ROL     R1              ;;GET THIS BIT
14084  061074  001406                          BEQ     2$              ;;DONE?
14085  061076  105537  061126                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
14086  061102  104401  061126                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
14087  061106  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
14088  061110  000765                          BR      1$              ;;GO DO THE NEXT BIT
14089  061112  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
14090  061114  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
14091  061122  012616                          MOV     (SP)+,(SP)
14092  061124  000002                          RTI                     ;;RETURN TO USER
14093  061126     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
14094                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
14095
14096                                  ;;*****************************************************************
14097                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
14098                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
14099                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
14100                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
14101                                  ;*REPLACED WITH SPACES.
14102                                  ;*CALL:
14103                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
14104                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
14105
14106  061130                          $TYPDS:
14107  061130  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
14108  061132  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
14109  061134  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
14110  061136  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
14111  061140  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
14112  061142  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
14113  061146  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
14114  061152  100004                          BPL     1$              ;;BR IF INPUT IS POS.
14115  061154  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
14116  061156  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
14117  061164  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
14118  061166  012703  061344                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
14119  061172  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
14120  061176  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
14121  061200  016001  061334                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
14122  061204  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
14123  061206  002402                          BLT     4$              ;;BR IF DONE
14124  061210  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
14125  061212  000774                          BR      3$
14126  061214  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
14127  061216  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
14128  061220  001002                          BNE     5$              ;;FALL THROUGH IF 0
14129  061222  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
14130  061224  100407                          BMI     7$              ;;BR IF YES
14131  061226  106316                  5$:     ASLB    (SP)            ;;MSD?
14132  061230  103003                          BCC     6$              ;;BR IF NO
14133  061232  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
14134  061240  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
14135  061244  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
14136  061250  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
14137  061252  005720                          TST     (R0)+           ;;JUST INCREMENTING
14138  061254  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
14139  061260  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
14140  061262  003002                          BGT     8$              ;;GO TO EXIT
14141  061264  010502                          MOV     R5,R2           ;;GET THE LSD
14142  061266  000764                          BR      6$              ;;GO CHANGE TO ASCII
14143  061270  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
14144  061272  100003                          BPL     9$              ;;BR IF NO
14145  061274  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
14146  061302  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
14147  061304  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
14148  061306  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
14149  061310  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
14150  061312  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
14151  061314  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
14152  061316  104401  061344                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
14153  061322  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
14154  061330  012616                          MOV     (SP)+,(SP)
14155  061332  000002                          RTI                     ;;RETURN TO USER
14156  061334  023420                  $DTBL:  10000.
14157  061336  001750                          1000.
14158  061340  000144                          100.
14159  061342  000012                          10.
14160  061344  000004                  $DBLK:  .BLKW   4
14161                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
14162
14163                                  ;;**************************************************************
14164                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
14165                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
14166                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
14167                                  ;*CALL:
14168                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14169                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
14170                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
14171                                  ;*      .BYTE   M               ;;M=1 OR 0
14172                                  ;*                                      ;;1=TYPE LEADING ZEROS
14173                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
14174                                  ;*
14175                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
14176                                  ;*$TYPOS OR $TYPOC
14177                                  ;*CALL:
14178                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14179                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
14180                                  ;*
14181                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
14182                                  ;*CALL:
14183                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14184                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
14185
14186  061354  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
14187  061360  116637  000001  061577          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
14188  061366  112637  061601                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
14189  061372  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
14190  061376  000406                          BR      $TYPON
14191  061400  112737  000001  061577  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
14192  061406  112737  000006  061601          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
14193  061414  112737  000005  061576  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
14194  061422  010346                          MOV     R3,-(SP)        ;;SAVE R3
14195  061424  010446                          MOV     R4,-(SP)        ;;SAVE R4
14196  061426  010546                          MOV     R5,-(SP)        ;;SAVE R5
14197  061430  113704  061601                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
14198  061434  005404                          NEG     R4
14199  061436  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
14200  061442  110437  061600                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
14201  061446  113704  061577                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
14202  061452  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
14203  061456  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
14204  061460  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
14205  061462  000404                          BR      3$              ;;GO DO MSB
14206  061464  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
14207  061466  006105                          ROL     R5
14208  061470  006105                          ROL     R5
14209  061472  010503                          MOV     R5,R3
14210  061474  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
14211  061476  105337  061600                  DECB    $OMODE          ;;TYPE THIS DIGIT?
14212  061502  100016                          BPL     7$              ;;BR IF NO
14213  061504  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
14214  061510  001002                          BNE     4$              ;;TEST FOR 0
14215  061512  005704                          TST     R4              ;;SUPPRESS THIS 0?
14216  061514  001403                          BEQ     5$              ;;BR IF YES
14217  061516  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
14218  061520  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
14219  061524  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
14220  061530  110337  061574                  MOVB    R3,8$           ;;SAVE FOR TYPING
14221  061534  104401  061574                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
14222  061540  105337  061576          7$:     DECB    $OCNT           ;;COUNT BY 1
14223  061544  003347                          BGT     2$              ;;BR IF MORE TO DO
14224  061546  002402                          BLT     6$              ;;BR IF DONE
14225  061550  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
14226  061552  000744                          BR      2$              ;;GO DO THE LAST DIGIT
14227  061554  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
14228  061556  012604                          MOV     (SP)+,R4        ;;RESTORE R4
14229  061560  012603                          MOV     (SP)+,R3        ;;RESTORE R3
14230  061562  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
14231  061570  012616                          MOV     (SP)+,(SP)
14232  061572  000002                          RTI                     ;;RETURN
14233  061574  000                     8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
14234  061575  000                             .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
14235  061576  000                     $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
14236  061577  000                     $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
14237  061600  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
14238                                  .SBTTL  TYPE ROUTINE
14239
14240                                  ;;******************************************************************
14241                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
14242                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
14243                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
14244                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
14245                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
14246                                  ;*
14247                                  ;*CALL:
14248                                  ;*1) USING A TRAP INSTRUCTION
```

```
14249                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
14250                                   ;*OR
14251                                   ;*      TYPE
14252                                   ;*      MESADR
14253                                   ;*
14254
14255  061602  105737  001173           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
14256  061606  100002                           BPL     1$              ;;BR IF YES
14257  061610  000000                           HALT                    ;;HALT HERE IF NO TERMINAL
14258  061612  000430                           BR      3$              ;;LEAVE
14259  061614  010046                   1$:     MOV     R0,-(SP)        ;;SAVE R0
14260  061616  017600  000002                   MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
14261  061622  122737  000001  001242           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
14262  061630  001011                           BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
14263  061632  132737  000100  001243           BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
14264  061640  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
14265  061642  010037  061652                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
14266  061646  004737  064672                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
14267  061652  000000                   61$:    .WORD   0               ;;MESSAGE ADDRESS
14268  061654  132737  000040  001243   62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
14269  061662  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
14270  061664  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
14271  061666  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
14272  061670  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
14273  061672  012600                   60$:    MOV     (SP)+,R0        ;;RESTORE R0
14274  061674  062716  000002           3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
14275  061700  000002                           RTI                     ;;RETURN
14276  061702  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
14277  061706  001430                           BEQ     8$
14278  061710  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
14279  061714  001006                           BNE     5$
14280  061716  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
14281  061720  104401                           TYPE                    ;;TYPE A CR AND LF
14282  061722  001217                           $CRLF
14283  061724  105037  062060                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
14284  061730  000755                           BR      2$              ;;GET NEXT CHARACTER
14285  061732  004737  062014           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
14286  061736  123726  001172           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
14287  061742  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
14288  061744  013746  001170                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
14289                                           ;;AND THE NULL CHAR.
14290  061750  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
14291  061754  002770                           BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
14292  061756  004737  062014                   JSR     PC,$TYPEC       ;;GO TYPE A NULL
14293  061762  105337  062060                   DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
14294  061766  000770                           BR      7$              ;;LOOP
14295
14296                                   ;HORIZONTAL TAB PROCESSOR
14297
14298  061770  112716  000040           8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
14299  061774  004737  062014           9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
14300  062000  132737  000007  062060           BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
14301  062006  001372                           BNE     9$              ;;TAB STOP
14302  062010  005726                           TST     (SP)+           ;;POP SPACE OFF STACK
14303  062012  000724                           BR      2$              ;;GET NEXT CHARACTER
14304  062014  105777  117144           $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
14305  062020  100375                          BPL     $TYPEC
14306  062022  116677  000002  117136          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
14307  062030  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
14308  062036  001003                          BNE     1$              ;;BRANCH IF NO
14309  062040  105037  062060                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
14310  062044  000406                          BR      $TYPEX          ;;EXIT
14311  062046  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
14312  062054  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
14313  062056  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
14314  062060  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
14315  062062  000207                  $TYPEX: RTS     PC
14316
14317                                  .SBTTL  SCOPE HANDLER ROUTINE
14318
14319                                  ;;**************************************************************
14320                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
14321                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
14322                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
14323                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
14324                                  ;*SW14=1        LOOP ON TEST
14325                                  ;*SW11=1        INHIBIT ITERATIONS
14326                                  ;*SW09=1        LOOP ON ERROR
14327                                  ;*SW08=1        LOOP ON TEST IN SWR<7:0>
14328                                  ;*CALL
14329                                  ;*      SCOPE           ;;SCOPE=IOT
14330
14331  062064                          $SCOPE:
14332  062064  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
14333  062066  032777  040000  117060  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
14334  062074  001131                          BNE     $OVER           ;;YES IF SW14=1
14335                                  ;#####START OF CODE FOR THE XOR TESTER#####
14336  062076  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
14337                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
14338  062100  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
14339  062104  012737  062124  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
14340  062112  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
14341  062116  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
14342  062122  000500                          BR      $SVLAD          ;;GO TO THE NEXT TEST
14343  062124  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
14344  062126  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
14345  062132  000440                          BR      7$              ;;LOOP ON THE PRESENT TEST
14346  062134                          6$:;#####END OF CODE FOR THE XOR TESTER#####
14347  062134  032777  000400  117012          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
14348  062142  001421                          BEQ     2$              ;;BR IF NO
14349  062144  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
14350  062146  117716  117002                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
14351  062152  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
14352  062154  022716  000121                  CMP     #121,(SP)       ;;CHECK THE NUMBER IN THE SWR
14353  062160  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
14354  062162  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
14355  062166  005316                          DEC     (SP)            ;;BACKUP BY ONE
14356  062170  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
14357  062172  062716  062376                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
14358  062176  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
14359  062202  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
14360  062204  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
```

```
14361  062206  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
14362  062212  001421                          BEQ     3$              ;;BR IF NO
14363  062214  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
14364  062222  101015                          BHI     3$              ;;BR IF NO
14365  062224  032777  001000  116722          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
14366  062232  001404                          BEQ     4$              ;;BR IF NO
14367  062234  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
14368  062242  000446                          BR      $OVER
14369  062244  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
14370  062250  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
14371  062254  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
14372  062256  032777  004000  116670  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
14373  062264  001011                          BNE     1$              ;;BR IF YES
14374  062266  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
14375  062272  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
14376  062274  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
14377  062300  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
14378  062306  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
14379  062310  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
14380  062316  013737  062374  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
14381  062324  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
14382  062330  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
14383  062336  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
14384  062342  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
14385  062346  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
14386  062352  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
14387  062360  013777  001116  116570  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
14388  062366  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
14389  062372  000002                          RTI                     ;;FIXES PS
14390  062374  000012          $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
14391  062376                  $SW08TBL:
14392  062376  006230                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
14393  062400  006604                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
14394  062402  007024                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
14395  062404  007240                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
14396  062406  007426                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
14397  062410  010564                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
14398  062412  011744                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
14399  062414  012104                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
14400  062416  012226                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
14401  062420  012524                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
14402  062422  013136                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
14403  062424  013764                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
14404  062426  014520                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
14405  062430  014656                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
14406  062432  015232                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
14407  062434  015372                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
14408  062436  015714                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
14409  062440  016070                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
14410  062442  016254                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
14411  062444  016564                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
14412  062446  017122                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
14413  062450  017604                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
14414  062452  017744                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
14415  062454  020270                          .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
14416  062456  020642                          .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
```

```
14417  062460  021170                .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
14420  062462  021744                .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
14421  062464  022314                .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
14422  062466  022644                .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
14423  062470  023224                .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
14424  062472  023612                .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
14425  062474  024332                .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
14426  062476  024662                .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
14427  062500  025124                .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
14430  062502  025564                .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
14431  062504  025772                .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
14432  062506  026410                .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
14433  062510  027156                .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
14434  062512  027510                .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
14435  062514  030150                .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
14436  062516  030442                .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
14437  062520  030706                .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
14440  062522  031576                .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
14441  062524  032106                .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
14442  062526  032340                .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
14443  062530  032602                .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
14444  062532  033034                .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
14445  062534  033252                .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
14446  062536  033526                .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
14447  062540  033724                .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
14450  062542  034200                .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
14451  062544  034344                .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
14452  062546  034546                .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
14453  062550  035034                .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
14454  062552  035302                .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
14455  062554  035560                .WORD   TST70+2         ;;STARTING ADDRESS OF TEST 70
14456  062556  036046                .WORD   TST71+2         ;;STARTING ADDRESS OF TEST 71
14457  062560  036310                .WORD   TST72+2         ;;STARTING ADDRESS OF TEST 72
14460  062562  036552                .WORD   TST73+2         ;;STARTING ADDRESS OF TEST 73
14461  062564  037036                .WORD   TST74+2         ;;STARTING ADDRESS OF TEST 74
14462  062566  037202                .WORD   TST75+2         ;;STARTING ADDRESS OF TEST 75
14463  062570  037364                .WORD   TST76+2         ;;STARTING ADDRESS OF TEST 76
14464  062572  037544                .WORD   TST77+2         ;;STARTING ADDRESS OF TEST 77
14465  062574  037726                .WORD   TST100+2              ;;STARTING ADDRESS OF TEST 100
14466  062576  040110                .WORD   TST101+2              ;;STARTING ADDRESS OF TEST 101
14467  062600  040330                .WORD   TST102+2              ;;STARTING ADDRESS OF TEST 102
14470  062602  040570                .WORD   TST103+2              ;;STARTING ADDRESS OF TEST 103
14471  062604  040754                .WORD   TST104+2              ;;STARTING ADDRESS OF TEST 104
14472  062606  041216                .WORD   TST105+2              ;;STARTING ADDRESS OF TEST 105
14473  062610  041404                .WORD   TST106+2              ;;STARTING ADDRESS OF TEST 106
14474  062612  041636                .WORD   TST107+2              ;;STARTING ADDRESS OF TEST 107
14475  062614  042064                .WORD   TST110+2              ;;STARTING ADDRESS OF TEST 110
14476  062616  042372                .WORD   TST111+2              ;;STARTING ADDRESS OF TEST 111
14477  062620  042734                .WORD   TST112+2              ;;STARTING ADDRESS OF TEST 112
14500  062622  043206                .WORD   TST113+2              ;;STARTING ADDRESS OF TEST 113
14501  062624  044630                .WORD   TST114+2              ;;STARTING ADDRESS OF TEST 114
14502  062626  046406                .WORD   TST115+2              ;;STARTING ADDRESS OF TEST 115
14503  062630  050572                .WORD   TST116+2              ;;STARTING ADDRESS OF TEST 116
14504  062632  051202                .WORD   TST117+2              ;;STARTING ADDRESS OF TEST 117
14505  062634  053354                .WORD   TST120+2              ;;STARTING ADDRESS OF TEST 120
14506  062636  055672                .WORD   TST121+2              ;;STARTING ADDRESS OF TEST 121
```

```
14473                                   .SBTTL  ERROR HANDLER ROUTINE
14474
14475                                   ;;**********************************************************************
14476                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
14477                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
14478                                   ;*AND GO TO ERRTYP ON ERROR
14479                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
14480                                   ;*SW15=1        HALT ON ERROR
14481                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
14482                                   ;*SW10=1        BELL ON ERROR
14483                                   ;*SW09=1        LOOP ON ERROR
14484                                   ;*CALL
14485                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
14486
14487  062640                           $ERROR:
14488  062640  104410                           CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
14489  062642  105237  001117           7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
14490  062646  001775                           BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
14491  062650  013777  001116  116300           MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
14492  062656  032777  002000  116270           BIT     #BIT10,@SWR             ;;BELL ON ERROR?
14493  062664  001402                           BEQ     1$                      ;;NO - SKIP
14494  062666  104401  001212                   TYPE    ,$BELL                  ;;RING BELL
14495  062672  005237  001126           1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
14496  062676  011637  001132                   MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
14497  062702  162737  000002  001132           SUB     #2,$ERRPC
14498  062710  117737  116216  001130           MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
14499  062716  032777  020000  116230           BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
14500  062724  001004                           BNE     20$                     ;;SKIP TYPEOUTS
14501  062726  004737  057314                   JSR     PC,ERRTYP               ;;GO TO USER ERROR ROUTINE
14502  062732  104401  001217                   TYPE    ,$CRLF
14503  062736                           20$:
14504  062736  122737  000001  001242           CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
14505  062744  001007                           BNE     2$                      ;;NO,SKIP APT ERROR REPORT
14506  062746  113737  001130  062760           MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
14507  062754  004737  064702                   JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
14508  062760     000           21$:     .BYTE   0
14509  062761     000                    .BYTE   0
14510  062762  000777           22$:     BR      22$                     ;;APT ERROR LOOP
14511  062764  005777  116164           2$:     TST     @SWR                    ;;HALT ON ERROR
14512  062770  100002                           BPL     3$                      ;;SKIP IF CONTINUE
14513  062772  000000                           HALT                            ;;HALT ON ERROR!
14514  062774  104410                           CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
14515  062776  032777  001000  116150   3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
14516  063004  001402                           BEQ     4$                      ;;BR IF NO
14517  063006  013716  001124                   MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
14518  063012  005737  001210           4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
14519  063016  001402                           BEQ     5$                      ;;BR IF NONE
14520  063020  013716  001210                   MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
14521  063024                           5$:
14522  063024  022737  057274  000042           CMP     #$ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
14523  063032  001001                           BNE     6$                      ;;BRANCH IF NO
14524  063034  000000                           HALT                            ;;YES
14525  063036                           6$:
14526  063036  000002                           RTI                             ;;RETURN
14527                                   .SBTTL  TTY INPUT ROUTINE
14528
```

```
14529                                   ;;*****************************************************************
14530                                   .ENABL  LSB
14531  063040  000000                   $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
14532  063042  000000                   $TKQIN: .WORD   0               ;;INPUT POINTER
14533  063044  000000                   $TKQOUT: .WORD  0               ;;OUTPUT POINTER
14534  063046  000001                   $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
14535          063047                   $TKQEND=.
14536          063050                   .EVEN
14537
14538                                   ;*TK INITIALIZE ROUTINE
14539                                   ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
14540                                   ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
14541                                   ;
14542                                   ;*CALL:
14543                                   ;*      JSR     PC,$TKINT
14544                                   ;*      RETURN
14545                                   ;
14546  063050  005037  063040           $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
14547  063054  012737  063046  063042           MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
14548  063062  013737  063042  063044           MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
14549  063070  012737  063120  000060           MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
14550  063076  012737  000200  000062           MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
14551  063104  005777  116052                   TST     @$TKB           ;;CLEAR DONE FLAG
14552  063110  012777  000100  116042           MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
14553  063116  000207                           RTS     PC              ;;RETURN TO CALLER
14554
14555                                   ;*TK SERVICE ROUTINE
14556                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
14557                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
14558                                   ;*IT IN THE QUEUE.
14559                                   ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
14560                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START)
14561                                   ;
14562  063120  117746  116036           $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
14563  063124  042716  177600                   BIC     #↑C177,(SP)     ;;STRIP THE JUNK
14564  063130  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
14565  063134  001007                           BNE     1$              ;;BRANCH IF NO
14566  063136  104401  064234                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
14567  063142  004737  063050                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
14568  063146  005726                           TST     (SP)+           ;;CLEAN UP STACK
14569  063150  000137  004542                   JMP     START           ;;CONTROL C RESTART
14570  063154  021627  000007           1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
14571  063160  001004                           BNE     2$              ;;BRANCH IF NO
14572  063162  022737  000176  001154           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
14573  063170  001500                           BEQ     6$              ;;GO TO SWR CHANGE
14574
14575  063172                           2$:
14576  063172  022737  000001  063040           CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
14577  063200  001004                           BNE     3$              ;;BRANCH IF NO
14578  063202  104401  001212                   TYPE    ,$BELL          ;;RING THE TTY BELL
14579  063206  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
14580  063210  000451                           BR      5$              ;;EXIT
14581  063212  021627  000023           3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
14582  063216  001021                           BNE     32$             ;;BRANCH IF NO
14583  063220  005077  115734                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
14584  063224  005726                           TST     (SP)+           ;;CLEAN CHAR OFF STACK
```

```
14585  063226  105777  115726          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
14586  063232  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
14587  063234  117746  115722                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
14588  063240  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
14589  063244  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
14590  063250  001366                          BNE     31$             ;;BRANCH IF NO
14591  063252  012777  000100  115700          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
14592  063260  000002                          RTI                     ;;RETURN
14593  063262  005237  063040          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
14594  063266  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
14595  063272  002405                          BLT     4$              ;;BRANCH IF YES
14596  063274  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
14597  063300  003002                          BGT     4$              ;;BRANCH IF YES
14598  063302  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
14599  063306  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
14600  063312  005237  063042                  INC     $TKQIN          ;;UPDATE THE POINTER
14601  063316  023727  063042  063047          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
14602  063324  001003                          BNE     5$              ;;BRANCH IF NO
14603  063326  012737  063046  063042          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
14604  063334  000002                  5$:     RTI                     ;;RETURN
14605
14606                                  ;;*****************************************************************
14607                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
14608                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
14609                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
14610                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
14611  063336  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
14612  063344  001124                          BNE     15$             ;;EXIT IF NOT
14613  063346  105777  115606                  TSTB    @$TKS           ;;IS A CHAR WAITING?
14614  063352  100121                          BPL     15$             ;;IF NOT, EXIT
14615  063354  117746  115602                  MOVB    @$TKB,-(SP)     ;;YES
14616  063360  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
14617  063364  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
14618  063370  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
14619                                                                  ;;AND EXIT
14620
14621                                  ;;*****************************************************************
14622                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
14623                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
14624                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
14625  063372  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
14626  063400  001674                          BEQ     2$              ;;BRANCH IF YES
14627  063402  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
14628  063404  004737  063050                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
14629  063410  005077  115544                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
14630  063414  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
14631
14632  063422  104401  064246                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
14633  063426  104401  064253          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
14634  063432  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
14635  063436  104402                          .POC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
14636  063440  104401  064264                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
14637  063444  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
14638  063446  005046                          CLR     -(SP)           ;;THE NEW SWR
14639  063450  105777  115504          7$:     TSTB    @$TKS           ;;CHAR THERE?
14640  063454  100375                          BPL     7$              ;;IF NOT TRY AGAIN
```

```
14641
14642  063456  117746  115500                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
14643  063462  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
14644
14645  063466  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
14646  063472  001015                          BNE     9$              ;;BRANCH IF NOT
14647  063474  104401  064234                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
14648  063500  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
14649  063504  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
14650  063512  001003                          BNE     8$              ;;BRANCH IF NO
14651  063514  012777  000100  115436          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
14652  063522  000137  004542          8$:     JMP     START           ;;CONTROL-C RESTART
14653
14654
14655  063526  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
14656  063532  001005                          BNE     10$             ;;BRANCH IF NOT
14657  063534  104401  064241                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
14658  063540  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
14659  063544  000737                          BR      19$             ;;LET'S TRY IT AGAIN
14660
14661
14662  063546  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
14663  063552  001022                          BNE     16$             ;;BRANCH IF NO
14664  063554  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
14665  063560  001403                          BEQ     11$             ;;BRANCH IF YES
14666  063562  016677  000002  115364          MOV     2(SP),@SWR      ;;SAVE NEW SWR
14667  063570  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
14668  063574  104401  001217          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
14669  063600  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
14670  063606  001003                          BNE     15$             ;;BRANCH IF NOT
14671  063610  012777  000100  115342          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
14672  063616  000002          15$:    RTI                             ;;RETURN
14673  063620  004737  062014          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
14674  063624  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
14675  063630  002420                          BLT     18$             ;;BRANCH IF YES
14676  063632  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
14677  063636  003015                          BGT     18$             ;;BRANCH IF YES
14678  063640  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
14679  063644  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
14680  063650  001403                          BEQ     17$             ;;BRANCH IF YES
14681  063652  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
14682  063654  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
14683  063656  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
14684  063660  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
14685  063664  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
14686  063670  000667                          BR      7$              ;;GET THE NEXT ONE
14687  063672  104401  001216          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
14688  063676  000720                          BR      20$             ;;SIMULATE CONTROL-U
14689                                  .DSABL  LSB
14690
14691
14692                                  ;;*****************************************************************
14693                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
14694                                  ;*CALL:
14695                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
14696                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
```

```
14697                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
14698                                   ;
14699
14700  063700  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
14701  063702  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE PS
14702  063710  005066  000004                   CLR     4(SP)           ;;GET READY FOR A CHARACTER
14703  063714  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
14704  063716  012746  063724                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
14705  063722  000002                           RTI                     ;;POP NEW PC AND PS
14706  063724                          64$:
14707  063724  005737  063040           1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
14708  063730  001775                           BEQ     1$
14709  063732  005337  063040                   DEC     $TKCNT          ;;DECREMENT THE COUNTER
14710  063736  117766  177102  000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
14711  063744  005237  063044                   INC     $TKQOUT         ;;UPDATE THE POINTER
14712  063750  023727  063044  063047           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
14713  063756  001003                           BNE     2$              ;;BRANCH IF NO
14714  063760  012737  063046  063044           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
14715  063766  000002           2$:             RTI                     ;;RETURN
14716                                  ;;*******************************************************************
14717                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
14718                                  ;*CALL:
14719                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
14720                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
14721                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
14722
14723  063770  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
14724  063772  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
14725  063774  012703  064224           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
14726  064000  022703  064234           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
14727  064004  101456                           BLOS    4$              ;;BR IF YES
14728  064006  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
14729  064010  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
14730  064012  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
14731  064016  001022                           BNE     5$              ;;BR IF NO
14732  064020  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
14733  064022  001007                           BNE     6$              ;;BR IF NO
14734  064024  112737  000134  064222           MOVB    #'\,9$          ;;TYPE A BACK SLASH
14735  064032  104401  064222                   TYPE    ,9$
14736  064036  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
14737  064042  005303           6$:             DEC     R3              ;;BACKUP BY ONE
14738  064044  020327  064224                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
14739  064050  103434                           BLO     4$              ;;BR IF YES
14740  064052  111337  064222                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
14741  064056  104401  064222                   TYPE    ,9$             ;;GO TYPE
14742  064062  000746                           BR      2$              ;;GO READ ANOTHER CHAR.
14743  064064  005716           5$:             TST     (SP)            ;;RUBOUT KEY SET?
14744  064066  001406                           BEQ     7$              ;;BR IF NO
14745  064070  112737  000134  064222           MOVB    #'\,9$          ;;TYPE A BACK SLASH
14746  064076  104401  064222                   TYPE    ,9$
14747  064102  005016                           CLR     (SP)            ;;CLEAR THE RUBOUT KEY
14748  064104  122713  000025           7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
14749  064110  001003                           BNE     8$              ;;BR IF NO
14750  064112  104401  064241                   TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
14751  064116  000726                           BR      1$              ;;GO START OVER
14752  064120  122713  000022           8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
```

```
14753  064124  001011                        BNE     3$                      ;;BRANCH IF NO
14754  064126  105013                        CLRB    (R3)                    ;;CLEAR THE CHARACTER
14755  064130  104401  001217                TYPE    ,$CRLF                  ;;TYPE A "CR" & "LF"
14756  064134  104401  064224                TYPE    ,$TTYIN                 ;;TYPE THE INPUT STRING
14757  064140  000717                        BR      2$                      ;;GO PICKUP ANOTHER CHACTER
14758  064142  104401  001216        4$:     TYPE    ,$QUES                  ;;TYPE A '?'
14759  064146  000712                        BR      1$                      ;;CLEAR THE BUFFER AND LOOP
14760  064150  111337  064222        3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
14761  064154  104401  064222                TYPE    ,9$
14762  064160  122723  000015                CMPB    #15,(R3)+               ;;CHECK FOR RETURN
14763  064164  001305                        BNE     2$                      ;;LOOP IF NOT RETURN
14764  064166  105063  177777                CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
14765  064172  104401  001220                TYPE    ,$LF                    ;;TYPE A LINE FEED
14766  064176  005726                        TST     (SP)+                   ;;CLEAN RUBOUT KEY FROM THE STACK
14767  064200  012603                        MOV     (SP)+,R3                ;;RESTORE R3
14768  064202  011646                        MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
14769  064204  016666  000004  000002        MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
14770  064212  012766  064224  000004        MOV     #$TTYIN,4(SP)
14771  064220  000002                        RTI                             ;;RETURN
14772  064222     000        9$:     .BYTE   0                               ;;STORAGE FOR ASCII CHAR. TO TYPE
14773  064223     000                .BYTE   0                               ;;TERMINATOR
14774  064224  000010        $TTYIN: .BLKB   8.                              ;;RESERVE 8 BYTES FOR TTY INPUT
14775  064234  041536  005015  000   $CNTLC: .ASCIZ  /↑C/<15><12>            ;;CONTROL "C"
14776  064241     136  006525  000012$CNTLU: .ASCIZ  /↑U/<15><12>            ;;CONTROL "U"
14777  064246  043536  005015  000   $CNTLG: .ASCIZ  /↑G/<15><12>            ;;CONTROL "G"
14778  064253     015  051412  051127$MSWR:  .ASCIZ  <15><12>/SWR = /
14779  064260  036440  000040
14780  064264  020040  042516  020127$MNEW:  .ASCIZ  /  NEW = /
14781  064272  020075     000
14782          064276                .EVEN
14783                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
14784
14785                                ;;*****************************************************************
14786                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
14787                                ;*CHANGE IT TO BINARY.
14788                                ;*CALL:
14789                                ;*      RDOCT                           ;;READ AN OCTAL NUMBER
14790                                ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
14791                                ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
14792
14793  064276  011646        $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
14794  064300  016666  000004  000002        MOV     4(SP),2(SP)             ;;INPUT NUMBER
14795  064306  010046                        MOV     R0,-(SP)                ;;PUSH R0 ON STACK
14796  064310  010146                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
14797  064312  010246                        MOV     R2,-(SP)                ;;PUSH R2 ON STACK
14798  064314  104412        1$:     RDLIN                           ;;READ AN ASCIZ LINE
14799  064316  012600                        MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
14800  064320  005001                        CLR     R1                      ;;CLEAR DATA WORD
14801  064322  005002                        CLR     R2
14802  064324  112046        2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
14803  064326  001412                        BEQ     3$                      ;;IF ZERO GET OUT
14804  064330  006301                        ASL     R1                      ;;*2
14805  064332  006102                        ROL     R2
14806  064334  006301                        ASL     R1                      ;;*4
14807  064336  006102                        ROL     R2
14808  064340  006301                        ASL     R1                      ;;*8
```

```
14809  064342  006102                          ROL     R2
14810  064344  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
14811  064350  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
14812  064352  000764                          BR      2$              ;;LOOP
14813  064354  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
14814  064356  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
14815  064362  010237  064376                  MOV     R2,$HIOCT
14816  064366  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
14817  064370  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
14818  064372  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
14819  064374  000002                          RTI                     ;;RETURN
14820  064376  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
14821                                  .SBTTL  TRAP DECODER
14822
14823                                  ;;*****************************************************************
14824                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
14825                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
14826                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
14827                                  ;*GO TO THAT ROUTINE.
14828
14829  064400  016646  000002          $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
14830  064404  042716  000020                  BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
14831  064410  012746  064416                  MOV     #1$,-(SP)       ;;  T-BIT TRAPS
14832  064414  000002                          RTI                     ;;SET THE NEW STATUS
14833  064416  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
14834  064420  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
14835  064424  005740                          TST     -(R0)           ;;BACKUP BY 2
14836  064426  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
14837  064430  006300                          ASL     R0              ;;POSITION FOR INDEXING
14838  064432  016000  064452                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
14839  064436  000200                          RTS     R0              ;;GO TO ROUTINE
14840
14841
14842                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
14843
14844  064440  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
14845  064442  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
14846  064450  000002                          RTI                     ;;RESTORE THE PSW
14847
14848                                  .SBTTL  TRAP TABLE
14849
14850                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
14851                                  ;*BY THE "TRAP" INSTRUCTION.
14852
14853                                  ;       ROUTINE
14854                                  ;       -------
14855  064452  064440                  $TRPAD: .WORD   $TRAP2
14856  064454  061602                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
14857  064456  061400                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
14858  064460  061354                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
14859  064462  061414                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
14860  064464  061130                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
14861  064466  061054                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
14862
14863  064470  063426                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
14864
```

```
14865  064472  063336                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
14866  064474  063700                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
14867  064476  063770                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
14868  064500  064276                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
14869  064502  060760                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
14870  064504  061016                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
14871                                  .SBTTL  POWER DOWN AND UP ROUTINES
14872
14873                                  ;;**************************************************************
14874                                  ;POWER DOWN ROUTINE
14875  064506  012737  064646  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
14876  064514  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
14877  064522  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
14878  064524  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
14879  064526  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
14880  064530  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
14881  064532  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
14882  064534  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
14883  064536  017746  114412                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
14884  064542  010637  064652                  MOV     SP,$SAVR6       ;;SAVE SP
14885  064546  012737  064560  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
14886  064554  000000                          HALT
14887  064556  000776                          BR      .-2             ;;HANG UP
14888
14889                                  ;;**************************************************************
14890                                  ;POWER UP ROUTINE
14891  064560  012737  064646  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
14892  064566  013706  064652                  MOV     $SAVR6,SP       ;;GET SP
14893  064572  005037  064652                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
14894  064576  005237  064652          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
14895  064602  001375                          BNE     1$              ;;OF  WORD
14896  064604  012677  114344                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
14897  064610  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
14898  064612  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
14899  064614  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
14900  064616  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
14901  064620  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
14902  064622  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
14903  064624  012737  064506  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
14904  064632  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
14905  064640  104401                          TYPE                    ;REPORT THE POWER FAILURE
14906  064642  064654                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
14907  064644  000002                          RTI
14908  064646  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
14909  064650  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
14910  064652  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
14911  064654  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
14912  064662  000122
14913                                          .EVEN
14914                                  .SBTTL  APT COMMUNICATIONS ROUTINE
14915
14916                                  ;;**************************************************************
14917  064664  112737  000001  065130  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
14918  064672  112737  000001  065126  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
14919  064700  000403                          BR      $ATYC
14920  064702  112737  000001  065130  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
```

```
14921  064710                          $ATYC:
14922  064710  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
14923  064712  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
14924  064714  105737  065126                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
14925  064720  001450                          BEQ     5$              ;;IF NOT:  BR
14926  064722  122737  000001  001242          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
14927  064730  001031                          BNE     3$              ;;IF NOT:  BR
14928  064732  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
14929  064740  001425                          BEQ     3$              ;;IF NOT:  BR
14930  064742  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
14931  064746  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
14932  064754  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
14933  064760  001375                          BNE     1$              ;;IF NOT:  WAIT
14934  064762  010037  001236                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
14935  064766  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
14936  064770  001376                          BNE     2$
14937  064772  163700  001236                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
14938  064776  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
14939  065000  010037  001240                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
14940  065004  012737  000004  001222          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
14941  065012  000413                          BR      5$
14942  065014  017637  000004  065040  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
14943  065022  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
14944  065030  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
14945  065034  004737  061602                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
14946  065040  0000C?          4$:     .WORD   0
14947  065042                  5$:
14948  065042  105737  065130          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
14949  065046  001416                          BEQ     12$             ;;IF NOT:  BR
14950  065050  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
14951  065054  001413                          BEQ     12$             ;;IF NOT:  BR
14952  065056  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
14953  065062  001375                          BNE     11$             ;;IF NOT:  WAIT
14954  065064  017637  000004  001224          MOV     @4(SP),$FATAL   ;;GET ERROR #
14955  065072  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
14956  065100  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
14957  065104  105037  065130          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
14958  065110  105037  065127                  CLRB    $LFLG           ;;CLEAR LOG FLAG
14959  065114  105037  065126                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
14960  065120  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
14961  065122  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
14962  065124  000207                          RTS     PC              ;;RETURN
14963  065126     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
14964  065127     000          $LFLG:  .BYTE   0               ;;LOG FLAG
14965  065130     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
14966          065132                  .EVEN
14967          000200          APTSIZE=200
14968          000001          APTENV=001
14969          000100          APTSPOOL=100
14970          000040          APTCSUP=040
14971
```

```
14972                                          .SBTTL  CONSOLE MESSAGES
14973
14974  065132     051                          CLSPRN: .ASCII  @)@
14975  065133     075      000                 EQUALS: .ASCIZ  @=@
14976  065135     015   025012      000        PROMPT: .ASCIZ  <CR><LF>@*@
14977  065141     077      000                 QSTMRK: .ASCIZ  @?@
14978  065143                                  HELPQST:
14979  065143     015   052012   050131        .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
14980  065150  020105   042510   050114
14981  065156  052040   054105   020124
14982  065164  054450   047440   020122
14983  065172  024516   037477      000
14984  065177                                  UBUSQST:
14985  065177     015   041412   040510        .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N) ??@
14986  065204  043516   020105   046522
14987  065212  031460   052440   044516
14988  065220  052502   020123   042101
14989  065226  051104   051505   020123
14990  065234  051117   053040   041505
14991  065242  047524   020122   042101
14992  065250  051104   051505   020123
14993  065256  054450   047440   020122
14994  065264  024516   037440   000077
14995  065272  005015   051525   020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
14996  065300  040523   042515   042040
14997  065306  053105   041511   051505
14998  065314  024040   020131   051117
14999  065322  047040   020051   037477
15000  065330     000
15001  065331     015   051012   030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
15002  065336  020063   052502   020123
15003  065344  042101   051104   051505
15004  065352  020123   000050
15005  065356  005015   047105   051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
15006  065364  020131   047516   020124
15007  065372  047111   044440   047457
15008  065400  050040   043501      105
15009  065405     015   040412   042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
15010  065412  042522   051523   046440
15011  065420  051525   020124   042502
15012  065426  037040   033061   030060
15013  065434  030060      000
15014  065437     015   051012   030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
15015  065444  020063   042526   052103
15016  065452  051117   040440   042104
15017  065460  042522   051523   024040
15018  065466     000
15019  065467     015   042412   052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
15020  065474  054522   047440   052125
15021  065502  047440   020106   040522
15022  065510  043516      105
15023  065513     015   040412   042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
15024  065520  042522   051523   046440
15025  065526  051525   020124   042502
15026  065534  036040   030061   030060
15027  065542     000
```

```
15028  065543    015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
15029  065550  020063  047111  042524
15030  065556  051122  050125  020124
15031  065564  051120  047511  044522
15032  065572  054524  024040     000
15033  065577    015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
15034  065604  054522  047440  052125
15035  065612  047440  020106  040522
15036  065620  043516  000105
15037  065624                          CNSL07:
15038  065624  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
15039  065632  024040  024501  052040
15040  065640  020117  042524  052123
15041  065646  040440  046114  042040
15042  065654  053105  041511  051505
15043  065662  020054  051117  052040
15044  065670  050131  020105  042504
15045  065676  044526  042503
15046  065702  047040  046525  042502  .ASCII  @ NUMBER(S)@
15047  065710  024122  024523
15048  065714  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
15049  065722  047111  052101  020105
15050  065730  047111  052520  020124
15051  065736  044527  044124  041440
15052  065744  051101  044522  043501
15053  065752  020105  042522  052524
15054  065760  047122     000
15055
15056          065764                  .EVEN
```

```
15057
15058  065764                  FNCDTB:                                 ;FUNCTION CODE TABLE
15059
15060  065764  020000          .WORD   OPI                             ;NOP
15061  065766  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (2)
15062  065770  132000          .WORD   ATA!OPI!IVC!IAE                 ;SEEK
15063  065772  130000          .WORD   ATA!OPI!IVC                     ;RECALIBRATE
15064  065774  020000          .WORD   OPI                             ;DRIVE CLEAR
15065  065776  030000          .WORD   OPI!IVC                         ;RELEASE
15066  066000  130000          .WORD   OPI!ATA!IVC                     ;OFFSET
15067  066002  130000          .WORD   OPI!ATA!IVC                     ;RETURN TO CENTERLINE
15068  066004  020000          .WORD   OPI                             ;READ IN PRESET
15069  066006  020000          .WORD   OPI                             ;PACK ACKNOWLEDGE
15070  066010  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (24)
15071  066012  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (26)
15072  066014  132000          .WORD   ATA!OPI!IVC!IAE                 ;SEARCH
15073  066016  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (32)
15074  066020  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (34)
15075  066022  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (36)
15076  066024  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (40)
15077  066026  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (42)
15078  066030  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (44)
15079  066032  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (46)
15080  066034  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK DATA
15081  066036  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK HEADER AND DATA
15082  066040  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (54)
15083  066042  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (56)
15084  066044  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE         ;WRITE DATA
15085  066046  037000          .WORD   OPI!IVC!WLE!IAE!AOE             ;WRITE HEADER AND DATA
15086  066050  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (64)
15087  066052  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (66)
15088  066054  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ DATA
15089  066056  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ HEADER AND DATA
15090  066060  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (74)
15091  066062  130001          .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (76)
15092
15093  066064      001         ATNTBL: .BYTE   1.
15094  066065      002                 .BYTE   2.
15095  066066      004                 .BYTE   4.
15096  066067      010                 .BYTE   8.
15097  066070      020                 .BYTE   16.
15098  066071      040                 .BYTE   32.
15099  066072      100                 .BYTE   64.
15100  066073      200                 .BYTE   128.
15101
15102  066074                  RGDTPT:
15103  066074  000000                  .WORD   0.
15104  066076  000001                  .WORD   1.
15105  066100  000003                  .WORD   3.
15106  066102  000007                  .WORD   7.
15107  066104  000017                  .WORD   15.
15108  066106  000037                  .WORD   31.
15109  066110  000077                  .WORD   63.
15110  066112  000177                  .WORD   127.
15111  066114  000377                  .WORD   255.
15112  066116  000777                  .WORD   511.
```

```
15113  066120  001777                      .WORD   1023.
15114  066122  003777                      .WORD   2047.
15115  066124  007777                      .WORD   4095.
15116  066126  017777                      .WORD   8191.
15117  066130  037777                      .WORD   16383.
15118  066132  077777                      .WORD   32767.
15119  066134  177777              ONES:   .WORD   65535.
15120  066136  177777                      .WORD   65535.
15121  066140  077777                      .WORD   32767.
15122  066142  037777                      .WORD   16383.
15123  066144  017777                      .WORD   8191.
15124  066146  007777                      .WORD   4095.
15125  066150  003777                      .WORD   2047.
15126  066152  001777                      .WORD   1023.
15127  066154  000777                      .WORD   511.
15128  066156  000377                      .WORD   255.
15129  066160  000177                      .WORD   127.
15130  066162  000077                      .WORD   63.
15131  066164  000037                      .WORD   31.
15132  066166  000017                      .WORD   15.
15133  066170  000007                      .WORD   7.
15134  066172  000003                      .WORD   3.
15135  066174  000001                      .WORD   1.
15136  066176  000000              ZEROS:  .WORD   0.
15137  066200  000000                      .WORD   0.
15138  066202  000001                      .WORD   1.
15139  066204  000002                      .WORD   2.
15140  066206  000004                      .WORD   4.
15141  066210  000010                      .WORD   8.
15142  066212  000020                      .WORD   16.
15143  066214  000040                      .WORD   32.
15144  066216  000100                      .WORD   64.
15145  066220  000200                      .WORD   128.
15146  066222  000400                      .WORD   256.
15147  066224  001000                      .WORD   512.
15148  066226  002000                      .WORD   1024.
15149  066230  004000                      .WORD   2048.
15150  066232  010000                      .WORD   4096.
15151  066234  020000                      .WORD   8192.
15152  066236  040000                      .WORD   16384.
15153  066240  100000                      .WORD   32768.
15154  066242  100000                      .WORD   32768.
15155  066244  040000                      .WORD   16384.
15156  066246  020000                      .WORD   8192.
15157  066250  010000                      .WORD   4096.
15158  066252  004000                      .WORD   2048.
15159  066254  002000                      .WORD   1024.
15160  066256  001000                      .WORD   512.
15161  066260  000400                      .WORD   256.
15162  066262  000200                      .WORD   128.
15163  066264  000100                      .WORD   64.
15164  066266  000040                      .WORD   32.
15165  066270  000020                      .WORD   16.
15166  066272  000010                      .WORD   8.
15167  066274  000004                      .WORD   4.
15168  066276  000002                      .WORD   2.
```

```
15169  066300  000001                .WORD   1.
15170  066302  000000                .WORD   0.
15171  066304  177777                .WORD   65535.
15172  066306  177776                .WORD   65534.
15173  066310  177774                .WORD   65532.
15174  066312  177770                .WORD   65528.
15175  066314  177760                .WORD   65520.
15176  066316  177740                .WORD   65504.
15177  066320  177700                .WORD   65472.
15178  066322  177600                .WORD   65408.
15179  066324  177400                .WORD   65280.
15180  066326  177000                .WORD   65024.
15181  066330  176000                .WORD   64512.
15182  066332  174000                .WORD   63488.
15183  066334  170000                .WORD   61440.
15184  066336  160000                .WORD   57344.
15185  066340  140000                .WORD   49152.
15186  066342  100000                .WORD   32768.
15187  066344  000000                .WORD   0.
15188  066346  000000                .WORD   0.
15189  066350  100000                .WORD   32768.
15190  066352  140000                .WORD   49152.
15191  066354  160000                .WORD   57344.
15192  066356  170000                .WORD   61440.
15193  066360  174000                .WORD   63488.
15194  066362  176000                .WORD   64512.
15195  066364  177000                .WORD   65024.
15196  066366  177400                .WORD   65280.
15197  066370  177600                .WORD   65408.
15198  066372  177700                .WORD   65472.
15199  066374  177740                .WORD   65504.
15200  066376  177760                .WORD   65520.
15201  066400  177770                .WORD   65528.
15202  066402  177774                .WORD   65532.
15203  066404  177776                .WORD   65534.
15204  066406  177777                .WORD   65535.
15205  066410  125252                .WORD   43690.
15206  066412  152525                .WORD   43690./2
15207  066414  125252                .WORD   43690.
15208  066416  177777                .WORD   65535.
15209  066420  177776                .WORD   65534.
15210  066422  177775                .WORD   65533.
15211  066424  177773                .WORD   65531.
15212  066426  177767                .WORD   65527.
15213  066430  177757                .WORD   65519.
15214  066432  177737                .WORD   65503.
15215  066434  177677                .WORD   65471.
15216  066436  177577                .WORD   65407.
15217  066440  177377                .WORD   65279.
15218  066442  176777                .WORD   65023.
15219  066444  175777                .WORD   64511.
15220  066446  173777                .WORD   63487.
15221  066450  167777                .WORD   61439.
15222  066452  157777                .WORD   57343.
15223  066454  137777                .WORD   49151.
15224  066456  077777                .WORD   32767.
```

```
 15225  066460  077777                          .WORD   32767.
 15226  066462  137777                          .WORD   49151.
 15227  066464  157777                          .WORD   57343.
 15228  066466  167777                          .WORD   61439.
 15229  066470  173777                          .WORD   63487.
 15230  066472  175777                          .WORD   64511.
 15231  066474  176777                          .WORD   65023.
 15232  066476  177377                          .WORD   65279.
 15233  066500  177577                          .WORD   65407.
 15234  066502  177677                          .WORD   65471.
 15235  066504  177737                          .WORD   65503.
 15236  066506  177757                          .WORD   65519.
 15237  066510  177767                          .WORD   65527.
 15238  066512  177773                          .WORD   65531.
 15239  066514  177775                          .WORD   65533.
 15240  066516  177776                          .WORD   65534.
 15241  066520  177777                          .WORD   65535.
 15242  066522                          ENRGDT:
```

```
15243                                          .EVEN
15244
15245  066522  101673  074550  000000  EMT1:   .WORD   EMS300,EMS1,0
15246  066530  101711  101734  101761  EMT2:   .WORD   EMS301,EMS302,EMS303,EMS1,EMS304
15247  066536  074550  101772
15248  066542  104677  104120  104260          .WORD   EMS511,EMS500,EMS501,EMS502,EMS503,0
15249  066550  104305  104354  000000
15250  066556  101711  102001  101734  EMT3:   .WORD   EMS301,EMS306,EMS302
15251  066564  104677  104424  104260          .WORD   EMS511,EMS505,EMS501,EMS502,0
15252  066572  104305  000000
15253  066576  101673  101734  102015  EMT4:   .WORD   EMS300,EMS302,EMS307,EMS2
15254  066604  074621
15255  066606  104677  104305  104260          .WORD   EMS511,EMS502,EMS501,EMS503,0
15256  066614  104354  000000
15257  066620  101711  102056  102073  EMT5:   .WORD   EMS301,EMS310,EMS311
15258  066626  104677  104305  104260          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504
15259  066634  104354  104400
15260  066640  102137  000000                  .WORD   EMS312,0
15261  066644  101711  102001  102073  EMT6:   .WORD   EMS301,EMS306,EMS311
15262  066652  104677  104305  104260          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504,0
15263  066660  104354  104400  000000
15264  066666  101711  102175  101734  EMT7:   .WORD   EMS301,EMS313,EMS302
15265  066674  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,EMS504,EMS503,0
15266  066702  104400  104354  000000
15267  066710  102303  102324  102226  EMT10:  .WORD   EMS316,EMS317,EMS314
15268  066716  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15269  066724  000000
15270  066726  102303  102324  102255  EMT11:  .WORD   EMS316,EMS317,EMS315
15271  066734  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15272  066742  000000
15273  066744  102303  102344  102226  EMT12:  .WORD   EMS316,EMS320,EMS314
15274  066752  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15275  066760  000000
15276  066762  102303  102344  102255  EMT13:  .WORD   EMS316,EMS320,EMS315
15277  066770  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15278  066776  000000
15279  067000  102303  102364  102226  EMT14:  .WORD   EMS316,EMS321,EMS314
15280  067006  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15281  067014  000000
15282  067016  102303  102364  102255  EMT15:  .WORD   EMS316,EMS321,EMS315
15283  067024  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15284  067032  000000
15285  067034  102303  102404  102226  EMT16:  .WORD   EMS316,EMS322,EMS314
15286  067042  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15287  067050  000000
15288  067052  102303  102404  102255  EMT17:  .WORD   EMS316,EMS322,EMS315
15289  067060  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15290  067066  000000
15291  067070  101711  102443  101161  EMT20:  .WORD   EMS301,EMS324,EMS250
15292  067076  104677  104400                  .WORD   EMS511,EMS504
15293  067102  102137  102474  000000          .WORD   EMS312,EMS326,0
15294  067110  101711  102424  101161  EMT21:  .WORD   EMS301,EMS323,EMS250
15295  067116  104677  104400                  .WORD   EMS511,EMS504
15296  067122  102137  102463  000000          .WORD   EMS312,EMS325,0
15297  067130  101711  102175  101161  EMT22:  .WORD   EMS301,EMS313,EMS250
15298  067136  104677  104400  000000          .WORD   EMS511,EMS504,0
```

```
15299  067144  101711  102443  101217  EMT23:  .WORD   EMS301,EMS324,EMS251
15300  067152  104677  104260                  .WORD   EMS511,EMS501
15301  067156  102137  102474  000000          .WORD   EMS312,EMS326,0
15302  067164  101711  102424  101217  EMT24:  .WORD   EMS301,EMS323,EMS251
15303  067172  104677  104260                  .WORD   EMS511,EMS501
15304  067176  102137  102463  000000          .WORD   EMS312,EMS325,0
15305  067204  101711  102175  101217  EMT25:  .WORD   EMS301,EMS313,EMS251
15306  067212  104677  104260  000000          .WORD   EMS511,EMS501,0
15307  067220  101711  102001  101263  EMT26:  .WORD   EMS301,EMS306,EMS252,EMS253,EMS327,EMS254
15308  067226  101316  102504  101351
15309  067234  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,EMS502
15310  067242  104305
15311  067244  102512  102226  000000          .WORD   EMS330,EMS314,0
15312  067252  101673  101263          EMT27:  .WORD   EMS300,EMS252
15313  067256  104677  104260  000000          .WORD   EMS511,EMS501,0
15314  067264  101673  101263          EMT30:  .WORD   EMS300,EMS252
15315  067270  104677  104260  104400          .WORD   EMS511,EMS501,EMS504,0
15316  067276  000000
15317  067300  101673  101263          EMT31:  .WORD   EMS300,EMS252
15318  067304  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15319  067312  000000
15320  067314  101711  102443  101263  EMT32:  .WORD   EMS301,EMS324,EMS252
15321  067322  104677  104260  000000          .WORD   EMS511,EMS501,0
15322  067330  101711  102443  101263  EMT33:  .WORD   EMS301,EMS324,EMS252
15323  067336  104677  104260  104400          .WORD   EMS511,EMS501,EMS504,0
15324  067344  000000
15325  067346  101711  102443  101263  EMT34:  .WORD   EMS301,EMS324,EMS252
15326  067354  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15327  067362  000000
15328  067364  101711  102424  101263  EMT35:  .WORD   EMS301,EMS323,EMS252
15329  067372  104677  104260  000000          .WORD   EMS511,EMS501,0
15330  067400  101711  102175  101263  EMT36:  .WORD   EMS301,EMS313,EMS252
15331  067406  104677  104260  000000          .WORD   EMS511,EMS501,0
15332  067414  101711  102443  101412  EMT37:  .WORD   EMS301,EMS324,EMS255
15333  067422  102137  102474  000000          .WORD   EMS312,EMS326,0
15334  067430  101711  102424  101412  EMT40:  .WORD   EMS301,EMS323,EMS255
15335  067436  104677  104400                  .WORD   EMS511,EMS504
15336  067442  102137  102463  000000          .WORD   EMS312,EMS325,0
15337  067450  101711  102175  101412  EMT41:  .WORD   EMS301,EMS313,EMS255
15338  067456  104677  104400  000000          .WORD   EMS511,EMS504,0
15339  067464  101711  102424  101161  EMT42:  .WORD   EMS301,EMS323,EMS250,EMS327,EMS255
15340  067472  102504  101412
15341  067476  104677  104400  104260          .WORD   EMS511,EMS504,EMS501,EMS503,0
15342  067504  104354  000000
15343  067510  101673  074670          EMT43:  .WORD   EMS300,EMS3
15344  067514  104677  104260  000000          .WORD   EMS511,EMS501,0
15345  067522  102531  074735  101316  EMT44:  .WORD   EMS331,EMS4,EMS253
15346  067530  104677  104260  000000          .WORD   EMS511,EMS501,0
15347
15348  067536  101673  101316          EMT45:  .WORD   EMS300,EMS253
15349  067542  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15350  067550  000000
15351
15352  067552  101673  101316          EMT46:  .WORD   EMS300,EMS253
15353  067556  104677  104260  104400          .WORD   EMS511,EMS501,EMS504,0
15354  067564  000000
```

```
15355
15356  067566  101711  102443  101316  EMT47:  .WORD   EMS301,EMS324,EMS253
15357  067574  104677  104260  104354          .WORD   EMS511,EMS501,EMS503
15358  067602  102137  102474  000000          .WORD   EMS312,EMS326,0
15359
15360  067610  101711  102443  101316  EMT50:  .WORD   EMS301,EMS324,EMS253
15361  067616  104677  104260  104400          .WORD   EMS511,EMS501,EMS504
15362  067624  102137  102474  000000          .WORD   EMS312,EMS326,0
15363
15364  067632  101711  102424  101316  EMT51:  .WORD   EMS301,EMS323,EMS253
15365  067640  104677  104260                  .WORD   EMS511,EMS501
15366  067644  102137  102463  000000          .WORD   EMS312,EMS325,0
15367
15368  067652  101711  102175  101316  EMT52:  .WORD   EMS301,EMS313,EMS253
15369  067660  104677  104260  000000          .WORD   EMS511,EMS501,0
15370
15371  067666  102531  074735  101454  EMT53:  .WORD   EMS331,EMS4,EMS256
15372  067674  104677  104260  000000          .WORD   EMS511,EMS501,0
15373
15374  067702  101711  102443  101454  EMT54:  .WORD   EMS301,EMS324,EMS256
15375  067710  104677  104260                  .WORD   EMS511,EMS501
15376  067714  102137  102474  000000          .WORD   EMS312,EMS326,0
15377
15378  067722  101711  102424  101454  EMT55:  .WORD   EMS301,EMS323,EMS256
15379  067730  104677  104260                  .WORD   EMS511,EMS501
15380  067734  102137  102463  000000          .WORD   EMS312,EMS325,0
15381
15382  067742  101711  102175  101454  EMT56:  .WORD   EMS301,EMS313,EMS256
15383  067750  104677  104260  000000          .WORD   EMS511,EMS501,0
15384
15385  067756  101504  102561          EMT57:  .WORD   EMS257,EMS332
15386  067762  104677  104477  104260          .WORD   EMS511,EMS506,EMS501,0
15387  067770  000000
15388
15389  067772  074766  102577          EMT60:  .WORD   EMS5,EMS333
15390  067776  104677  104534  104260          .WORD   EMS511,EMS507,EMS501,0
15391  070004  000000
15392
15393  070006  101711  102443  101540  EMT61:  .WORD   EMS301,EMS324,EMS260
15394  070014  104677  104260                  .WORD   EMS511,EMS501
15395  070020  102137  102474  000000          .WORD   EMS312,EMS326,0
15396
15397  070026  101711  102424  101540  EMT62:  .WORD   EMS301,EMS323,EMS260
15398  070034  104677  104260                  .WORD   EMS511,EMS501
15399  070040  102137  102463  000000          .WORD   EMS312,EMS325,0
15400
15401  070046  101711  102175  101540  EMT63:  .WORD   EMS301,EMS313,EMS260
15402  070054  104677  104260  000000          .WORD   EMS511,EMS501,0
15403
15404  070062  101673  075307          EMT64:  .WORD   EMS300,EMS12
15405  070066  104677  104260  000000          .WORD   EMS511,EMS501,0
15406
15407  070074  075307  102623  102721  EMT65:  .WORD   EMS12,EMS335,EMS342
15408  070102  104677  104260  000000          .WORD   EMS511,EMS501,0
15409
15410  070110  075307  102646  102721  EMT66:  .WORD   EMS12,EMS336,EMS342
```

```
15411  070116  104677  104260  000000          .WORD   EMS511,EMS501,0
15412
15413  070124  101673  075036          EMT67:  .WORD   EMS300,EMS6
15414  070130  104677  104260                  .WORD   EMS511,EMS501
15415  070134  075102  102577  000000          .WORD   EMS7,EMS333,0
15416
15417  070142  101673  075036  102504  EMT70:  .WORD   EMS300,EMS6,EMS327,EMS7
15418  070150  075102
15419  070152  104677  104260  104400          .WORD   EMS511,EMS501,EMS504,0
15420  070160  000000
15421
15422  070162  075036  102623  102701  EMT71:  .WORD   EMS6,EMS335,EMS340,EMS10,EMS333,EMS342
15423  070170  075153  102577  102721
15424  070176  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15425  070204  000000
15426
15427  070206  075036  102646  102701  EMT72:  .WORD   EMS6,EMS336,EMS340,EMS10,EMS334,EMS342
15428  070214  075153  102613  102721
15429  070222  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15430  070230  000000
15431
15432  070232  101711  101540  101761  EMT73:  .WORD   EMS301,EMS260,EMS303,EMS11
15433  070240  075216
15434  070242  104677  104260  104305          .WORD   EMS511,EMS501,EMS502,0
15435  070250  000000
15436
15437  070252  102753  102767  102721  EMT74:  .WORD   EMS343,EMS344,EMS342,0
15438  070260  000000
15439
15440  070262  101673  075365          EMT75:  .WORD   EMS300,EMS13
15441  070266  104677  104354  000000          .WORD   EMS511,EMS503,0
15442
15443  070274  103044  075365  103016  EMT76:  .WORD   EMS346,EMS13,EMS345
15444  070302  104677  104354  000000          .WORD   EMS511,EMS503,0
15445
15446  070310  102665  075365  103016  EMT77:  .WORD   EMS337,EMS13,EMS345
15447  070316  104677  104354  000000          .WORD   EMS511,EMS503,0
15448
15449  070324  101711  102175  101351  EMT100: .WORD   FMS301,EMS313,EMS254,EMS347,EMS13
15450  070332  103062  075365
15451  070336  104677  104354  000000          .WORD   EMS511,EMS503,0
15452
15453  070344  103044  075434  102712  EMT101: .WORD   EMS346,EMS14,EMS341,EMS15
15454  070352  075501
15455  070354  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15456  070362  000000
15457
15458  070364  102665  075434  102712  EMT102: .WORD   EMS337,EMS14,EMS341,EMS15
15459  070372  075501
15460  070374  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15461  070402  000000
15462
15463  070404  101711  102175  101351  EMT103: .WORD   EMS301,EMS313,EMS254,EMS347,EMS15
15464  070412  103062  075501
15465  070416  104677  104354                  .WORD   EMS511,EMS503
15466  070422  075434  102561  000000          .WORD   EMS14,EMS332,0
```

```
15467
15468  070430  103044  075640  102712  EMT104: .WORD   EMS346,EMS17,EMS341,EMS16
15469  070436  075557
15470  070440  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15471  070446  000000
15472
15473  070450  102665  075640  102712  EMT105: .WORD   EMS337,EMS17,EMS341,EMS16
15474  070456  075557
15475  070460  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15476  070466  000000
15477
15478  070470  101711  102175  101351  EMT106: .WORD   EMS301,EMS313,EMS254,EMS347,EMS16
15479  070476  103062  075557
15480  070502  104677  104354                  .WORD   EMS511,EMS503
15481  070506  075640  102561  000000          .WORD   EMS17,EMS332,0
15482
15483  070514  103044  075701  102712  EMT107: .WORD   EMS346,EMS20,EMS341,EMS21
15484  070522  075745
15485  070524  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15486  070532  000000
15487
15488  070534  075701  103110  075745  EMT110: .WORD   EMS20,EMS351,EMS21,EMS350,EMS22,EMS315
15489  070542  103103  076024  102255
15490  070550  104677  104260  000000          .WORD   EMS511,EMS501,0
15491
15492  070556  075701  102613  103103  EMT111: .WORD   EMS20,EMS334,EMS350,EMS22,EMS333
15493  070564  076024  102577
15494  070570  104677  104260  000000          .WORD   EMS511,EMS501,0
15495
15496  070576  102665  075701  102712  EMT112: .WORD   EMS337,EMS20,EMS341,EMS21
15497  070604  075745
15498  070606  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15499  070614  000000
15500
15501  070616  102665  075701  102712  EMT113: .WORD   EMS337,EMS20,EMS341,EMS21,EMS350,EMS22,EMS334
15502  070624  075745  103103  076024
15503  070632  102613
15504  070634  104677  104260  000000          .WORD   EMS511,EMS501,0
15505
15506  070642  075701  103126  075745  EMT114: .WORD   EMS20,EMS352,EMS21,EMS350,EMS22,EMS333
15507  070650  103103  076024  102577
15508  070656  104677  104260  000000          .WORD   EMS511,EMS501,0
15509
15510  070664  101711  102175  101351  EMT115: .WORD   EMS301,EMS313,EMS254,EMS347,EMS21
15511  070672  103062  075745
15512  070676  104677  104354                  .WORD   EMS511,EMS503
15513  070702  075701  102561  000000          .WORD   EMS20,EMS332,0
15514
15515  070710  103044  076071  102712  EMT116: .WORD   EMS346,EMS23,EMS341,EMS24
15516  070716  076147
15517  070720  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15518  070726  000000
15519
15520  070730  102665  076071  102712  EMT117: .WORD   EMS337,EMS23,EMS341,EMS24
15521  070736  076147
15522  070740  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
```

```
15523  070746  000000
15524
15525  070750  101711  102175  101351  EMT120: .WORD   EMS301,EMS313,EMS254,EMS347,EMS24
15526  070756  103062  076147
15527  070762  104677  104354                  .WORD   EMS511,EMS503
15528  070766  076071  102561  000000          .WORD   EMS23,EMS332,0
15529
15530  070774  103044  076226  102712  EMT121: .WORD   EMS346,EMS25,EMS341,EMS26
15531  071002  076304
15532  071004  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15533  071012  000000
15534
15535  071014  102665  076226  102712  EMT122: .WORD   EMS337,EMS25,EMS341,EMS26
15536  071022  076304
15537  071024  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15538  071032  000000
15539
15540  071034  101711  102175  101351  EMT123: .WORD   EMS301,EMS313,EMS254,EMS347,EMS26
15541  071042  103062  076304
15542  071046  104677  104354                  .WORD   EMS511,EMS503
15543  071052  076226  102561  000000          .WORD   EMS25,EMS332,0
15544
15545  071060  101673  076363  102015  EMT124: .WORD   EMS300,EMS27,EMS307,EMS2
15546  071066  074621
15547  071070  104677  104354  000000          .WORD   EMS511,EMS503,0
15548
15549  071076  102531  076363  103142  EMT125: .WORD   EMS331,EMS27,EMS353
15550  071104  104677  104354                  .WORD   EMS511,EMS503
15551  071110  076427  102255  000000          .WORD   EMS30,EMS315,0
15552
15553  071116  102665  076363  102712  EMT126: .WORD   EMS337,EMS27,EMS341,EMS30
15554  071124  076427
15555  071126  104677  104354  000000          .WORD   EMS511,EMS503,0
15556
15557  071134  101711  102175  101351  EMT127: .WORD   EMS301,EMS313,EMS254,EMS347,EMS30
15558  071142  103062  076427
15559  071146  104677  104354                  .WORD   EMS511,EMS503
15560  071152  076363  102561  000000          .WORD   EMS27,EMS332,0
15561
15562  071160  076507  103166  101161  EMT130: .WORD   EMS31,EMS354,EMS250
15563  071166  104677  104400  104354          .WORD   EMS511,EMS504,EMS503,0
15564  071174  000000
15565
15566  071176  076560  103166  101161  EMT131: .WORD   EMS32,EMS354,EMS250
15567  071204  104677  104400  104354          .WORD   EMS511,EMS504,EMS503,0
15568  071212  000000
15569
15570  071214  103221  076640  101161  EMT132: .WORD   EMS355,EMS33,EMS250,EMS341,EMS30
15571  071222  102712  076427
15572  071226  104677  104400  000000          .WORD   EMS511,EMS504,0
15573
15574  071234  103221  076677  101161  EMT133: .WORD   EMS355,EMS34,EMS250,EMS341,EMS30
15575  071242  102712  076427
15576  071246  104677  104400  000000          .WORD   EMS511,EMS504,0
15577
15578  071254  102531  074735  101412  EMT134: .WORD   EMS331,EMS4,EMS255
```

```
15579  071262  104677  104400  000000          .WORD   EMS511,EMS504,0
15580
15581  071270  076735  103263  103305  EMT135: .WORD   EMS35,EMS357,EMS360,EMS15
15582  071276  075501
15583  071300  104677  104260  000000          .WORD   EMS511,EMS501,0
15584
15585  071306  101571  103361          EMT136: .WORD   EMS261,EMS362
15586  071312  104677  104354  000000          .WORD   EMS511,EMS503,0
15587
15588  071320  101673  076777  102015  EMT137: .WORD   EMS300,EMS36,EMS307,EMS2
15589  071326  074621
15590  071330  104677  104260  000000          .WORD   EMS511,EMS501,0
15591
15592  071336  103221  077027  101412  EMT140: .WORD   EMS355,EMS37,EMS255,EMS341,EMS30
15593  071344  102712  076427
15594  071350  104677  104400  000000          .WORD   EMS511,EMS504,0
15595
15596  071356  103044  077067  103016  EMT141: .WORD   EMS346,EMS40,EMS345
15597  071364  104677  104400  000000          .WORD   EMS511,EMS504,0
15598
15599  071372  102665  077067  102712  EMT142: .WORD   EMS337,EMS40,EMS341,EMS30
15600  071400  076427
15601  071402  104677  104400  000000          .WORD   EMS511,EMS504,0
15602
15603  071410  103402  102056  077145  EMT143: .WORD   EMS363,EMS310,EMS41
15604  071416  104677  104260  000000          .WORD   EMS511,EMS501,0
15605
15606  071424  102665  077145  102712  EMT144: .WORD   EMS337,EMS41,EMS341,EMS252,EMS327,EMS253
15607  071432  101263  102504  101316
15608  071440  104677  104260  000000          .WORD   EMS511,EMS501,0
15609
15610  071446  077145  103166  103417  EMT145: .WORD   EMS41,EMS354,EMS364,EMS252,EMS365,EMS253
15611  071454  101263  103440  101316
15612  071462  104677  104260  000000          .WORD   EMS511,EMS501,0
15613
15614  071470  101711  102001  076777  EMT146: .WORD   EMS301,EMS306,EMS36
15615  071476  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15616  071504  000000
15617
15618  071506  103445  077213          EMT147: .WORD   EMS366,EMS42
15619  071512  104677  104354  000000          .WORD   EMS511,EMS503,0
15620
15621  071520  103470  103142  103440  EMT150: .WORD   EMS367,EMS353,EMS365,EMS42,EMS354,EMS3
15622  071526  077213  103166  074670
15623  071534  104677  104354  000000          .WORD   EMS511,EMS503,0
15624
15625  071542  102665  076777          EMT151: .WORD   EMS337,EMS36
15626  071546  104677  104260  000000          .WORD   EMS511,EMS501,0
15627
15628  071554  077275  103166  076777  EMT152: .WORD   EMS43,EMS354,EMS36
15629  071562  104677  104260  000000          .WORD   EMS511,EMS501,0
15630
15631  071570  103470  103142  103440  EMT153: .WORD   EMS367,EMS353,EMS365,EMS36,EMS370
15632  071576  076777  103540
15633  071602  104677  104354  000000          .WORD   EMS511,EMS503,0
15634
```

```
15635  071610  103470  103142  103440  EMT154: .WORD   EMS367,EMS353,EMS365,EMS36,EMS371
15636  071616  076777  103555
15637  071622  104677  104354  000000          .WORD   EMS511,EMS503,0
15638
15639  071630  101673  077346  102015  EMT155: .WORD   EMS300,EMS44,EMS307,EMS2
15640  071636  074621
15641  071640  104677  104354  000000          .WORD   EMS511,EMS503,0
15642
15643  071646  103470  103142  103440  EMT156: .WORD   EMS367,EMS353,EMS365,EMS44,EMS354,EMS3
15644  071654  077346  103166  074670
15645  071662  104677  104354  000000          .WORD   EMS511,EMS503,0
15646
15647  071670  101673  077407  102015  EMT157: .WORD   EMS300,EMS45,EMS307,EMS2
15648  071676  074621
15649  071700  104677  104354  000000          .WORD   EMS511,EMS503,0
15650
15651  071706  103470  103142  103440  EMT160: .WORD   EMS367,EMS353,EMS365,EMS45,EMS354,EMS3
15652  071714  077407  103166  074670
15653  071722  104677  104354  104260          .WORD   EMS511,EMS503,EMS501
15654  071730  076735  102577  000000          .WORD   EMS35,EMS333,0
15655
15656  071736  101673  077464  102015  EMT161: .WORD   EMS300,EMS46,EMS307,EMS2
15657  071744  074621
15658  071746  104677  104354  000000          .WORD   EMS511,EMS503,0
15659
15660  071754  102665  077464  103142  EMT162: .WORD   EMS337,EMS46,EMS353
15661  071762  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15662  071770  000000
15663
15664  071772  076735  102623  102665  EMT163: .WORD   EMS35,EMS335,EMS337,EMS41,EMS334,EMS372
15665  072000  077145  102613  103604
15666  072006  104677  104260  000000          .WORD   EMS511,EMS501,0
15667
15668  072014  077546  102623  102665  EMT164: .WORD   EMS47,EMS335,EMS337,EMS41,EMS335,EMS372
15669  072022  077145  102623  103604
15670  072030  104677  104260  000000          .WORD   EMS511,EMS501,0
15671
15672  072036  102665  076735  102504  EMT165: .WORD   EMS337,EMS35,EMS327,EMS47
15673  072044  077546
15674  072046  104677  104260                  .WORD   EMS511,EMS501
15675  072052  077145  102577  103604          .WORD   EMS41,EMS333,EMS372,0
15676  072060  000000
15677
15678  072062  101673  077546  102015  EMT166: .WORD   EMS300,EMS47,EMS307,EMS2
15679  072070  074621
15680  072072  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15681  072100  000000
15682
15683  072102  077606  102623  102701  EMT167: .WORD   EMS50,EMS335,EMS340,EMS36,EMS333
15684  072110  076777  102577
15685  072114  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15686  072122  000000
15687
15688  072124  102665  076735          EMT170: .WORD   EMS337,EMS35
15689  072130  104677  104260  000000          .WORD   EMS511,EMS501,0
15690
```

```
15691  072136  077606  076677  074670  EMT171: .WORD   EMS50,EMS34,EMS3
15692  072144  104677  104260  000000          .WORD   EMS511,EMS501,0
15693
15694  072152  101711  102001  077655  EMT172: .WORD   EMS301,EMS306,EMS51
15695  072160  104677  104260  104400          .WORD   EMS511,EMS501,EMS504,0
15696  072166  000000
15697
15698  072170  103470  103142  103440  EMT173: .WORD   EMS367,EMS353,EMS365,EMS47,EMS354,EMS3
15699  072176  077546  103166  074670
15700  072204  104677  104260  000000          .WORD   EMS511,EMS501,0
15701
15702  072212  101673  101161  102504  EMT174: .WORD   EMS300,EMS250,EMS327,EMS255,EMS327,EMS256
15703  072220  101412  102504  101454
15704  072226  102712  104776                  .WORD   EMS341,EMS600
15705  072232  104677  104260  000000          .WORD   EMS511,EMS501,0
15706
15707  072240  101673  101454  102712  EMT175: .WORD   EMS300,EMS256,EMS341,EMS600
15708  072246  104776
15709  072250  104677  104260  000000          .WORD   EMS511,EMS501,0
15710
15711  072256  101673  101161  102712  EMT176: .WORD   EMS300,EMS250,EMS341,EMS600
15712  072264  104776
15713  072266  104677  104400  000000          .WORD   EMS511,EMS504,0
15714
15715  072274  101673  101412  102712  EMT177: .WORD   EMS300,EMS255,EMS341,EMS600
15716  072302  104776
15717  072304  104677  104400  000000          .WORD   EMS511,EMS504,0
15718
15719  072312  102665  077724  102712  EMT200: .WORD   EMS337,EMS52,EMS341,EMS601
15720  072320  105026
15721  072322  104677  104260  000000          .WORD   EMS511,EMS501,0
15722
15723  072330  103044  077724  102712  EMT201: .WORD   EMS346,EMS52,EMS341,EMS602
15724  072336  105046
15725  072340  104677  104260  000000          .WORD   EMS511,EMS501,0
15726
15727  072346  103044  077655  102712  EMT202: .WORD   EMS346,EMS51,EMS341,EMS602
15728  072354  105046
15729  072356  104677  104260  000000          .WORD   EMS511,EMS501,0
15730
15731  072364  103044  077724  103016  EMT203: .WORD   EMS346,EMS52,EMS345,EMS373,EMS255
15732  072372  103636  101412
15733  072376  104677  104400  104260          .WORD   EMS511,EMS504,EMS501,0
15734  072404  000000
15735
15736  072406  103044  077724  102712  EMT204: .WORD   EMS346,EMS52,EMS341,EMS27
15737  072414  076363
15738  072416  104677  104400  104260          .WORD   EMS511,EMS504,EMS501,0
15739  072424  000000
15740
15741  072426  077765  103166  074670  EMT205: .WORD   EMS53,EMS354,EMS3
15742  072434  104677  104354  104305          .WORD   EMS511,EMS503,EMS502,EMS510,0
15743  072442  104624  000000
15744
15745  072446  102665  100026  103642  EMT206: .WORD   EMS337,EMS54,EMS374,EMS250,EMS327,EMS255
15746  072454  101161  102504  101412
```

```
15747  072462  102504  074670                  .WORD   EMS327,EMS3
15748  072466  104677  104400  104260          .WORD   EMS511,EMS504,EMS501,0
15749  072474  000000
15750
15751  072476  100026  103166  074670  EMT207: .WORD   EMS54,EMS354,EMS3
15752  072504  104677  104260  000000          .WORD   EMS511,EMS501,0
15753
15754  072512  100026  103166  103417  EMT210: .WORD   EMS54,EMS354,EMS364,EMS250
15755  072520  101161
15756  072522  104677  104400  000000          .WORD   EMS511,EMS504,0
15757
15758  072530  100026  103166  103417  EMT211: .WORD   EMS54,EMS354,EMS364,EMS255
15759  072536  101412
15760  072540  104677  104400  000000          .WORD   EMS511,EMS504,0
15761
15762  072546  102665  100103          EMT212: .WORD   EMS337,EMS55
15763  072552  104677  104400  000000          .WORD   EMS511,EMS504,0
15764
15765  072560  100161  102613  103016  EMT213: .WORD   EMS56,EMS334,EMS345,EMS373,EMS262,EMS327,EMS251
15766  072566  103636  101630  102504
15767  072574  101217
15768  072576  104677  104260                  .WORD   EMS511,EMS501
15769  072602  100161  102623  000000          .WORD   EMS56,EMS335,0
15770
15771  072610  102665  100161          EMT214: .WORD   EMS337,EMS56
15772  072614  104677  104260  000000          .WORD   EMS511,EMS501,0
15773
15774  072622  100254  102577  103103  EMT215: .WORD   EMS57,EMS333,EMS350,EMS60,EMS334
15775  072630  100340  102613
15776  072634  104677  104260  000000          .WORD   EMS511,EMS501,0
15777
15778  072642  103402  102001  074766  EMT216: .WORD   EMS363,EMS306,EMS5
15779  072650  104677  104260  000000          .WORD   EMS511,EMS501,0
15780
15781  072656  103402  102001  100412  EMT217: .WORD   EMS363,EMS306,EMS61
15782  072664  104677  104260  000000          .WORD   EMS511,EMS501,0
15783
15784  072672  100465  102577  103673  EMT220: .WORD   EMS62,EMS333,EMS375,EMS251
15785  072700  101217
15786  072702  104677  104260  000000          .WORD   EMS511,EMS501,0
15787
15788  072710  100465  102577  103707  EMT221: .WORD   EMS62,EMS333,EMS376,EMS262
15789  072716  101630
15790  072720  104677  104260  000000          .WORD   EMS511,EMS501,0
15791
15792  072726  100465  102577  103707  EMT222: .WORD   EMS62,EMS333 EMS376,EMS250
15793  072734  101161
15794  072736  104677  104260  000000          .WORD   EMS511,EMS501,0
15795
15796  072744  103044  100465  102712  EMT223: .WORD   EMS346,EMS62,EMS341,EMS603
15797  072752  105107
15798  072754  104677  104260  000000          .WORD   EMS511,EMS501,0
15799
15800  072762  103044  100546  103707  EMT224: .WORD   EMS346,EMS63,EMS376,EMS262
15801  072770  101630
15802  072772  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
```

```
15803  073000  000000
15804
15805  073002  100546  102577  103103  EMT225: .WORD   EMS63,EMS333,EMS350,EMS363,EMS310,EMS262
15806  073010  103402  102056  101630
15807  073016  104677  104260  000000          .WORD   EMS511,EMS501,0
15808
15809  073024  100546  103263  076777  EMT226: .WORD   EMS63,EMS357,EMS36,EMS372
15810  073032  103604
15811  073034  104677  104260  000000          .WORD   EMS511,EMS501,0
15812
15813  073042  100546  103243  103305  EMT227: .WORD   EMS63,EMS356,EMS360,EMS15
15814  073050  075501
15815  073052  104677  104260  000000          .WORD   EMS511,EMS501,0
15816
15817  073060  100546  103243  103333  EMT230: .WORD   EMS63,EMS356,EMS361,EMS15
15818  073066  075501
15819  073070  104677  104260  000000          .WORD   EMS511,EMS501,0
15620
15821  073076  100546  103243  077145  EMT231: .WORD   EMS63,EMS356,EMS41
15822  073104  104677  104260  000000          .WORD   EMS511,EMS501,0
15823
15824  073112  077145  103723  102721  EMT232: .WORD   EMS41,EMS377,EMS342,EMS365,EMS63,EMS332
15825  073120  103440  100546  102561
15826  073126  104677  104260  000000          .WORD   EMS511,EMS501,0
15827
15828  073134  103470  103142  103440  EMT233: .WORD   EMS367,EMS353,EMS365,EMS63,EMS401
15829  073142  100546  103751
15830  073146  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15831  073154  000000
15832
15833  073156  075557  103723  103604  EMT234: .WORD   EMS16,EMS377,EMS372,EMS365,EMS64,EMS354,EMS3
15834  073164  103440  100606  103166
15835  073172  074670
15836  073174  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15837  073202  000000
15838
15839  073204  101673  100656  102015  EMT235: .WORD   EMS300,EMS65,EMS307,EMS2
15840  073212  074621
15841  073214  104677  104305  104354          .WORD   EMS511,EMS502,EMS503,0
15842  073222  000000
15843
15844  073224  100161  103723  103707  EMT236: .WORD   EMS56,EMS377,EMS376,EMS252,EMS372,EMS350
15845  073232  101263  103604  103103
15846  073240  100656  103751                  .WORD   EMS65,EMS401
15847  073244  104677  104305  104354          .WORD   EMS511,EMS502,EMS503,EMS501,0
15848  073252  104260  000000
15849
15850  073256  101673  100717  102015  EMT237: .WORD   EMS300,EMS66,EMS307,EMS2
15851  073264  074621
```

```
15852  073266  104677  104260  104354          .WORD   EMS511,EMS501,EMS503,0
15853  073274  000000
15854
15855  073276  075501  103734  103604  EMT240: .WORD   EMS15,EMS400,EMS372,EMS350,EMS66,EMS401
15856  073304  103103  100717  103751
15857  073312  104677  104354  104260          .WORD   EMS511,EMS503,EMS501,0
15858  073320  000000
15859
15860  073322  100717  102646  102701  EMT241: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS604
15861  073330  075501  104065  104055
15862  073336  105130
15863  073340  104677  104354  000000          .WORD   EMS511,EMS503,0
15864
15865  073346  103775  105130  103766  EMT242: .WORD   EMS403,EMS604,EMS402,EMS21,EMS377
15866  073354  075745  103723
15867  073360  104677  104354                  .WORD   EMS511,EMS503
15868  073364  076777  102623  000000          .WORD   EMS36,EMS335,0
15869
15870  073372  100717  102646  102701  EMT243: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS604
15871  073400  076304  104036  104055
15872  073406  105130
15873  073410  104677  104354  000000          .WORD   EMS511,EMS503,0
15874
15875  073416  100546  103751  104055  EMT244: .WORD   EMS63,EMS401,EMS405,EMS604
15876  073424  105130
15877  073426  104677  104354  000000          .WORD   EMS511,EMS503,0
15878
15879  073434  076777  103540  104055  EMT245: .WORD   EMS36,EMS370,EMS405,EMS604
15880  073442  105130
15881  073444  104677  104354  000000          .WORD   EMS511,EMS503,0
15882
15883  073452  103775  105130  103766  EMT246: .WORD   EMS403,EMS604,EMS402,EMS24,EMS377
15884  073460  076147  103723
15885  073464  104677  104354                  .WORD   EMS511,EMS503
15886  073470  076777  102623  000000          .WORD   EMS36,EMS335,0
15887
15888  073476  101001  102561  104055  EMT247: .WORD   EMS67,EMS332,EMS405,EMS604
15889  073504  105130
15890  073506  104677  104354  000000          .WORD   EMS511,EMS503,0
15891
15892  073514  100717  102646  102701  EMT250: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS605
15893  073522  075501  104065  104055
15894  073530  105155
15895  073532  104677  104354  000000          .WORD   EMS511,EMS503,0
15896
15897  073540  103775  105155  103766  EMT251: .WORD   EMS403,EMS605,EMS402,EMS21,EMS377
15898  073546  075745  103723
15899  073552  104677  104354                  .WORD   EMS511,EMS503
15900  073556  076777  102623  000000          .WORD   EMS36,EMS335,0
15901
15902  073564  100717  102646  102701  EMT252: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS605
15903  073572  076304  104036  104055
15904  073600  105155
15905  073602  104677  104354  000000          .WORD   EMS511,EMS503,0
15906
15907  073610  100546  103751  104055  EMT253: .WORD   EMS63,EMS401,EMS405,EMS605
```

```
15908  073616  105155
15909  073620  104677  104354  000000          .WORD   EMS511,EMS503,0
15910
15911  073626  076777  103540  104055  EMT254: .WORD   EMS36,EMS370,EMS405,EMS605
15912  073634  105155
15913  073636  104677  104354  000000          .WORD   EMS511,EMS503,0
15914
15915  073644  103775  105155  103766  EMT255: .WORD   EMS403,EMS605,EMS402,EMS24,EMS377
15916  073652  076147  103723
15917  073656  104677  104354                  .WORD   EMS511,EMS503
15918  073662  076777  102623  000000          .WORD   EMS36,EMS335,0
15919
15920  073670  101001  102561  104055  EMT256: .WORD   EMS67,EMS332,EMS405,EMS605
15921  073676  105155
15922  073700  104677  104354  000000          .WORD   EMS511,EMS503,0
15923
15924  073706  100717  102646  102701  EMT257: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS606
15925  073714  075501  104065  104055
15926  073722  105173
15927  073724  104677  104354  000000          .WORD   EMS511,EMS503,0
15928
15929  073732  103775  105173  103766  EMT260: .WORD   EMS403,EMS606,EMS402,EMS21,EMS377
15930  073740  075745  103723
15931  073744  104677  104354                  .WORD   EMS511,EMS503
15932  073750  076777  102623  000000          .WORD   EMS36,EMS335,0
15933
15934  073756  100717  102646  102701  EMT261: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS606
15935  073764  076304  104036  104055
15936  073772  105173
15937  073774  104677  104354  000000          .WORD   EMS511,EMS503,0
15938
15939  074002  100546  103751  104055  EMT262: .WORD   EMS63,EMS401,EMS405,EMS606
15940  074010  105173
15941  074012  104677  104354  000000          .WORD   EMS511,EMS503,0
15942
15943  074020  076777  103540  104055  EMT263: .WORD   EMS36,EMS370,EMS405,EMS606
15944  074026  105173
15945  074030  104677  104354  000000          .WORD   EMS511,EMS503,0
15946
15947  074036  103775  105173  103766  EMT264: .WORD   EMS403,EMS606,EMS402,EMS24,EMS377
15948  074044  076147  103723
15949  074050  104677  104354                  .WORD   EMS511,EMS503
15950  074054  076777  102623  000000          .WORD   EMS36,EMS335,0
15951
15952  074062  101113  103751  104055  EMT265: .WORD   EMS70,EMS401,EMS405,EMS606
15953  074070  105173
15954  074072  104677  104354  000000          .WORD   EMS511,EMS503,0
15955
15956  074100  101001  102561  104055  EMT266: .WORD   EMS67,EMS332,EMS405,EMS606
15957  074106  105173
15958  074110  104677  104354  000000          .WORD   EMS511,EMS503,0
15959
15960  074116  100717  103243  104100  EMT267: .WORD   EMS66,EMS356,EMS407
15961  074124  104677  104354  000000          .WORD   EMS511,EMS503,0
15962
15963  074132  100717  102646  102701  EMT270: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS607
```

```
15964  074140  075501  104065  104055
15965  074146  105213
15966  074150  104677  104354  000000          .WORD   EMS511,EMS503,0
15967
15968  074156  103775  105213  103766  EMT271: .WORD   EMS403,EMS607,EMS402,EMS21,EMS377
15969  074164  075745  103723
15970  074170  104677  104354                  .WORD   EMS511,EMS503
15971  074174  076363  102646  000000          .WORD   EMS27,EMS336,0
15972
15973  074202  076363  103540  104055  EMT272: .WORD   EMS27,EMS370,EMS405,EMS607
15974  074210  105213
15975  074212  104677  104354  000000          .WORD   EMS511,EMS503,0
15976
15977  074220  076363  103555  104055  EMT273: .WORD   EMS27,EMS371,EMS405,EMS607
15978  074226  105213
15979  074230  104677  104354  000000          .WORD   EMS511,EMS503,0
15980
15981  074236  076777  104036  104055  EMT274: .WORD   EMS36,EMS404,EMS405,EMS607
15982  074244  105213
15983  074246  104677  104354  000000          .WORD   EMS511,EMS503,0
15984
15985  074254  077765  103751  104055  EMT275: .WORD   EMS53,EMS401,EMS405,EMS607
15986  074262  105213
15987  074264  104677  104354  104305          .WORD   EMS511,EMS503,EMS502,0
15988  074272  000000
15989
15990  074274  101001  102561  104055  EMT276: .WORD   EMS67,EMS332,EMS405,EMS607
15991  074302  105213
15992  074304  104677  104354  000000          .WORD   EMS511,EMS503,0
15993
15994  074312  100717  102646  102701  EMT277: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS607
15995  074320  076304  104036  104055
15996  074326  105213
15997  074330  104677  104354  000000          .WORD   EMS511,EMS503,0
15998
15999  074336  103775  105213  103766  EMT300: .WORD   EMS403,EMS607,EMS402,EMS24,EMS377
16000  074344  076147  103723
16001  074350  104677  104354                  .WORD   EMS511,EMS503
16002  074354  076363  102646  000000          .WORD   EMS27,EMS336,0
16003
16004  074362  101113  103751  104055  EMT301: .WORD   EMS70,EMS401,EMS405,EMS607
16005  074370  105213
16006  074372  104677  104354  000000          .WORD   EMS511,EMS503,0
16007
16008  074400  105231  000000          EHT1:   .WORD   EH1,0
16009  074404  105310  000000          EHT2:   .WORD   EH2,0
16010  074410  105326  000000          EHT5:   .WORD   EH5,0
16011  074414  105365  000000          EHT7:   .WORD   EH7,0
16012  074420  105404  000000          EHT57:  .WORD   EH57,0
16013  074424  105503  000000          EHT65:  .WORD   EH65,0
16014  074430  105560  000000          EHT71:  .WORD   EH71,0
16015  074434  105317  000000          EHT74:  .WORD   EH3,0
16016  074440  105636  000000          EHT115: .WORD   EH115,0
16017  074444  105734  000000          EHT130: .WORD   EH130,0
16018  074450  106053  000000          EHT132: .WORD   EH132,0
16019  074454  106152  000000          EHT145: .WORD   EH145,0
```

```
16020  074460  106270  000000          EHT150: .WORD   EH150,0
16021  074464  106367  000000          EHT213: .WORD   EH213,0
16022  074470  106465  000000          EHT220: .WORD   EH220,0
16023
16024  074474  106524                  EDT1:   .WORD   ED1
16025  074476  106534                  EDT2:   .WORD   ED2
16026  074500  106540                  EDT5:   .WORD   ED5
16027  074502  106546                  EDT57:  .WORD   ED57
16028  074504  106560                  EDT65:  .WORD   ED65
16029  074506  106570                  EDT71:  .WORD   ED71
16030  074510  106534                  EDT74:  .WORD   ED2
16031  074512  106600                  EDT115: .WORD   ED115
16032  074514  106612                  EDT130: .WORD   ED130
16033  074516  106600                  EDT132: .WORD   ED115
16034  074520  106626                  EDT220: .WORD   ED220
16035
16036  074522  106632                  EFT1:   .WORD   EF1
16037  074524  106635                  EFT2:   .WORD   EF2
16038  074526  106636                  EFT5:   .WORD   EF5
16039  074530  106640                  EFT57:  .WORD   EF57
16040  074532  106632                  EFT65:  .WORD   EF1
16041  074534  106632                  EFT71:  .WORD   EF1
16042  074536  106635                  EFT74:  .WORD   EF2
16043  074540  106640                  EFT115: .WORD   EF57
16044  074542  106644                  EFT130: .WORD   EF130
16045  074544  106640                  EFT132: .WORD   EF57
16046  074546  106636                  EFT220: .WORD   EF5
16047
16048  074550  047516  042516  044530  EMS1:   .ASCIZ  @NONEXISTENT DEVICE "NED" (RMCS2,BIT 12) @
16049  074556  052123  047105  020124
16050  074564  042504  044526  042503
16051  074572  021040  042516  021104
16052  074600  024040  046522  051503
16053  074606  026062  044502  020124
16054  074614  031061  020051     000
16055  074621     103  047117  051124  EMS2:   .ASCIZ  @CONTROLLER CLEAR "CLR" (RMCS2,BIT 05) @
16056  074626  046117  042514  020122
16057  074634  046103  040505  020122
16058  074642  041442  051114  020042
16059  074650  051050  041515  031123
16060  074656  041054  052111  030040
16061  074664  024465  000040
16062  074670  052506  041516  044524  EMS3:   .ASCIZ  @FUNCTION CODE (RMCS1, BITS 01 - 05) @
16063  074676  047117  041440  042117
16064  074704  020105  051050  041515
16065  074712  030523  020054  044502
16066  074720  051524  030040  020061
16067  074726  020055  032460  020051
16068  074734     000
16069  074735     125  052516  042523  EMS4:   .ASCIZ  @UNUSED BIT POSITIONS OF @
16070  074742  020104  044502  020124
16071  074750  047520  044523  044524
16072  074756  047117  020123  043117
16073  074764  000040
16074  074766  042504  044526  042503  EMS5:   .ASCIZ  @DEVICE AVAILABLE "DVA" (RMCS1, BIT 11) @
16075  074774  040440  040526  046111
```

```
16076  075002  041101  042514  021040
16077  075010  053104  021101  024040
16078  075016  046522  051503  026061
16079  075024  041040  052111  030440
16080  075032  024461  000040
16081  075036  040520  052122  054511  EMS6:   .ASCIZ  @PARTIY ERROR "PAR" (RMER1, BIT 03) @
16082  075044  042440  051122  051117
16083  075052  021040  040520  021122
16084  075060  024040  046522  051105
16085  075066  026061  041040  052111
16086  075074  030040  024463  000040
16087  075102  040504  040524  050040  EMS7:   .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 03) @
16088  075110  051101  052111  020131
16089  075116  051105  047522  020122                          .
16090  075124  042042  042520  020042                          .
16091  075132  051050  042515  031122                          .
16092  075140  020054  044502  020124
16093  075146  031460  020051     000
16094  075153     120  051101  052111  EMS10:  .ASCIZ  @PARITY TEST "PAT" (RMCS2, BIT 04) @
16095  075160  020131  042524  052123
16096  075166  021040  040520  021124
16097  075174  024040  046522  051503
16098  075202  026062  041040  052111
16099  075210  030040  024464  000040
16100  075216  040515  051523  052502  EMS11:  .ASCII  @MASSBUS CONTROL BUS PARITY ERROR "MCPE" @
16101  075224  020123  047503  052116
16102  075232  047522  020114  052502
16103  075240  020123  040520  044522
16104  075246  054524  042440  051122
16105  075254  051117  021040  041515
16106  075262  042520  020042
16107  075266  051050  041515  030523          .ASCIZ  @(RMCS1, BIT 13) @
16108  075274  020054  044502  020124
16109  075302  031461  020051     000
16110  075307     111  046114  043505  EMS12:  .ASCIZ  @ILLEGAL REGISTER ERROR "ILR" (RMER1, BIT 01) @
16111  075314  046101  051040  043505
16112  075322  051511  042524  020122
16113  075330  051105  047522  020122
16114  075336  044442  051114  020042
16115  075344  051050  042515  030522
16116  075352  020054  044502  020124
16117  075360  030460  020051     000
16118  075365     104  040511  047107  EMS13:  .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 00) @
16119  075372  051517  044524  020103
16120  075400  047515  042504  021040
16121  075406  046504  021104  024040
16122  075414  046522  051115  026061
16123  075422  041040  052111  030040
16124  075430  024460  000040
16125  075434  042515  044504  046525  EMS14:  .ASCIZ  @MEDIUM ON LINE "MOL" (RMDS, BIT 12) @
16126  075442  047440  020116  044514
16127  075450  042516  021040  047515
16128  075456  021114  024040  046522
16129  075464  051504  020054  044502
16130  075472  020124  031061  020051
16131  075500     000
```

```
16132  075501     115  044501  052116  EMS15:  .ASCIZ  @MAINTENANCE UNIT READY "MUR" (RMMR1, BIT 09) @
16133  075506  047105  047101  042503
16134  075514  052440  044516  020124
16135  075522  042522  042101  020131
16136  075530  046442  051125  020042
16137  075536  051050  046515  030522
16138  075544  020054  044502  020124
16139  075552  034460  020051     000
16140  075557     115  044501  052116  EMS16:  .ASCIZ  @MAINTENANCE WRITE PROTECT "MWP" (RMMR1, BIT 03) @
16141  075564  047105  047101  042503
16142  075572  053440  044522  042524
16143  075600  050040  047522  042524
16144  075606  052103  021040  053515
16145  075614  021120  024040  046522
16146  075622  051115  026061  041040
16147  075630  052111  030040  024463
16148  075636  000040
16149  075640  051127  052111  020105  EMS17:  .ASCIZ  @WRITE LOCK "WRL" (RMDS, BIT 11) @
16150  075646  047514  045503  021040
16151  075654  051127  021114  024040
16152  075662  046522  051504  020054
16153  075670  044502  020124  030461
16154  075676  020051     000
16155  075701     104  053105  041511  EMS20:  .ASCIZ  @DEVICE CHECK "DVC" (RMER2, BIT 07) @
16156  075706  020105  044103  041505
16157  075714  020113  042042  041526
16158  075722  020042  051050  042515
16159  075730  031122  020054  044502
16160  075736  020124  033460  020051
16161  075744     000
16162  075745     115  044501  052116  EMS21:  .ASCIZ  @MAINTENANCE DRIVE FAULT "MDF" (RMMR1, BIT 06) @
16163  075752  047105  047101  042503
16164  075760  042040  044522  042526
16165  075766  043040  052501  052114
16166  075774  021040  042115  021106
16167  076002  024040  046522  051115
16168  076010  026061  041040  052111
16169  076016  030040  024466  000040
16170  076024  047125  040523  042506  EMS22:  .ASCIZ  @UNSAFE STATUS "UNS" (RMER1, BIT 14) @
16171  076032  051440  040524  052524
16172  076040  020123  052442  051516
16173  076046  020042  051050  042515
16174  076054  030522  020054  044502
16175  076062  020124  032061  020051
16176  076070     000
16177  076071     123  042505  020113  EMS23:  .ASCIZ  @SEEK INCOMPLETE STATUS "SKI" (RMER2, BIT 14) @
16178  076076  047111  047503  050115
16179  076104  042514  042524  051440
16180  076112  040524  052524  020123
16181  076120  051442  044513  020042
16182  076126  051050  042515  031122
16183  076134  020054  044502  020124
16184  076142  032061  020051     000
16185  076147     115  044501  052116  EMS24:  .ASCIZ  @MAINTENANCE SEEK ERROR "MSER" (RMMR1, BIT 07) @
16186  076154  047105  047101  042503
16187  076162  051440  042505  020113
```

```
16188  076170  051105  047522  020122
16189  076176  046442  042523  021122
16190  076204  024040  046522  051115
16191  076212  026061  041040  052111
16192  076220  030040  024467  000040
16193  076226  047520  044523  044524  EMS25:  .ASCIZ  @POSITIONING IN PROGRESS "PIP" (RMDS, BIT 13) @
16194  076234  047117  047111  020107
16195  076242  047111  050040  047522
16196  076250  051107  051505  020123
16197  076256  050042  050111  020042
16198  076264  051050  042115  026123
16199  076272  041040  052111  030440
16200  076300  024463  000040
16201  076304  040515  047111  042524  EMS26:  .ASCIZ  @MAINTENANCE ON CYLINDER "MOC" (RMMR1, BIT 08) @
16202  076312  040503  041516  020105
16203  076320  04711   041440  046131
16204  076326  047111  042504  020122
16205  076334  046442  041517  020042
16206  076342  051050  046515  030522
16207  076350  020054  044502  020124
16208  076356  034060  020051     000
16209  076363     105  042116  047440  EMS27:  .ASCIZ  @END OF BLOCK "EBL" (RMMR1, BIT 13) @
16210  076370  020106  046102  041517
16211  076376  020113  042442  046102
16212  076404  020042  051050  046515
16213  076412  030522  020054  044502
16214  076420  020124  031461  020051
16215  076426     000
16216  076427     104  040511  047107  EMS30:  .ASCIZ  @DIAGNOSTIC END OF BLOCK "DEBL" (RMMR1, BIT 13) @
16217  076434  051517  044524  020103
16218  076442  047105  020104  043117
16219  076450  041040  047514  045503
16220  076456  021040  042504  046102
16221  076464  020042  051050  046515
16222  076472  030522  020054  044502
16223  076500  020124  031461  020051
16224  076506     000
16225  076507     114  051501  020124  EMS31:  .ASCIZ  @LAST SECTOR STATUS "LS" (RMMR1, BIT 02) @
16226  076514  042523  052103  051117
16227  076522  051440  040524  052524
16228  076530  020123  046042  021123
16229  076536  024040  046522  051115
16230  076544  026061  041040  052111
16231  076552  030040  024462  000040
16232  076560  040514  052123  051440  EMS32:  .ASCIZ  @LAST SECTOR/TRACK STATUS "LST" (RMMR1, BIT 01) @
16233  076566  041505  047524  027522
16234  076574  051124  041501  020113
16235  076602  052123  052101  051525
16236  076610  021040  051514  021124
16237  076616  024040  046522  051115
16238  076624  026061  041040  052111
16239  076632  030040  024461  000040
16240  076640  042523  052103  051117  EMS33:  .ASCIZ  @SECTOR ADDRESS, BITS 00-04 OF @
16241  076646  040440  042104  042522
16242  076654  051523  020054  044502
16243  076662  051524  030040  026460
```

```
16244  076670  032060  047440  020106
16245  076676     000
16246  076677     124  040522  045503  EMS34:  .ASCIZ  @TRACK ADDRESS, BITS 08-10 OF @
16247  076704  040440  042104  042522
16248  076712  051523  020054  044502
16249  076720  051524  030040  026470
16250  076726  030061  047440  020106
16251  076734     000
16252  076735     126  046117  046525  EMS35:  .ASCIZ  @VOLUME VALID "VV" (RMDS, BIT 06) @
16253  076742  020105  040526  044514
16254  076750  020104  053042  021126
16255  076756  024040  046522  051504
16256  076764  020054  044502  020124
16257  076772  033060  020051     000
16258  076777     107  020117  044502  EMS36:  .ASCIZ  @GO BIT (RMCS1, BIT 00) @
16259  077004  020124  051050  041515
16260  077012  030523  020054  044502
16261  077020  020124  030060  020051
16262  077026     000
16263  077027     103  046131  047111  EMS37:  .ASCIZ  @CYLINDER ADDRESS, BIT 00-09 OF @
16264  077034  042504  020122  042101
16265  077042  051104  051505  026123
16266  077050  041040  052111  030040
16267  077056  026460  034460  047440
16268  077064  020106     000
16269  077067     114  051501  020124  EMS40:  .ASCIZ  @LAST BLOCK TAKEN STATUS "LBT" (RMDS, BIT 10) @
16270  077074  046102  041517  020113
16271  077102  040524  042513  020116
16272  077110  052123  052101  051525
16273  077116  021040  041114  021124
16274  077124  024040  046522  051504
16275  077132  020054  044502  020124
16276  077140  030061  020051     000
16277  077145     103  046517  047520  EMS41:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
16278  077152  044523  042524  042440
16279  077160  051122  051117  021040
16280  077166  051105  021122  024040
16281  077174  046522  051504  020054
16282  077202  044502  020124  032061
16283  077210  020051     000
16284  077213     103  046517  040515  EMS42:  .ASCIZ  @COMMAND SEQUENCER TEST BIT "TST" (RMMR2, BIT 12) @
16285  077220  042116  051440  050505
16286  077226  042525  041516  051105
16287  077234  052040  051505  020124
16288  077242  044502  020124  052042
16289  077250  052123  020042  051050
16290  077256  046515  031122  020054
16291  077264  044502  020124  031061
16292  077272  020051     000
```

```
16293  077275      104  044522  042526  EMS43:  .ASCIZ  @DRIVE READY STATUS "DRY" (RMDS, BIT 07) @
16294  077302  051040  040505  054504
16295  077310  051440  040524  052524
16296  077316  020123  042042  054522
16297  077324  020042  051050  042115
16298  077332  026123  041040  052111
16299  077340  030040  024467  000040
16300  077346  047503  052116  047111  EMS44:  .ASCIZ  @CONTINUE "CONT" (RMMR1, BIT 06) @
16301  077354  042525  021040  047503
16302  077362  052116  020042  051050
16303  077370  046515  030522  020054
16304  077376  044502  020124  033060
16305  077404  020051      000
16306  077407      111  053116  046101  EMS45:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
16307  077414  042111  041440  046517
16308  077422  040515  042116  042440
16309  077430  051122  051117  021040
16310  077436  053111  021103  024040
16311  077444  046522  051105  026062
16312  077452  041040  052111  030440
16313  077460  024462  000040
16314  077464  047514  051523  047440  EMS46:  .ASCIZ  @LOSS OF SYSTEM CLOCK ERROR "LSC" (RMER2, BIT 11) @
16315  077472  020106  054523  052123
16316  077500  046505  041440  047514
16317  077506  045503  042440  051122
16318  077514  051117  021040  051514
16319  077522  021103  024040  046522
16320  077530  051105  026062  041040
16321  077536  052111  030440  024461
16322  077544  000040
16323  077546  041517  052503  044520  EMS47:  .ASCIZ  @OCCUPIED "OCC" (RMMR1, BIT 15) @
16324  077554  042105  021040  041517
16325  077562  021103  024040  046522
16326  077570  051115  026061  041040
16327  077576  052111  030440  024465
16328  077604  000040
16329  077606  046111  042514  040507  EMS50:  .ASCIZ  @ILLEGAL FUNCTION "ILF" (RMER1, BIT 0) @
16330  077614  020114  052506  041516
16331  077622  044524  047117  021040
16332  077630  046111  021106  024040
16333  077636  046522  051105  026061
16334  077644  041040  052111  030040
16335  077652  020051      000
16336  077655      117  043106  042523  EMS51:  .ASCIZ  @OFFSET DIRECTION "OFD" (RMOF, BIT 07) @
16337  077662  020124  044504  042522
16338  077670  052103  047511  020116
16339  077676  047442  042106  020042
16340  077704  051050  047515  026106
16341  077712  041040  052111  030040
16342  077720  024467  000040
16343  077724  043117  051506  052105  EMS52:  .ASCIZ  @OFFSET MODE "OM" (RMDS, BIT 00) @
16344  077732  046440  042117  020105
16345  077740  047442  021115  024040
16346  077746  046522  051504  020054
16347  077754  044502  020124  030060
16348  077762  020051      000
```

```
16349  077765     122  047125  040440  EMS53:  .ASCIZ  @RUN AND GO "RG" (RMMR1, BIT 14) @
16350  077772  042116  043440  020117
16351  100000  051042  021107  024040
16352  100006  046522  051115  026061
16353  100014  041040  052111  030440
16354  100022  024464  000040
16355  100026  047111  040526  044514  EMS54:  .ASCIZ  @INVALID ADDRESS ERROR "IAE" (RMER1, BIT 10) @
16356  100034  020104  042101  051104
16357  100042  051505  020123  051105
16358  100050  047522  020122  044442
16359  100056  042501  020042  051050
16360  100064  042515  030522  020054
16361  100072  044502  020124  030061
16362  100100  020051     000
16363  100103     101  042104  042522  EMS55:  .ASCIZ  @ADDRESS OVERFLOW ERROR "AOE" (RMER1, BIT 09) @
16364  100110  051523  047440  042526
16365  100116  043122  047514  020127
16366  100124  051105  047522  020122
16367  100132  040442  042517  020042
16368  100140  051050  042515  030522
16369  100146  020054  044502  020124
16370  100154  034460  020051     000
16371  100161     122  043505  051511  EMS56:  .ASCII  @REGISTER MODIFICATION REFUSED ERROR @
16372  100166  042524  020122  047515
16373  100174  044504  044506  040503
16374  100202  044524  047117  051040
16375  100210  043105  051525  042105
16376  100216  042440  051122  051117
16377  100224     040
16378  100225     042  046522  021122          .ASCIZ  @"RMR" (RMER1, BIT 02) @
16379  100232  024040  046522  051105
16380  100240  026061  041040  052111
16381  100246  030040  024462  000040
16382  100254  051104  053111  020105  EMS57:  .ASCIZ  @DRIVE REQUEST REQUIRED STATUS "DRQ" (RMDT, BIT 11) @
16383  100262  042522  052521  051505
16384  100270  020124  042522  052521
16385  100276  051111  042105  051440
16386  100304  040524  052524  020123
16387  100312  042042  050522  020042
16388  100320  051050  042115  026124
16389  100326  041040  052111  030440
16390  100334  024461  000040
16391  100340  051120  043517  040522  EMS60:  .ASCIZ  @PROGRAMMABLE STATUS "PGM" (RMDS, BIT 09) @
16392  100346  046515  041101  042514
16393  100354  051440  040524  052524
16394  100362  020123  050042  046507
16395  100370  020042  051050  042115
16396  100376  026123  041040  052111
16397  100404  030040  024471  000040
16398  100412  051104  053111  020105  EMS61:  .ASCIZ  @DRIVE PRESENT STATUS "DPR" (RMDS, BIT 08) @
16399  100420  051120  051505  047105
16400  100426  020124  052123  052101
16401  100434  051525  021040  050104
16402  100442  021122  024040  046522
16403  100450  051504  020054  044502
16404  100455  020124  034060  020051
```

```
16405  100464     000
16406  100465     120  051117  020124  EMS62:  .ASCIZ  @PORT REQUEST FLOP "RQA,RQB" (RMMR2, BITS 15,14) @
16407  100472  042522  052521  051505
16408  100500  020124  046106  050117
16409  100506  021040  050522  026101
16410  100514  050522  021102  024040
16411  100522  046522  051115  026062
16412  100530  041040  052111  020123
16413  100536  032461  030454  024464
16414  100544  000040
16415  100546  052101  042524  052116  EMS63:  .ASCIZ  @ATTENTION "ATA" (RMDS, BIT 15) @
16416  100554  047511  020116  040442
16417  100562  040524  020042  051050
16418  100570  042115  026123  041040
16419  100576  052111  030440  024465
16420  100604  000040
16421  100606  051127  052111  020105  EMS64:  .ASCIZ  @WRITE LOCK ERROR "WLE" (RMER1, BIT 11) @
16422  100614  047514  045503  042440
16423  100622  051122  051117  021040
16424  100630  046127  021105  024040
16425  100636  046522  051105  026061
16426  100644  041040  052111  030440
16427  100652  024461  000040
16428  100656  054105  042503  052120  EMS65:  .ASCIZ  @EXCEPTION "REX" (RMMR1, BIT 12) @
16429  100664  047511  020116  051042
16430  100672  054105  020042  051050
16431  100700  046515  030522  020054
16432  100706  044502  020124  031061
16433  100714  020051     000
16434  100717     117  042520  040522  EMS66:  .ASCIZ  @OPERATION INCOMPLETE ERROR "OPI" (RMER1, BIT 13) @
16435  100724  044524  047117  044440
16436  100732  041516  046517  046120
16437  100740  052105  020105  051105
16438  100746  047522  020122  047442
16439  100754  044520  020042  051050
16440  100762  042515  030522  020054
16441  100770  044502  020124  031461
16442  100776  020051     000
16443  101001     124  043501  041040  EMS67:  .ASCIZ  @TAG BUS (RMMR2, BITS 00-09) OR TAG CONTROL @
16444  101006  051525  024040  046522
16445  101014  051115  026062  041040
16446  101022  052111  020123  030060
16447  101030  030055  024471  047440
16448  101036  020122  040524  020107
16449  101044  047503  052116  047522
16450  101052  020114     000
16451  101055     114  047111  051505          .ASCIZ  @LINES (RMMR2, BITS 10,11,13) @
16452  101062  024040  046522  051115
16453  101070  026062  041040  052111
16454  101076  020123  030061  030454
16455  101104  026061  031461  020051
16456  101112     000
16457  101113     123  040505  041522  EMS70:  .ASCIZ  @SEARCH ENABLE "ESRC" (RMMR1, BIT 11) @
16458  101120  020110  047105  041101
16459  101126  042514  021040  051505
16460  101134  041522  020042  051050
```

```
16461  101142  046515  030522  020054
16462  101150  044502  020124  030461
16463  101156  020051     000
16464
16465  101161     104  051511  020113  EMS250: .ASCIZ  @DISK ADDRESS REGISTER (RMDA) @
16466  101166  042101  051104  051505
16467  101174  020123  042522  044507
16468  101202  052123  051105  024040
16469  101210  046522  040504  020051
16470  101216     000
16471  101217     103  047117  051124  EMS251: .ASCIZ  @CONTROL STATUS REGISTER #1 (RMCS1) @
16472  101224  046117  051440  040524
16473  101232  052524  020123  042522
16474  101240  044507  052123  051105
16475  101246  021440  020061  051050
16476  101254  041515  030523  020051
16477  101262     000
16478  101263     105  051122  051117  EMS252: .ASCIZ  @ERROR REGISTER #1 (RMER1) @
16479  101270  051040  043505  051511
16480  101276  042524  020122  030443
16481  101304  024040  046522  051105
16482  101312  024461  000040
16483  101316  051105  047522  020122  EMS253: .ASCIZ  @ERROR REGISTER #2 (RMER2) @
16484  101324  042522  044507  052123
16485  101332  051105  021440  020062
16486  101340  051050  042515  031122
16487  101346  020051     000
16488  101351     115  044501  052116  EMS254: .ASCIZ  @MAINTENANCE REGISTER #1 (RMMR1) @
16489  101356  047105  047101  042503
16490  101364  051040  043505  051511
16491  101372  042524  020122  030443
16492  101400  024040  046522  051115
16493  101406  024461  000040
16494  101412  042504  044523  042522  EMS255: .ASCIZ  @DESIRED CYLINDER REGISTER (RMDC) @
16495  101420  020104  054503  044514
16496  101426  042116  051105  051040
16497  101434  043505  051511  042524
16498  101442  020122  051050  042115
16499  101450  024503  000040
16500  101454  043117  051506  052105  EMS256: .ASCIZ  @OFFSET REGISTER (RMOF) @
16501  101462  051040  043505  051511
16502  101470  042524  020122  051050
16503  101476  047515  024506  000040
16504  101504  051104  053111  020105  EMS257: .ASCIZ  @DRIVE TYPE REGISTER (RMDT) @
16505  101512  054524  042520  051040
16506  101520  043505  051511  042524
16507  101526  020122  051050  042115
16508  101534  024524  000040
16509  101540  047510  042114  047111  EMS260: .ASCIZ  @HOLDING REGISTER (RMHR) @
16510  101546  020107  042522  044507
16511  101554  052123  051105  024040
16512  101562  046522  051110  020051
16513  101570     000
16514  101571     123  051105  040511  EMS261: .ASCIZ  @SERIAL NUMBER REGISTER (RMSN) @
16515  101576  020114  052516  041115
16516  101604  051105  051040  043505
```

```
16517  101612  051511  042524  020122
16518  101620  051050  051515  024516
16519  101626  000040
16520  101630  052101  042524  052116  EMS262: .ASCIZ  @ATTENTION SUMMARY REGISTER (RMAS) @
16521  101636  047511  020116  052523
16522  101644  046515  051101  020131
16523  101652  042522  044507  052123
16524  101660  051105  024040  046522
16525  101666  051501  020051     000
16526
16527  101673     103  047101  047516  EMS300: .ASCIZ  @CANNOT CLEAR @
16528  101700  020124  046103  040505
16529  101706  020122     000
16530  101711     103  047101  047516  EMS301: .ASCIZ  @CANNOT WRITE/READ @
16531  101716  020124  051127  052111
16532  101724  027505  042522  042101
16533  101732  000040
16534  101734  047101  020131  042504  EMS302: .ASCIZ  @ANY DEVICE REGISTER @
16535  101742  044526  042503  051040
16536  101750  043505  051511  042524
16537  101756  020122     000
16538  101761     127  052111  047510  EMS303: .ASCIZ  @WITHOUT @
16539  101766  052125  000040
16540  101772  051105  047522  020122  EMS304: .ASCIZ  @ERROR @
16541  102000     000
16542  102001     101  047440  042516  EMS306: .ASCIZ  @A ONE FROM @
16543  102006  043040  047522  020115
16544  102014     000
16545  102015     125  044523  043516  EMS307: .ASCIZ  @USING MASSBUS INITIALIZE, I.E., @
16546  102022  046440  051501  041123
16547  102030  051525  044440  044516
16548  102036  044524  046101  055111
16549  102044  026105  044440  042456
16550  102052  026056  000040
16551  102056  020101  042532  047522  EMS310: .ASCIZ  @A ZERO FROM @
16552  102064  043040  047522  020115
16553  102072     000
16554  102073     105  042526  054522  EMS311: .ASCIZ  @EVERY DEVICE REGISTER BIT POSITION @
16555  102100  042040  053105  041511
16556  102106  020105  042522  044507
16557  102114  052123  051105  041040
16558  102122  052111  050040  051517
16559  102130  052111  047511  020116
16560  102136     000
16561  102137     124  042510  043040  EMS312: .ASCIZ  @THE FOLLOWING BITS ARE STUCK @
16562  102144  046117  047514  044527
16563  102152  043516  041040  052111
16564  102160  020123  051101  020105
16565  102166  052123  041525  020113
16566  102174     000
16567  102175     101  051440  044510  EMS313: .ASCIZ  @A SHIFTING ONE BIT FROM @
16568  102202  052106  047111  020107
16569  102210  047117  020105  044502
16570  102216  020124  051106  046517
16571  102224  000040
16572  102226  050101  042520  051101  EMS314: .ASCIZ  @APPEARS STUCK AT ZERO @
```

F10


```
 16573  102234  020123  052123  041525
 16574  102242  020113  052101  055040
 16575  102250  051105  020117     000
 16576  102255     101  050120  040505  EMS315: .ASCIZ  @APPEARS STUCK AT ONE @
 16577  102262  051522  051440  052524
 16578  102270  046503  040440  020124
 16579  102276  047117  020105     000
 16580  102303     122  043505  051511  EMS316: .ASCIZ  @REGISTER SELECT @
 16581  102310  042524  020122  042523
 16582  102316  042514  052103  000040
 16583  102324  020061  030450  031054  EMS317: .ASCIZ  @1 (1,2,4,8,16) @
 16584  102332  032054  034054  030454
 16585  102340  024466  000040
 16586  102344  020062  030450  031054  EMS320: .ASCIZ  @2 (1,2,4,8,16) @
 16587  102352  032054  034054  030454
 16588  102360  024466  000040
 16589  102364  020064  030450  031054  EMS321: .ASCIZ  @4 (1,2,4,8,16) @
 16590  102372  032054  034054  030454
 16591  102400  024466  000040
 16592  102404  020070  030450  031054  EMS322: .ASCIZ  @8 (1,2,4,8,16) @
 16593  102412  032054  034054  030454
 16594  102420  024466  000040
 16595  102424  046101  020114  047117  EMS323: .ASCIZ  @ALL ONES FROM @
 16596  102432  051505  043040  047522
 16597  102440  020115     000
 16598  102443     101  046114  055040  EMS324: .ASCIZ  @ALL ZEROS FROM @
 16599  102450  051105  051517  043040
 16600  102456  047522  020115     000
 16601  102463     101  020124  042532  EMS325: .ASCIZ  @AT ZERO @
 16602  102470  047522  000040
 16603  102474  052101  047440  042516  EMS326: .ASCIZ  @AT ONE @
 16604  102502  000040
 16605  102504  020054  051117  000040  EMS327: .ASCIZ  @, OR @
 16606  102512  005015  051503  046440  EMS330: .ASCIZ  <CR><LF>@CS MBA CLRL @
 16607  102520  040502  041440  051114
 16608  102536  020114     000
 16609  102531     103  047101  047516  EMS331: .ASCIZ  @CANNOT READ ZEROS FROM @
 16610  102536  020124  042522  042101
 16611  102544  055040  051105  051517
 16612  102552  043040  047522  020115
 16613  102560     000
 16614  102561     111  020123  047111  EMS332: .ASCIZ  @IS INCORRECT @
 16615  102566  047503  051122  041505
 16616  102574  020124     000
 16617  102577     111  020123  047516  EMS333: .ASCIZ  @IS NOT SET @
 16618  102604  020124  042523  020124
 16619  102612     000
 16620  102613     111  020123  042523  EMS334: .ASCIZ  @IS SET @
 16621  102620  020124     000
 16622  102623     123  047510  046125  EMS335: .ASCIZ  @SHOULD NOT BE SET @
 16623  102630  020104  047516  020124
 16624  102636  042502  051440  052105
 16625  102644  000040
 16626  102646  044123  052517  042114  EMS336: .ASCIZ  @SHOULD BE SET @
 16627  102654  041040  020105  042523
 16628  102662  020124     000
```

```
16629  102665     103  047101  047516  EMS337: .ASCIZ  @CANNOT SET @
16630  102672  020124  042523  020124
16631  102700     000
16632  102701     102  041505  052501  EMS340: .ASCIZ  @BECAUSE @
16633  102706  042523  000040
16634  102712  051525  047111  020107  EMS341: .ASCIZ  @USING @
16635  102720     000
16636  102721     104  051125  047111  EMS342: .ASCIZ  @DURING REGISTER TRANSFER @
16637  102726  020107  042522  044507
16638  102734  052123  051105  052040
16639  102742  040522  051516  042506
16640  102750  020122     000
16641  102753     125  042516  050130  EMS343: .ASCIZ  @UNEXPECTED @
16642  102760  041505  042524  020104
16643  102766     000
16644  102767     102  051525  052040  EMS344: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
16645  102774  046511  047505  052125
16646  103002  024040  032060  052040
16647  103010  040522  024520  000040
16648  103016  054502  051040  043505  EMS345: .ASCIZ  @BY REGISTER TRANSFER @
16649  103024  051511  042524  020122
16650  103032  051124  047101  043123
16651  103040  051105  000040
16652  103044  040503  047116  052117  EMS346: .ASCIZ  @CANNOT RESET @
16653  103052  051040  051505  052105
16654  103060  000040
16655  103062  044527  044124  052517  EMS347: .ASCIZ  @WITHOUT SETTING @
16656  103070  020124  042523  052124
16657  103076  047111  020107     000
16658  103103     102  052125  000040  EMS350: .ASCIZ  @BUT @
16659  103110  040527  020123  042522  EMS351: .ASCIZ  @WAS RESET BY @
16660  103116  042523  020124  054502
16661  103124  000040
16662  103126  040527  020123  042523  EMS352: .ASCIZ  @WAS SET BY @
16663  103134  020124  054502  000040
16664  103142  047111  042040  040511  EMS353: .ASCIZ  @IN DIAGNOSTIC MODE @
16665  103150  047107  051517  044524
16666  103156  020103  047515  042504
16667  103164  000040
16668  103166  051511  044440  041516  EMS354: .ASCIZ  @IS INCORRECT ACCORDING TO @
16669  103174  051117  042522  052103
16670  103202  040440  041503  051117
16671  103210  044504  043516  052040
16672  103216  020117     000
16673  103221     103  047101  047516  EMS355: .ASCIZ  @CANNOT INCREMENT @
16674  103226  020124  047111  051103
16675  103234  046505  047105  020124
16676  103242     000
16677  103243     127  051501  047040  EMS356: .ASCIZ  @WAS NOT SET BY @
16678  103250  052117  051440  052105
16679  103256  041040  020131     000
16680  103263     127  051501  047040  EMS357: .ASCIZ  @WAS NOT RESET BY @
16681  103270  052117  051040  051505
16682  103276  052105  041040  020131
16683  103304     000
16684  103305     060  052040  020117  EMS360: .ASCIZ  @0 TO 1 TRANSITION OF @
```

```
16685  103312  020061  051124  047101
16686  103320  044523  044524  047117
16687  103326  047440  020106     000
16688  103333     061  052040  020117  EMS361: .ASCIZ  @1 TO 0 TRANSITION OF @
16689  103340  020060  051124  047101
16690  103346  044523  044524  047117
16691  103354  047440  020106     000
16692  103361     111  020123  047111  EMS362: .ASCIZ  @IS INCONSISTENT @
16693  103366  047503  051516  051511
16694  103374  042524  052116  000040
16695  103402  040503  047116  052117  EMS363: .ASCIZ  @CANNOT READ @
16696  103410  051040  040505  020104
16697  103416     000
16698  103417     124  051505  020124  EMS364: .ASCIZ  @TEST PATTERN IN @
16699  103424  040520  052124  051105
16700  103432  020116  047111  000040
16701  103440  047101  020104     000  EMS365: .ASCIZ  @AND @
16702  103445     103  047101  047516  EMS366: .ASCIZ  @CANNOT INITIALIZE @
16703  103452  020124  047111  052111
16704  103460  040511  044514  042532
16705  103466  000040
16706  103470  044124  020105  047503  EMS367: .ASCIZ  @THE COMMAND SEQUENCER HAS BEEN CLOCKED @
16707  103476  046515  047101  020104
16708  103504  042523  052521  047105
16709  103512  042503  020122  040510
16710  103520  020123  042502  047105
16711  103526  041440  047514  045503
16712  103534  042105  000040
```

```
16713  103540  042522  042523  020124  EMS370: .ASCIZ  @RESET EARLY @
16714  103546  040505  046122  020131
16715  103554     000
16716  103555     104  042111  047040  EMS371: .ASCIZ  @DID NOT RESET ON TIME @
16717  103562  052117  051040  051505
16718  103570  052105  047440  020116
16719  103576  044524  042515  000040
16720  103604  052504  044522  043516  EMS372: .ASCIZ  @DURING COMMAND EXECUTION @
16721  103612  041440  046517  040515
16722  103620  042116  042440  042530
16723  103626  052503  044524  047117
16724  103634  000040
16725  103636  047524  000040          EMS373: .ASCIZ  @TO @
16726  103642  044527  044124  040440  EMS374: .ASCIZ  @WITH ANY COMBINATION OF @
16727  103650  054516  041440  046517
16728  103656  044502  040516  044524
16729  103664  047117  047440  020106
16730  103672     000
16731  103673     102  020131  042522  EMS375: .ASCIZ  @BY READING @
16732  103700  042101  047111  020107
16733  103706     000
16734  103707     102  020131  051127  EMS376: .ASCIZ  @BY WRITING @
16735  103714  052111  047111  020107
16736  103722     000
16737  103723     127  051501  051440  EMS377: .ASCIZ  @WAS SET @
16738  103730  052105  000040
16739  103734  040527  020123  047516  EMS400: .ASCIZ  @WAS NOT SET @
16740  103742  020124  042523  020124
16741  103750     000
16742  103751     104  042111  047040  EMS401: .ASCIZ  @DID NOT SET @
16743  103756  052117  051440  052105
16744  103764  000040
16745  103766  044127  046111  020105  EMS402: .ASCIZ  @WHILE @
16746  103774     000
16747  103775     103  046517  040515  EMS403: .ASCIZ  @COMMAND SEQUENCER DID NOT ABORT @
16748  104002  042116  051440  050505
16749  104010  042525  041516  051105
16750  104016  042040  042111  047040
16751  104024  052117  040440  047502
16752  104032  052122  000040
16753  104036  040527  020123  047516  EMS404: .ASCIZ  @WAS NOT RESET @
16754  104044  020124  042522  042523
16755  104052  020124     000
16756  104055     104  051125  047111  EMS405: .ASCIZ  @DURING @
16757  104062  020107     000
16758  104065     127  051501  051040  EMS406: .ASCIZ  @WAS RESET @
16759  104072  051505  052105  000040
16760  104100  042523  051101  044103  EMS407: .ASCIZ  @SEARCH TIMEOUT @
16761  104106  052040  046511  047505
16762  104114  052125  000040
16763
16764  104120  042011  053105  041511  EMS500: .ASCII  @       DEVICE IS NONEXISTENT,@<CR><LF>
16765  104126  020105  051511  047040
16766  104134  047117  054105  051511
16767  104142  042524  052116  006454
16768  104150     012
```

```
16769  104151     011  042504  044526          .ASCII  @       DEVICE IS SWITCHED TO OTHER PORT@<CR><LF>
16770  104156  042503  044440  020123
16771  104164  053523  052111  044103
16772  104172  042105  052040  020117
16773  104200  052117  042510  020122
16774  104206  047520  052122  005015
16775  104214  052011  040522  051516          .ASCIZ  @       TRANSCEIVER ENABLE SWITCH IS OFF@<CR><LF>
16776  104222  042503  053111  051105
16777  104230  042440  040516  046102
16778  104236  020105  053523  052111
16779  104244  044103  044440  020123
16780  104252  043117  006506  000012
16781  104260  044411  020106  047515  EMS501: .ASCIZ  @       IF MODULE, M7686,@<CR><LF>
16782  104266  052504  042514  020054
16783  104274  033515  034066  026066
16784  104302  005015     000
16785  104305     011  040515  051523  EMS502: .ASCIZ  @       MASSBUS TRANSCEIVER,M5922 OR M5923      @<CR><LF>
16786  104312  052502  020123  051124
16787  104320  047101  041523  044505
16788  104326  042526  026122  032515
16789  104334  031071  020062  051117
16790  104342  046440  034465  031462
16791  104350  006411  000012
16792  104354  041411  020123  047515  EMS503: .ASCIZ  @       CS MODULE,M7684,@<CR><LF>
16793  104362  052504  042514  046454
16794  104370  033067  032070  006454
16795  104376  000012
16796  104400  042011  020123  047515  EMS504: .ASCIZ  @       DS MODULE,M7685,@<CR><LF>
16797  104406  052504  042514  046454
16798  104414  033067  032470  006454
16799  104422  000012
16800  104424  042011  053105  041511  EMS505: .ASCIZ  @       DEVICE IS SWITCHED TO A/B PORT POSITION@<CR><LF>
16801  104432  020105  051511  051440
16802  104440  044527  041524  042510
16803  104446  020104  047524  040440
16804  104454  041057  050040  051117
16805  104462  020124  047520  044523
16806  104470  044524  047117  005015
16807  104476     000
16808  104477     011  042504  044526  EMS506: .ASCIZ  @       DEVICE IS NOT AN RM03, OR@<CR><LF>
16809  104504  042503  044440  020123
16810  104512  047516  020124  047101
16811  104520  051040  030115  026063
16812  104526  047440  006522  000012
16813  104534  042011  053105  041511  EMS507: .ASCIZ  @       DEVICE IS SWITCHED TO PROGRAMMABLE PORT POSITION, OR@<CR><LF>
16814  104542  020105  051511  051440
16815  104550  044527  041524  042510
16816  104556  020104  047524  050040
16817  104564  047522  051107  046501
16818  104572  040515  046102  020105
16819  104600  047520  052122  050040
16820  104606  051517  052111  047511
16821  104614  026116  047440  006522
16822  104622  000012
16823  104624  040411  051523  046525  EMS510: .ASCIZ  @       ASSUMING THE RH CONTROLLER HAS NO FAULT@<CR><LF>
16824  104632  047111  020107  044124
```

```
16825  104646  020105  044122  041440
16826  104646  047117  051124  046117
16827  104654  042514  020122  040510
16828  104662  020123  047516  043040
16829  104670  052501  052114  005015
16830  104676     000
16831  104677     015  004412  051120  EMS511: .ASCII  <CR><LF>@       PROBABLE FAULT(S):@<CR><LF>
16832  104704  041117  041101  042514
16833  104712  043040  052501  052114
16834  104720  051450  035051  005015
16835  104726  024011  047516  020124          .ASCIZ  @       (NOT INCLUDING CABLES OR CONNECTORS)@<CR><LF>
16836  104734  047111  046103  042125
16837  104742  047111  020107  040503
16838  104750  046102  051505  047440
16839  104756  020122  047503  047116
16840  104764  041505  047524  051522
16841  104772  006451  000012
16842
16843  104776  042522  042101  044440  EMS600: .ASCIZ  @READ IN PRESET COMMAND @
16844  105004  020116  051120  051505
16845  105012  052105  041440  046517
16846  105020  040515  042116  000040
16847  105026  043117  051506  052105  EMS601: .ASCIZ  @OFFSET COMMAND @
16848  105034  041440  046517  040515
16849  105042  042116  000040
16850  105046  042522  052524  047122  EMS602: .ASCIZ  @RETURN TO CENTER CENTER COMMAND @
16851  105054  052040  020117  042503
16852  105062  052116  051105  041440
16853  105070  047105  042524  020122
16854  105076  047503  046515  047101
16855  105104  020104     000
16856  105107     122  046105  040505  EMS603: .ASCIZ  @RELEASE COMMAND @
16857  105114  042523  041440  046517
16858  105122  040515  042116  000040
16859  105130  042522  040503  044514  EMS604: .ASCIZ  @RECALIBRATE COMMAND @
16860  105136  051102  052101  020105
16861  105144  047503  046515  047101
16862  105152  020104     000
16863  105155     123  042505  020113  EMS605: .ASCIZ  @SEEK COMMAND @
16864  105162  047503  046515  047101
16865  105170  020104     000
16866  105173     123  040505  041522  EMS606: .ASCIZ  @SEARCH COMMAND @
16867  105200  020110  047503  046515
16868  105206  047101  020104     000
16869  105213     104  052101  020101  EMS607: .ASCIZ  @DATA COMMAND @
16870  105220  047503  046515  047101
16871  105226  020104     000
16872
16873  105231     105  050130  052103  EH1:    .ASCII  @EXPCTD  RECEVD  REGSTR@<CR><LF>
16874  105236  020104  051040  041505
16875  105244  053105  020104  051040
16876  105252  043505  052123  006522
16877  105260     012
16878  105261     123  040524  052524          .ASCIZ  @STATUS  STATUS  ADRESS@
16879  105266  020123  051440  040524
16880  105274  052524  020123  040440
```

```
16881  105302  051104  051505  000123
16882  105310  041040  051501  006505  EH2:    .ASCII  @ BASE@<CR><LF>
16883  105316     012
16884  105317     101  051104  051505  EH3:    .ASCIZ  @ADRESS@
16885  105324  000123
16886  105326  054105  041520  042124  EH5:    .ASCII  @EXPCTD   STUCK@<CR><LF>
16887  105334  020040  051440  052524
16888  105342  045503  005015
16889  105346  042522  052523  052114          .ASCIZ  @RESULT  BIT(S)@
16890  105354  020040  044502  024124
16891  105362  024523     000
16892  105365     105  050130  052103  EH7:    .ASCIZ  @EXPCTD  RECEVD@
16893  105372  020104  051040  041505
16894  105400  053105  000104
16895  105404  047123  050107  052122  EH57:   .ASCII  @SNGPRT  DULPRT  RECEVD  DRVTYP@<CR><LF>
16896  105412  020040  052504  050114
16897  105420  052122  020040  042522
16898  105426  042503  042126  020040
16899  105434  051104  052126  050131
16900  105442  005015
16901  105444  051104  052126  050131          .ASCIZ  @DRVTYP  DRVTYP  DRVTYP  REGADR@
16902  105452  020040  051104  052126
16903  105460  050131  020040  051104
16904  105466  052126  050131  020040
16905  105474  042522  040507  051104
16906  105502     000
16907  105503     105  050130  052103  EH65:   .ASCII  @EXPCTD  RECEVD  TEST@<CR><LF>
16908  105510  020104  051040  041505
16909  105516  053105  020104  052040
16910  105524  051505  006524     012
16911  105531     123  040524  052524          .ASCIZ  @STATUS  STATUS  REGSTR@
16912  105536  020123  051440  040524
16913  105544  052524  020123  051040
16914  105552  043505  052123  000122
16915  105560  054105  041520  042124  EH71:   .ASCII  @EXPCTD  RECEVD   TEST@<CR><LF>
16916  105566  020040  042522  042503
16917  105574  042126  020040  052040
16918  105602  051505  006524     012
16919  105607     123  040524  052524          .ASCIZ  @STATUS  STATUS  PATTRN@
16920  105614  020123  051440  040524
16921  105622  052524  020123  050040
16922  105630  052101  051124  000116
16923  105636  054105  041520  042124  EH115:  .ASCII  @EXPCTD  RECEVD  REGSTR   TEST@<CR><LF>
16924  105644  020040  042522  042503
16925  105652  042126  020040  042522
16926  105660  051507  051124  020040
16927  105666  052040  051505  006524
16928  105674     012
16929  105675     123  040524  052524          .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN@
16930  105702  020123  051440  040524
16931  105710  052524  020123  040440
16932  105716  051104  051505  020123
16933  105724  050040  052101  051124
16934  105732  000116
16935  105734  054105  041520  042124  EH130:  .ASCII  @EXPCTD  RECEVD  REGSTR   TEST   OFFSET@<CR><LF>
16936  105742  020040  042522  042503
```

```
16937  105750  042126  020040  042522
16938  105756  051507  051124  020040
16939  105764  052040  051505  020124
16940  105772  020040  043117  051506
16941  106000  052105  005015
16942  106004  052123  052101  051525          .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN  REGSTR@
16943  106012  020040  052123  052101
16944  106020  051525  020040  042101
16945  106026  042522  051523  020040
16946  106034  040520  052124  047122
16947  106042  020040  042522  051507
16948  106050  051124     000
16949  106053     105  050130  052103  EH132:  .ASCII  @EXPCTD  ACTUAL  REGSTR  OFFSET@<CR><LF>
16950  106060  020104  040440  052103
16951  106066  040525  020114  051040
16952  106074  043505  052123  020122
16953  106102  047440  043106  042523
16954  106110  006524     012
16955  106113     103  052517  052116          .ASCIZ  @COUNT   COUNT   ADRESS  REGSTR@
16956  106120  020040  041440  052517
16957  106126  052116  020040  040440
16958  106134  051104  051505  020123
16959  106142  051040  043505  052123
16960  106150  000122
16961  106152  054105  041520  042124  EH145:  .ASCII  @EXPCTD  ACTUAL  REGSTR  RMER1   RMER2@<CR><LF>
16962  106160  020040  041501  052524
16963  106166  046101  020040  042522
16964  106174  051507  051124  020040
16965  106202  046522  051105  020061
16966  106210  020040  046522  051105
16967  106216  006462     012
16968  106221     103  050115  051105          .ASCIZ  @CMPERR  CMPERR  ADRESS  PATTRN  PATTRN@
16969  106226  020122  041440  050115
16970  106234  051105  020122  040440
16971  106242  051104  051505  020123
16972  106250  050040  052101  051124
16973  106256  020116  050040  052101
16974  106264  051124  000116
16975
16976  106270  054105  041520  042124  EH150:  .ASCII  @EXPCTD  ACTUAL  REGSTR FUNCTION@<CR><LF>
16977  106276  020040  041501  052524
16978  106304  046101  020040  042522
16979  106312  051507  051124  043040
16980  106320  047125  052103  047511
16981  106326  006516     012
16982  106331     122  051505  046125          .ASCIZ  @RESULT  RESULT  ADRESS   CODE@
16983  106336  020124  051040  051505
16984  106344  046125  020124  040440
16985  106352  051104  051505  020123
16986  106360  020040  047503  042504
16987  106366     000
16988  106367     105  050130  052103  EH213:  .ASCII  @EXPCTD  ACTUAL  STATUS   TEST@<CR><LF>
16989  106374  020104  040440  052103
16990  106402  040525  020114  051440
16991  106410  040524  052524  020123
16992  106416  020040  042524  052123
```

```
16993  106424  005015
16994  106426  042522  052523  052114          .ASCIZ  @RESULT  RESULT  ADRESS  REGSTR@
16995  106434  020040  042522  052523
16996  106442  052114  020040  042101
16997  106450  042522  051523  020040
16998  106456  042522  051507  051124
16999  106464     000
17000  106465     101  052103  040525  EH220:  .ASCII  @ACTUAL  REGSTR@<CR><LF>
17001  106472  020114  051040  043505
17002  106500  052123  006522     012
17003  106505     122  051505  046125          .ASCIZ  @RESULT  ADRESS@
17004  106512  020124  040440  051104
17005  106520  051505  000123
17006                                  .EVEN
17007  106524  001140  001142  001136  ED1:    .WORD   $GDDAT,$BDDAT,$BDADR,0
17008  106532  000000
17009  106534  001136  000000          ED2:    .WORD   $BDADR,0
17010  106540  001140  001142  000000  ED5:    .WORD   $GDDAT,$BDDAT,0
17011  106546  001174  001176  001142  ED57:   .WORD   $TMP0,$TMP1,$BDDAT,$BDADR,0
17012  106554  001136  000000
17013  106560  001140  001142  001174  ED65:   .WORD   $GDDAT,$BDDAT,$TMP0,0
17014  106566  000000
17015  106570  001140  001142  001440  ED71:   .WORD   $GDDAT,$BDDAT,RMHR0,0
17016  106576  000000
17017  106600  001140  001142  001136  ED115:  .WORD   $GDDAT,$BDDAT,$BDADR,$TMP0,0
17018  106606  001174  000000
17019  106612  001140  001142  001136  ED130:  .WORD   $GDDAT $BDDAT,$BDADR,$TMP0,$TMP1,0
17020  106620  001174  001176  000000
17021  106626  001142  001136          ED220:  .WORD   $BDDAT,$BDADR
17022
17023  106632     000     000     000  EF1:    .BYTE   0,0,0
17024  106635     000                  EF2:    .BYTE   0
17025  106636     000     000          EF5:    .BYTE   0,0
17026  106640     000     000     000  EF57:   .BYTE   0,0,0,0
17027  106643     000
17028  106644     000     000     000  EF130:  .BYTE   0,0,0,0,0
17029  106647     000     000
17030
17031          106652                  .EVEN
17032  106652                  BUFFER:
17033  106652  000402          BUFONE: .BLKW   258.
17034  107656  000402          BUFTWO: .BLKW   258.
17035          106652                  .       =       BUFFER
17036  106652                  HELP:
17037  106652  005015          .ASCII  <CR><LF>
17038  106654  044514  052123  047440  .ASCII  @LIST OF TESTS@<CR><LF>
17039  106662  020106  042524  052123
17040  106670  006523     012
17041  106673     055  026455  026455  .ASCII  @------------@<CR><LF>
17042  106700  026455  026455  026455
17043  106706  026455  005015
17044  106712  005015          .ASCII  <CR><LF>
17045  106714  030524  052011  040522  .ASCII  @T1     TRANSFER TEST@<CR><LF>
17046  106722  051516  042506  020122
17047  106730  042524  052123  005015
17048  106736  031124  041411  047524  .ASCII  @T2     CTOD TEST@<CR><LF>
```

```
17049  106744  020104  042524  052123
17050  106752  005015
17051  106754  031524  046411  051501  .ASCII  @T3     MASSBUS INITIALIZE TEST@<CR><LF>
17052  106762  041123  051525  044440
17053  106770  044516  044524  046101
17054  106776  055111  020105  042524
17055  107004  052123  005015
17056  107010  032124  041411  042514  .ASCII  @T4     CLEAR STUCK ACTIVE TEST@<CR><LF>
17057  107016  051101  051440  052524
17058  107024  045503  040440  052103
17059  107032  053111  020105  042524
17060  107040  052123  005015
17061  107044  032524  052011  044522  .ASCII  @T5     TRISTATE TRANSFER TEST@<CR><LF>
17062  107052  052123  052101  020105
17063  107060  051124  047101  043123
17064  107066  051105  052040  051505
17065  107074  006524     012
17066  107077     124  004466  042522  .ASCII  @T6     REGISTER SELECT TEST@<CR><LF>
17067  107104  044507  052123  051105
17068  107112  051440  046105  041505
17069  107120  020124  042524  052123
17070  107126  005015
17071  107130  033524  042011  044522  .ASCII  @T7     DRIVE TYPE TEST@<CR><LF>
17072  107136  042526  052040  050131
17073  107144  020105  042524  052123
17074  107152  005015
17075  107154  030524  004460  042504  .ASCII  @T10    DEVICE AVAILABLE TEST@<CR><LF>
17076  107162  044526  042503  040440
17077  107170  040526  046111  041101
17078  107176  042514  052040  051505
17079  107204  006524     012
17080  107207     124  030461  044011  .ASCII  @T11    HOLDING REGISTER TRANSFER TEST@<CR><LF>
17081  107214  046117  044504  043516
17082  107222  051040  043505  051511
17083  107230  042524  020122  051124
17084  107236  047101  043123  051105
17085  107244  052040  051505  006524
17086  107252     012
17087  107253     124  031061  041411  .ASCII  @T12    CONTROL STATUS #1 TRANSFER TEST@<CR><LF>
17088  107260  047117  051124  046117
17089  107266  051440  040524  052524
17090  107274  020123  030443  052040
17091  107302  040522  051516  042506
17092  107310  020122  042524  052123
17093  107316  005015
17094  107320  030524  004463  051105   ASCII  @T13    ERROR REGISTER 1 TRANSFER TEST@<CR><LF>
17095  107326  047522  020122  042522
17096  107334  044507  052123  051105
17097  107342  030440  052040  040522
17098  107350  051516  042506  020122
17099  107356  042524  052123  005015
17100  107364  030524  004464  051105  .ASCII  @T14    ERROR REGISTER 2 TRANSFER TEST@<CR><LF>
17101  107372  047522  020122  042522
17102  107400  044507  052123  051105
17103  107406  031040  052040  040522
17104  107414  051516  042506  020122
```

```
17105  107422  042524  052123  005015
17106  107430  030524  004465  046103  .ASCII  @T15    CLEAR OFFSET STUCK ACTIVE TEST@<CR><LF>
17107  107436  040505  020122  043117
17108  107444  051506  052105  051440
17109  107452  052524  045503  040440
17110  107460  052103  053111  020105
17111  107466  042524  052123  005015
17112  107474  030524  004466  043117  .ASCII  @T16    OFFSET REGISTER TRANSFER TEST@<CR><LF>
17113  107502  051506  052105  051040
17114  107510  043505  051511  042524
17115  107516  020122  051124  047101
17116  107524  043123  051105  052040
17117  107532  051505  006524     012
17118  107537     124  033461  051411  .ASCII  @T17    SERIAL NUMBER TEST@<CR><LF>
17119  107544  051105  040511  020114
17120  107552  052516  041115  051105
17121  107560  052040  051505  006524
17122  107566     012
17123  107567     124  030062  041411  .ASCII  @T20    CONTROL BUS PARITY DETECTION TEST@<CR><LF>
17124  107574  047117  051124  046117
17125  107602  041040  051525  050040
17126  107610  051101  052111  020131
17127  107616  042504  042524  052103
17128  107624  047511  020116  042524
17129  107632  052123  005015
17130  107636  031124  004461  047503  .ASCII  @T21    CONTROL BUS PARITY GENERATION TEST@<CR><LF>
17131  107644  052116  047522  020114
17132  107652  052502  020123  040520
17133  107660  044522  054524  043440
17134  107666  047105  051105  052101
17135  107674  047511  020116  042524
17136  107702  052123  005015
17137  107706  031124  004462  046522  .ASCII  @T22    RMDA,RMDC FAULT TEST@<CR><LF>
17138  107714  040504  051054  042115
17139  107722  020103  040506  046125
17140  107730  020124  042524  052123
17141  107736  005015
17142  107740  031124  004463  044504  .ASCII  @T23    DISK ADDRESS TRANSFER TEST@<CR><LF>
17143  107746  045523  040440  042104
17144  107754  042522  051523  052040
17145  107762  040522  051516  042506
17146  107770  020122  042524  052123
17147  107776  005015
17148  110000  031124  004464  042504  .ASCII  @T24    DESIRED CYLINDER TRANSFER TEST@<CR><LF>
17149  110006  044523  042522  020104
17150  110014  054503  044514  042116
17151  110022  051105  052040  040522
17152  110030  051516  042506  020122
17153  110036  042524  052123  005015
17154  110044  031124  004465  046111  .ASCII  @T25    ILLEGAL REGISTER TEST@<CR><LF>
17155  110052  042514  040507  020114
17156  110060  042522  044507  052123
17157  110066  051105  052040  051505
17158  110074  006524     012
17159  110077     124  033062  051011  .ASCII  @T26    RESET GO BY INIT TEST@<CR><LF>
17160  110104  051505  052105  043440
```

```
17161  110112  020117  054502  044440
17162  110120  044516  020124  042524
17163  110126  052123  005015
17164  110132  031124  004467  044504  .ASCII  @T27    DIAGNOSTIC MODE TEST@<CR><LF>
17165  110140  043501  047516  052123
17166  110146  041511  046440  042117
17167  110154  020105  042524  052123
17168  110162  005015
17169  110164  031524  004460  047515  .ASCII  @T30    MOL TEST@<CR><LF>
17170  110172  020114  042524  052123
17171  110200  005015
17172  110202  031524  004461  051127  .ASCII  @T31    WRITE LOCK TEST@<CR><LF>
17173  110210  052111  020105  047514
17174  110216  045503  052040  051505
17175  110224  006524     012
17176  110227     124  031063  042011  .ASCII  @T32    DRIVE FAULT TEST@<CR><LF>
17177  110234  044522  042526  043040
17178  110242  052501  052114  052040
17179  110250  051505  006524     012
17180  110255     124  031463  051411  .ASCII  @T33    SEEK ERROR TEST@<CR><LF>
17181  110262  042505  020113  051105
17182  110270  047522  020122  042524
17183  110276  052123  005015
17184  110302  031524  004464  044520  .ASCII  @T34    PIP TEST@<CR><LF>
17185  110310  020120  042524  052123
17186  110316  005015
17187  110320  031524  004465  041105  .ASCII  @T35    EBL TEST@<CR><LF>
17188  110326  020114  042524  052123
17189  110334  005015
17190  110336  031524  004466  040514  .ASCII  @T36    LAST SECTOR, LAST TRACK TEST@<CR><LF>
17191  110344  052123  051440  041505
17192  110352  047524  026122  046040
17193  110360  051501  020124  051124
17194  110366  041501  020113  042524
17195  110374  052123  005015
17196  110400  031524  004467  046522  .ASCII  @T37    RMDA COUNT TEST@<CR><LF>
17197  110406  040504  041440  052517
17198  110414  052116  052040  051505
17199  110422  006524     012
17200  110425     124  030064  051011  .ASCII  @T40    RMDC COUNT TEST@<CR><LF>
17201  110432  042115  020103  047503
17202  110440  047125  020124  042524
17203  110446  052123  005015
17204  110452  032124  004461  041114  .ASCII  @T41    LBT TEST@<CR><LF>
17205  110460  020124  042524  052123
17206  110466  005015
17207  110470  032124  004462  047503  .ASCII  @T42    COMPOSITE ERROR TEST@<CR><LF>
17208  110476  050115  051517  052111
17209  110504  020105  051105  047522
17210  110512  020122  042524  052123
17211  110520  005015
17212  110522  032124  004463  051127  .ASCII  @T43    WRITE GO TEST@<CR><LF>
17213  110530  052111  020105  047507
17214  110536  052040  051505  006524
17215  110544     012
17216  110545     124  032064  041011  .ASCII  @T44    BRANCH MULTIPLEXOR TEST@<CR><LF>
```

```
17217  110552  040522  041516  020110
17218  110560  052515  052114  050111
17219  110566  042514  047530  020122
17220  110574  042524  052123  005015
17221  110602  032124  004465  042523  .ASCII  @T45    SET/RESET GO TEST@<CR><LF>
17222  110610  027524  042522  042523
17223  110616  020124  047507  052040
17224  110624  051505  006524     012
17225  110631     124  033064  042411  .ASCII  @T46    END 1 RESET GO TEST@<CR><LF>
17226  110636  042116  030440  051040
17227  110644  051505  052105  043440
17228  110652  020117  042524  052123
17229  110660  005015
17230  110662  032124  004467  042523  .ASCII  @T47    SET PULSE TEST@<CR><LF>
17231  110670  020124  052520  051514
17232  110676  020105  042524  052123
17233  110704  005015
17234  110706  032524  004460  042523  .ASCII  @T50    SET/RESET IVC TEST@<CR><LF>
17235  110714  027524  042522  042523
17236  110722  020124  053111  020103
17237  110730  042524  052123  005015
17238  110736  032524  004461  042523  .ASCII  @T51    SET LSC TEST@<CR><LF>
17239  110744  020124  051514  020103
17240  110752  042524  052123  005015
17241  110760  032524  004462  042504  .ASCII  @T52    DECODE TEST@<CR><LF>
17242  110766  047503  042504  052040
17243  110774  051505  006524     012
17244  111001     124  031465  051411  .ASCII  @T53    SET/RESET VOLUME VALID TEST@<CR><LF>
17245  111006  052105  051057  051505
17246  111014  052105  053040  046117
17247  111022  046525  020105  040526
17248  111030  044514  020104  042524
17249  111036  052123  005015
17250  111042  032524  004464  046111  .ASCII  @T54    ILLEGAL FUNCTION TEST@<CR><LF>
17251  111050  042514  040507  020114
17252  111056  052506  041516  044524
17253  111064  047117  052040  051505
17254  111072  006524     012
17255  111075     124  032465  047411  .ASCII  @T55    OCCUPIED TEST@<CR><LF>
17256  111102  041503  050125  042511
17257  111110  020104  042524  052123
17258  111115  005015
17259  111120  032524  004466  042522  .ASCII  @T56    READ IN PRESET TEST@<CR><LF>
17260  111126  042101  044440  020116
17261  111134  051120  051505  052105
17262  111142  052040  051505  006524
17263  111150     012
17264  111151     124  033465  051011  .ASCII  @T57    RIP/RMOF TEST@<CR><LF>
17265  111156  050111  051057  047515
17266  111164  020106  042524  052123
17267  111172  005015
17268  111174  033124  004460  046522  .ASCII  @T60    RMDA/RMDC/RIP TEST@<CR><LF>
17269  111202  040504  051057  042115
17270  111210  027503  044522  020120
17271  111216  042524  052123  005015
17272  111224  033124  004461  043117  .ASCII  @T61    OFFSET COMMAND TEST@<CR><LF>
```

```
17273  111232  051506  052105  041440
17274  111240  046517  040515  042116
17275  111246  052040  051505  006524
17276  111254     012
17277  111255     124  031066  051011  .ASCII  @T62    RETURN TO CENTER TEST@<CR><LF>
17278  111262  052105  051125  020116
17279  111270  047524  041440  047105
17280  111276  042524  020122  042524
17281  111304  052123  005015
17282  111310  033124  004463  046522  .ASCII  @T63    RMDC CLEAR OFFSET TEST@<CR><LF>
17283  111316  041504  041440  042514
17284  111324  051101  047440  043106
17285  111332  042523  020124  042524
17286  111340  052123  005015
17287  111344  033124  004464  041105  .ASCII  @T64    EBL CLEAR OFFSET TEST@<CR><LF>
17288  111352  020114  046103  040505
17289  111360  020122  043117  051506
17290  111366  052105  052040  051505
17291  111374  006524     012
17292  111377     124  032466  051011  .ASCII  @T65    RUN AND GO TEST@<CR><LF>
17293  111404  047125  040440  042116
17294  111412  043440  020117  042524
17295  111420  052123  005015
17296  111424  033124  004466  042523  .ASCII  @T66    SET IAE TEST@<CR><LF>
17297  111432  020124  040511  020105
17298  111440  042524  052123  005015
17299  111446  033124  004467  042523  .ASCII  @T67    SEARCH, SEEK, READ WRITE TEST@<CR><LF>
17300  111454  051101  044103  020054
17301  111462  042523  045505  020054
17302  111470  042522  042101  053440
17303  111476  044522  042524  052040
17304  111504  051505  006524     012
17305  111511     124  030067  044411  .ASCII  @T70    INVALID SECTOR/TRACK TEST@<CR><LF>
17306  111516  053116  046101  042111
17307  111524  051440  041505  047524
17308  111532  027522  051124  041501
17309  111540  020113  042524  052123
17310  111546  005015
17311  111550  033524  004461  047111  .ASCII  @T71    INVALID CYLINDER TEST@<CR><LF>
17312  111556  040526  044514  020104
17313  111564  054503  044514  042116
17314  111572  051105  052040  051505
17315  111600  006524     012
17316  111603     124  031067  051411  .ASCII  @T72    SET AOE TEST@<CR><LF>
17317  111610  052105  040440  042517
17318  111616  052040  051505  006524
17319  111624     012
17320  111625     124  031467  051411  .ASCII  @T73    SET RMR TEST@<CR><LF>
17321  111632  052105  051040  051115
17322  111640  052040  051505  006524
17323  111646     012
17324  111647     124  032067  050011  .ASCII  @T74    PGM STATUS CHECK@<CR><LF>
17325  111654  046507  051440  040524
17326  111662  052524  020123  044103
17327  111670  041505  006513     012
17328  111675     124  032467  042011  .ASCII  @T75    DPR STATUS CHECK@<CR><LF>
```

```
17329  111702  051120  051440  040524
17330  111710  052524  020123  044103
17331  111716  041505  006513     012
17332  111723     124  033067  050011  .ASCII  @T76    PORT REQUEST TEST, PART 1@<CR><LF>
17333  111730  051117  020124  042522
17334  111736  052521  051505  020124
17335  111744  042524  052123  020054
17336  111752  040520  052122  030440
17337  111760  005015
17338  111762  033524  004467  047520  .ASCII  @T77    PORT REQUEST TEST, PART 2@<CR><LF>
17339  111770  052122  051040  050505
17340  111776  042525  052123  052040
17341  112004  051505  026124  050040
17342  112012  051101  020124  006462
17343  112020     012
17344  112021     124  030061  004460  .ASCII  @T100   PORT REQUEST TEST, PART 3@<CR><LF>
17345  112026  047520  052122  051040
17346  112034  050505  042525  052123
17347  112042  052040  051505  026124
17348  112050  050040  051101  020124
17349  112056  006463     012
17350  112061     124  030061  004461  .ASCII  @T101   RELEASE TEST@<CR><LF>
17351  112066  042522  042514  051501
17352  112074  020105  042524  052123
17353  112102  005015
17354  112104  030524  031060  053411  .ASCII  @T102   WRITE ATA TEST@<CR><LF>
17355  112112  044522  042524  040440
17356  112120  040524  052040  051505
17357  112126  006524     012
17358  112131     124  030061  004463  .ASCII  @T103   RESET ATA BY GO TEST@<CR><LF>
17359  112136  042522  042523  020124
17360  112144  052101  020101  054502
17361  112152  043440  020117  042524
17362  112160  052123  005015
17363  112164  030524  032060  052411  .ASCII  @T104   UNIT READY ATA TEST@<CR><LF>
17364  112172  044516  020124  042522
17365  112200  042101  020131  052101
17366  112206  020101  042524  052123
17367  112214  005015
17368  112216  030524  032460  042411  .ASCII  @T105   ERROR ATA TEST@<CR><LF>
17369  112224  051122  051117  040440
17370  112232  040524  052040  051505
17371  112240  006524     012
17372  112243     124  030061  004466  .ASCII  @T106   REGISTER TRANSFER ATA TEST@<CR><LF>
17373  112250  042522  044507  052123
17374  112256  051105  052040  040522
17375  112264  051516  042506  020122
17376  112272  052101  020101  042524
17377  112300  052123  005015
17378  112304  030524  033460  050011  .ASCII  @T107   P SET ATA TEST@<CR><LF>
17379  112312  051440  052105  040440
17380  112320  040524  052040  051505
17381  112326  006524     012
17382  112331     124  030461  004460  .ASCII  @T110   SET WLE TEST@<CR><LF>
17383  112336  042523  020124  046127
17384  112344  020105  042524  052123
```

```
17385  112352  005015
17386  112354  030524  030461  042411  .ASCII  @T111   EXCEPTION TEST@<CR><LF>
17387  112362  041530  050105  044524
17388  112370  047117  052040  051505
17389  112376  006524     012
17390  112401     124  030461  004462  .ASCII  @T112   SET OPI TEST@<CR><LF>
17391  112406  042523  020124  050117
17392  112414  020111  042524  052123
17393  112422  005015
17394  112424  030524  031461  051011  .ASCII  @T113   RECALIBRATE TEST@<CR><LF>
17395  112432  041505  046101  041111
17396  112440  040522  042524  052040
17397  112446  051505  006524     012
17398  112453     124  030461  004464  .ASCII  @T114   SEEK TEST@<CR><LF>
17399  112460  042523  045505  052040
17400  112466  051505  006524     012
17401  112473     124  030461  004465  .ASCII  @T115   SEARCH TEST@<CR><LF>
17402  112500  042523  051101  044103
17403  112506  052040  051505  006524
17404  112514     012
17405  112515     124  030461  004466  .ASCII  @T116   SEARCH TIMEOUT TEST@<CR><LF>
17406  112522  042523  051101  044103
17407  112530  052040  046511  047505
17408  112536  052125  052040  051505
17409  112544  006524     012
17410  112547     124  030461  004467  .ASCII  @T117   DATA COMMAND TESTS (1)@<CR><LF>
17411  112554  040504  040524  041440
17412  112562  046517  040515  042116
17413  112570  052040  051505  051524
17414  112576  024040  024461  005015
17415  112604  030524  030062  042011  .ASCII  @T120   DATA COMMAND TESTS (2)@<CR><LF>
17416  112612  052101  020101  047503
17417  112620  046515  047101  020104
17418  112626  042524  052123  020123
17419  112634  031050  006451     012
17420  112641     124  031061  004461  .ASCII  @T121   DATA COMMAND TESTS(3)@<CR><LF>
17421  112646  040504  040524  041440
17422  112654  046517  040515  042116
17423  112662  052040  051505  051524
17424  112670  031450  006451     012
17425  112675     015     012          .ASCII  <CR><LF>
17426  112677     015     012          .ASCII  <CR><LF>
17427  112701     123  044527  041524  .ASCII  @SWITCH                 USE@<CR><LF>
17428  112706  004510  004411  051525               ▲
17429  112714  006505     012
17430  112717     055  026455  026455  .ASCII  @------          --------------------@<CR><LF>
17431  112724  004455  026411  026455
17432  112732  026455  026455  026455
17433  112740  026455  026455  026455
17434  112746  026455  026455  006455
17435  112754     012
17436  112755     040  030440  004465  .ASCII  @  15           HALT ON ERROR@<CR><LF>
17437  112762  044011  046101  020124
17438  112770  047117  042440  051122
17439  112776  051117  005015
17440  113002  020040  032061  004411  .ASCII  @  14           LOOP ON TEST@<CR><LF>
```

```
17441  113010  047514  050117  047440
17442  113016  020116  042524  052123
17443  113024  005015
17444  113026  020040  031461  004411  .ASCII  @  13          INHIBIT ERROR TYPEOUTS@<CR><LF>
17445  113034  047111  044510  044502
17446  113042  020124  051105  047522
17447  113050  020122  054524  042520
17448  113056  052517  051524  005015
17449  113064  020040  030461  004411  .ASCII  @  11          INHIBIT ITERATIONS@<CR><LF>
17450  113072  047111  044510  044502
17451  113100  020124  052111  051105
17452  113106  052101  047511  051516
17453  113114  005015
17454  113116  020040  030061  004411  .ASCII  @  10          BELL ON ERROR@<CR><LF>
17455  113124  042502  046114  047440
17456  113132  020116  051105  047522
17457  113140  006522     012
17458  113143     040  020040  004471  .ASCII  @   9          LOOP ON ERROR@<CR><LF>
17459  113150  046011  047517  020120
17460  113156  047117  042440  051122
17461  113164  051117  005015
17462  113170  020040  034040  004411  .ASCII  @   8          LOOP ON TEST IN SWR<7:0>@<CR><LF>
17463  113176  047514  050117  047440
17464  113204  020116  042524  052123
17465  113212  044440  020116  053523
17466  113220  036122  035067  037060
17467  113226  005015
17468  113230  020040  033440  004411  .ASCII  @   7          TN128@<CR><LF>
17469  113236  047124  031061  006470
17470  113244     012
17471  113245     040  020040  004466  .ASCII  @   6          TN64@<CR><LF>
17472  113252  052011  033116  006464
17473  113260     012
17474  113261     040  020040  004465  .ASCII  @   5          TN32@<CR><LF>
17475  113266  052011  031516  006462
17476  113274     012
17477  113275     040  020040  004464  .ASCII  @   4          TN16@<CR><LF>
17478  113302  052011  030516  006466
17479  113310     012
17480  113311     040  020040  004463  .ASCII  @   3          TN8@<CR><LF>
17481  113316  052011  034116  005015
17482  113324  020040  031040  004411  .ASCII  @   2          TN4@<CR><LF>
17483  113332  047124  006464     012
17484  113337     040  020040  004461  .ASCII  @   1          TN2@<CR><LF>
17485  113344  052011  031116  005015
17486  113352  020040  030040  004411  .ASCII  @   0          TN1@<CR><LF>
17487  113360  047124  006461     012
17488  113365     015  000012          .ASCIZ  <CR><LF>
17489          000001               .END
```

```
ABASE = 176700           3119#   3227    3268
ACDW1 = 000000           3227    3270
ACDW2 = 000000           3227    3271
ACPUOP= 000000           3227    3242
ADDW0 = 000000           3227    3272
ADDW1 = 000000           3227    3273
ADDW10= 000000           3227
ADDW11= 000000           3227
ADDW12= 000000           3227
ADDW13= 000000           3227
ADDW14= 000000           3227
ADDW15= 000000           3227
ADDW2 = 000000           3227    3274
ADDW3 = 000000           3227    3275
ADDW4 = 000000           3227    3276
ADDW5 = 000000           3227    3277
ADDW6 = 000000           3227    3278
ADDW7 = 000000           3227    3279
ADDW8 = 000000           3227
ADDW9 = 000000           3227
ADEVCT= 000000           3227    3233
ADEVM = 000000           3227    3269
AENV  = 000000           3227    3238
AENVM = 000000           3227    3239
AFATAL= 000000           3227    3230
AMADR1= 000000           3227    3255
AMADR2= 000000           3227    3259
AMADR3= 000000           3227    3262
AMADR4= 000000           3227    3265
AMAMS1= 000000           3227    3249
AMAMS2= 000000           3227    3257
AMAMS3= 000000           3227    3260
AMAMS4= 000000           3227    3263
AMSGAD= 000000           3227    3235
AMSGLG= 000000           3227    3236
AMSGTY= 000000           3227    3229
AMTYP1= 000000           3227    3250
AMTYP2= 000000           3227    3258
AMTYP3= 000000           3227    3261
AMTYP4= 000000           3227    3264
AOE   = 001000           2896#   2907    6204    6235    9241    10118   10122   15080   15081   15084   15085   15088   15089
APASS = 000000           3227    3232
APE   = 100000           3098#
APRIOR= 000000           3227
APTCSU= 000040          14268   14970#
APTENV= 000001          14261   14504   14926   14968#
APTSIZ= 000200           4985   14967#
APTSPO= 000100          14263   14928   14969#
ASWREG= 000000           3227    3240
ATA   = 100000           2876#  10512   10563   10604   10608   10618   10622   10659   10661   10700   10722   10776   10780
                        11262   11264   11632   11634   12045   12047   15061   15062   15063   15066   15067   15070   15071
                        15072   15073   15074   15075   15076   15077   15078   15079   15082   15083   15086   15087   15090
                        15091
ATESTN= 000000           3227    3231
ATNMSK= 000377           2913#  10506
ATNTBL  066064           5155    5173   10505   15093#
```

```
AUNIT = 000000          3227    3234
AUSWR = 000000          3227    3241
AVECT1= 120254          3120#   3227    3266
AVECT2= 000000          3227    3267
A16   = 000400          3069#   6981
A17   = 001000          3068#   6981
BAI   = 000010          3088#
BB00  = 000001          3005#
BB01  = 000002          3004#
BB02  = 000004          3003#   13525
BB03  = 000010          3002#   13521   13524
BB04  = 000020          3001#
BB05  = 000040          3000#
BB06  = 000100          2999#   11393   11394   11395   11396   11397   11398   11399   11400   11401   11402
BB07  = 000200          2998#
BB08  = 000400          2997#
BB09  = 001000          2996#
BIT0  = 000001          2792#   5540    6039    6129    6233    6241    6331    6338    6345    6362    6369    6385    6490
                        6497    6513    6778    6846    6858    6984    7077    7145    7231    7316    7443    7529    7603
                        7617    8664
BIT00 = 000001          2782#   2792    2818    2867    2886    2905    2943    2962    3005    3092    3108
BIT01 = 000002          2781#   2791    2817    2866    2904    2942    2961    3004    3091    3107
BIT02 = 000004          2780#   2790    2816    2865    2903    2941    2960    3003    3090    3106
BIT03 = 000010          2779#   2789    2815    2864    2902    2940    2959    3002    3017    3088    3105
BIT04 = 000020          2778#   2788    2814    2863    2901    2958    3001    3087
BIT05 = 000040          2777#   2787    2813    2900    2939    2957    3000    3086
BIT06 = 000100          2776#   2786    2885    2899    2921    2938    2956    2999    3071    3085    3104
BIT07 = 000200          2775#   2785    2884    2898    2920    2937    2955    2979    2998    3016    3070    3084
BIT08 = 000400          2774#   2784    2862    2883    2897    2919    2936    2954    2997    3069    3082    14347
BIT09 = 001000          2773#   2783    2861    2882    2896    2918    2935    2953    2996    3068    3081    14365   14515
BIT1  = 000002          2791#   5597    6054    6145    6265    6394    6522    6793    6872    6984    7090    7096    7158
                        7244    7349    7458    7540    7630
BIT10 = 002000          2772#   2860    2881    2895    2917    2934    2952    2978    2995    3015    3067    3080    3103
                        14492
BIT11 = 004000          2771#   2812    2880    2894    2933    2951    2969    2977    2994    3014    3079    3102    14372
BIT12 = 010000          2770#   2879    2893    2932    2950    2976    2993    3013    3078    3101
BIT13 = 020000          2769#   2878    2892    2931    2949    2968    2992    3012    3065    3077    3100    14499
BIT14 = 040000          2768#   2877    2891    2930    2948    2967    2991    3011    3023    3064    3076    3099    14333
BIT15 = 100000          2767#   2876    2890    2929    2947    2966    2990    3010    3022    3063    3075    3098
BIT2  = 000004          2790#
BIT3  = 000010          2789#
BIT4  = 000020          2788#
BIT5  = 000040          2787#
BIT6  = 000100          2786#
BIT7  = 000200          2785#   5024
BIT8  = 000400          2784#
BIT9  = 001000          2783#
BOTADR  060054          13765*  13783*  13786   13801   13846#
BOTFLG  060056          13751*  13793*  13796   13799*  13847#
BPTVEC= 000014          2799#
BSE   = 100000          3010#
BUFFER  106652          10096   13409   17032#  17035
BUFONE  106652          17033#
BUFTWO  107656          17034#
CC    = 004000          2994#   11393   11394   11395   11396   11397   11398   11399   11400   11401   11402   11782   11783
                        11784   11785   11786   11787   11788   11789   11790   11791   12289   12290   12291   12292   12293
```

```
                    12294  12295  12296  12297  12298  12651  13518  13586  13587  13588  13589  13590  13591
                    13592  13593  13594  13595  13629  13630  13631  13632  13633  13634  13635
CH     = 002000      2995#  11393  11394  11395  11396  11397  11398  11399  11400  11401  11402  12845  12846
                    12847  12848  12849  12850  12851  12852  12853  12854  13516  13597  13598  13599  13600
                    13601  13602  13603  13604  13605  13606  13629  13630  13631  13632  13633  13634  13635
CHRCNT 060057       13752* 13770* 13776* 13777  13780* 13784  13789* 13800* 13848#
CKSWR  = 104410     14332  14488  14514  14865#
CLOCK  001526        3361#  8866   9687  10960  12391  12685  12777  12920  12978  13090  13173  13285  13463
                    13872* 13885* 13896*
CLR    = 000040      3086#  5252   5273   5335   5388   5398   5436   5478   5543   5604   5726   5749   5781
                     5813   5837   5869   5892   5914   5993   6020   6098   6104   6191   6273   6321   6403
                     6442   6564   6610   6683   6720   6763   6832   6919   7020   7026   7057   7131   7221
                     7297   7426   7512   7596   7679   7779   7836   7903   7924   7990   8048   8082   8108
                     8158   8211   8354   8528   8623   8779   8845   9002   9076   9277   9344   9395   9672
                    10147  10220  10254  10492  10548  10589  10644  10686  10922  11016  13968
CLSPRN 065132       5076   5092   5113  14974#
CMNSTA 006052       5030   5162#
CNSL00 065272       5040  14995#
CNSL01 065331       5073  15001#
CNSL02 065356       5082  15005#
CNSL03 065437       5087  15014#
CNSL04 065467       5098  15019#
CNSL05 065543       5108  15028#
CNSL06 065577       5119  15033#
CNSL07 065624       5132  15037#
CONT   = 000100     2956#  8636   8662   8666
CR     = 000015     2707#  5145  13768  14307  14317  14976  14979  14985  14995  15001  15005  15009  15014
                    15019  15023  15028  15033  15038  15048  16606  16764  16769  16775  16781  16785  16792
                    16796  16800  16808  16813  16823  16831  16835  16873  16882  16886  16895  16907  16915
                    16923  16935  16949  16961  16976  16988  17000  17037  17038  17041  17044  17045  17048
                    17051  17056  17061  17066  17071  17075  17080  17087  17094  17100  17106  17112  17118
                    17123  17130  17137  17142  17148  17154  17159  17164  17169  17172  17176  17180  17184
                    17187  17190  17196  17200  17204  17207  17212  17216  17221  17225  17230  17234  17238
                    17241  17244  17250  17255  17259  17264  17268  17272  17277  17282  17287  17292  17296
                    17299  17305  17311  17316  17320  17324  17328  17332  17338  17344  17350  17354  17358
                    17363  17368  17372  17378  17382  17386  17390  17394  17398  17401  17405  17410  17415
                    17420  17425  17426  17427  17430  17436  17440  17444  17449  17454  17458  17462  17468
                    17471  17474  17477  17480  17482  17484  17486  17488
CRLF   = 000200      2708#  5013  14278  14317
CYLMSK= 001777       2984#  5343
DBCK   = 100000      2929#  8246   8401   8559   8657   8804   8910   8937   8964   9103   9170   9231   9300
                     9362   9415   9476   9535   9758   9854   9921  10023  10105  10449  10768  10836  10839
                    10951  11037  11098  11109  11150  11162  11200  11240  11254  11294  11306  11327  11381
                    11453  11464  11510  11522  11565  11610  11624  11669  11681  11702  11763  11842  11853
                    11899  11911  11957  12003  12020  12037  12082  12094  12115  12158  12175  12188  12206
                    12267  12347  12361  12371  12451  12463  12506  12518  12549  12581  12623  12643  12708
                    12720  12797  12821  12940  12998  13010  13019  13031  13110  13125  13193  13208  13220
                    13232  13305  13318  13328  13351  13492  13539  13566
DBEN   = 040000      2930#  7793   7800   7802   7845   7850   7852   7933   7938   7940   8020   8022   8163
                     8216   8246   8248   8359   8401   8403   8533   8559   8561   8628   8657   8659   8784
                     8804   8806   8850   8910   8912   8937   8939   8964   8966   9007   9081   9103   9105
                     9163   9170   9172   9222   9231   9233   9282   9300   9302   9349   9362   9364   9400
                     9415   9417   9470   9476   9478   9529   9535   9537   9636   9640   9642   9677   9747
                     9758   9760   9845   9854   9856   9915   9921   9923  10017  10023  10025  10088  10105
                    10107  10112  10114  10152  10162  10443  10449  10451  10557  10760  10768  10770  10827
                    10945  10951  10953  11029  11037  11039  11087  11098  11100  11105  11109  11111  11139
```

```
                    11150  11152  11157  11162  11164  11189  11200  11202  11229  11240  11242  11247  11249
                    11254  11256  11283  11294  11296  11301  11306  11308  11322  11327  11329  11354  11381
                    11383  11438  11453  11455  11460  11464  11466  11495  11510  11512  11517  11522  11524
                    11550  11565  11567  11595  11610  11612  11617  11619  11624  11626  11654  11669  11671
                    11676  11681  11683  11697  11702  11704  11731  11763  11765  11827  11842  11844  11849
                    11853  11855  11884  11899  11901  11906  11911  11913  11942  11957  11959  11988  12003
                    12005  12012  12015  12020  12022  12028  12030  12032  12037  12039  12067  12082  12084
                    12089  12094  12096  12110  12115  12117  12143  12158  12160  12167  12170  12175  12177
                    12188  12190  12201  12206  12208  12235  12267  12269  12332  12347  12349  12354  12356
                    12361  12363  12371  12373  12389  12436  12451  12453  12458  12463  12465  12491  12506
                    12508  12513  12518  12520  12529  12531  12549  12551  12560  12562  12581  12583  12608
                    12623  12625  12643  12645  12672  12708  12710  12715  12720  12722  12731  12733  12764
                    12797  12799  12804  12806  12821  12823  12907  12940  12942  12965  12998  13000  13005
                    13010  13012  13019  13021  13026  13031  13033  13042  13044  13081  13110  13112  13118
                    13120  13125  13127  13164  13193  13195  13201  13203  13208  13210  13215  13220  13222
                    13227  13232  13234  13243  13245  13272  13305  13307  13313  13318  13320  13328  13330
                    13346  13351  13353  13362  13364  13454  13492  13494  13506  13508  13539  13541  13554
                    13556  13566  13568
DBL    = 002000     3103#
DCK    = 100000     2890#  2907   6204   6235
DDISP  = 177570     2714#  3203   4973
DEBL   = 020000     2931#  7609   7623   7800   7850   7938   8020   9640   10112
DISPLA  001156      3203#  4973*  4981*  14387* 14491*
DISPRE  000174      3130#  4981
DLT    = 100000     3075#
DMD    = 000001     2943#  2962#  5444   5456   7061   7068   7073   7083   7086   7088   7104   7138   7149
                    7151   7174   7191   7193   7196   7225   7235   7237   7254   7271   7273   7276   7306
                    7339   7341   7380   7400   7402   7405   7434   7448   7452   7468   7486   7488   7491
                    7520   7522   7533   7549   7565   7567   7570   7607   7609   7611   7621   7623   7639
                    7654   7656   7659   7782   7793   7800   7802   7843   7845   7850   7852   7931   7933
                    7938   7940   8012   8020   8022   8055   8085   8111   8161   8163   8214   8216   8246
                    8248   8357   8359   8401   8403   8531   8533   8559   8561   8626   8628   8657   8659
                    8782   8784   8804   8806   8848   8850   8872   8910   8912   8937   8939   8964   8966
                    9005   9007   9079   9081   9103   9105   9163   9170   9172   9222   9231   9233   9280
                    9282   9300   9302   9347   9349   9362   9364   9398   9400   9415   9417   9470   9476
                    9478   9529   9535   9537   9636   9640   9642   9675   9677   9747   9758   9760   9845
                    9854   9856   9915   9921   9923   10017  10023  10025  10088  10105  10107  10112  10114
                    10150  10152  10162  10443  10449  10451  10495  10501  10551  10557  10592  10601  10615
                    10647  10689  10760  10768  10770  10827  10945  10951  10953  11027  11029  11037  11039
                    11087  11098  11100  11105  11109  11111  11139  11150  11152  11157  11162  11164  11189
                    11200  11202  11229  11240  11242  11247  11249  11254  11256  11283  11294  11296  11301
                    11306  11308  11322  11327  11329  11354  11381  11383  11438  11453  11455  11460  11464
                    11466  11495  11510  11512  11517  11522  11524  11550  11565  11567  11595  11610  11612
                    11617  11619  11624  11626  11654  11669  11671  11676  11681  11683  11697  11702  11704
                    11731  11763  11765  11827  11842  11844  11849  11853  11855  11884  11899  11901  11906
                    11911  11913  11942  11957  11959  11988  12003  12005  12012  12015  12020  12022  12028
                    12030  12032  12037  12039  12067  12082  12084  12089  12094  12096  12110  12115  12117
                    12143  12158  12160  12167  12170  12175  12177  12188  12190  12201  12206  12208  12235
                    12267  12269  12332  12347  12349  12354  12356  12361  12363  12371  12373  12389  12436
                    12451  12453  12458  12463  12465  12491  12506  12508  12513  12518  12520  12529  12531
                    12549  12551  12560  12562  12581  12583  12608  12623  12625  12643  12645  12672  12708
                    12710  12715  12720  12722  12731  12733  12764  12797  12799  12804  12806  12821  12823
                    12907  12940  12942  12965  12998  13000  13005  13010  13012  13019  13021  13026  13031
                    13033  13042  13044  13081  13110  13112  13118  13120  13125  13127  13164  13193  13195
                    13201  13203  13208  13210  13215  13220  13222  13227  13232  13234  13243  13245  13272
                    13305  13307  13313  13318  13320  13328  13330  13346  13351  13353  13362  13364  13454
```

DZRMJA.P11 01-AUG-77 11:10 CROSS REFERENCE TABLE -- USER SYMBOLS

```
                    13492  13494  13506  13508  13539  13541  13554  13556  13566  13568  13971  13973  14006
DPE    = 000010      3017#  6342   6366   6620
DPEHI  = 040000      3099#
DPELO  = 020000      3100#
DPR    = 000400      2883#  10267  10269
DRQ    = 004000      2969#  10227  10461
DRVCLR= 000010       2827#  8281   8704
DRY    = 000200      2884#  8385   8389   8424   8431
DSWR   = 177570      2713#  3202   4972
DTE    = 010000      2893#  2907   6212   6244
DTO    = 010000      2932#  12015  12020  12022  12028  12030  12032  12037  12039  12170  12175  12177  12188
                    12190  12201  12206  12208  12361  12363  12371  12373
DULPRT= 024024       2972#  5967   5970
DVA    = 004000      2812#  5997   5999   10258  10260
DVC    = 000200      3016#  5454   6412   6414   7313   7337   7344   7387   7391
EBL    = 020000      2949#  7600   7614   7626   7628   7644   7646   12534  12539  12569  12736  12741  13047
                    13052  13248  13253  13368  13373
ECH    = 000100      2899#  2907   6204   6235   15080  15081  15088  15089
ECI    = 004000      2977#  5345   6518
ECRC   = 001000      2953#
EDT1     074474      3388   3509   3517   3525   3533   3541   3549   3565   3573   3581   3589   3597   3605
                     3613   3621   3629   3637   3645   3661   3669   3677   3685   3693   3701   3709   3717
                     3725   3733   3741   3749   3765   3773   3781   3789   3797   3821   3829   3869   3877
                     3885   3901   3909   3925   3933   3949   3957   3965   3973   3981   3989   4005   4013
                     4029   4037   4053   4061   4069   4117   4125   4133   4141   4157   4165   4173   4181
                     4197   4205   4221   4229   4237   4245   4253   4269   4285   4293   4301   4309   4325
                     4333   4357   4381   4389   4397   4405   4413   4421   4429   4437   4485   4509   4517
                     4525   4565   4573   4581   4589   4597   4605   4638   4654   4670   4678   4687   4695
                     4703   4711   4719   4728   4736   4745   4753   4761   4769   4777   4785   4793   4802
                     4810   4818   4826   4834   4842   4850   4858   4866   4874   4882   4890   4898   4906
                     4915   4923   4931   16024#
EDT115   074512      3997   4021   4045   4077   4213   4261   4277   4341   4349   4365   4445   4461   4469
                     4477   4493   4501   4614   4622   4630   4646   4662   16031#
EDT130   074514      4085   4093   4189   16032#
EDT132   074516      4101   4109   4149   16033#
EDT2     074476      3396   16025#
EDT220   074520      4533   4541   4549   4557   16034#
EDT5     074500      3420   3428   3437   16026#
EDT57    074502      3757   16027#
EDT65    074504      3805   3813   3893   3917   3941   16028#
EDT71    074506      3837   3845   3853   16029#
EDT74    074510      3861   16030#
ED1      106524     16024  17007#
ED115    106600     16031  16033  17017#
ED130    106612     16032  17019#
ED2      106534     16025  16030  17009#
ED220    106626     16034  17021#
ED5      106540     16026  17010#
ED57     106546     16027  17011#
ED65     106560     16028  17013#
ED71     106570     16029  17015#
EECC   = 000020      2958#
EFT1     074522      3389   3510   3518   3526   3534   3542   3550   3566   3574   3582   3590   3598   3606
                     3614   3622   3630   3638   3646   3662   3670   3678   3686   3694   3702   3710   3718
                     3726   3734   3742   3750   3766   3774   3782   3790   3798   3822   3830   3870   3878
                     3886   3902   3910   3926   3934   3950   3958   3966   3974   3982   3990   4006   4014
```

```
                    4030    4038    4054    4062    4070    4118    4126    4134    4142    4158    4166    4174    4182
                    4198    4206    4222    4230    4238    4246    4254    4270    4286    4294    4302    4310    4326
                    4334    4358    4382    4390    4398    4406    4414    4422    4430    4438    4486    4510    4518
                    4526    4566    4574    4582    4590    4598    4606    4639    4655    4671    4679    4688    4696
                    4704    4712    4720    4729    4737    4746    4754    4762    4770    4778    4786    4794    4803
                    4811    4819    4827    4835    4843    4851    4859    4867    4875    4883    4891    4899    4907
                    4916    4924    4932   16036#
EFT115  074540      3998    4022    4046    4078    4214    4262    4278    4342    4350    4366    4446    4462    4470
                    4478    4494    4502    4615    4623    4631    4647    4663   16043#
EFT130  074542      4086    4094    4190   16044#
EFT132  074544      4102    4110    4150   16045#
EFT2    074524      3397   16037#
EFT220  074546      4534    4542    4550    4558   16046#
EFT5    074526      3421    3429    3438   16038#
EFT57   074530      3758   16039#
EFT65   074532      3806    3814    3894    3918    3942   16040#
EFT71   074534      3838    3846    3854   16041#
EFT74   074536      3862   16042#
EF1     106632     16036   16040   16041   17023#
EF130   106644     16044   17028#
EF2     106635     16037   16042   17024#
EF5     106636     16038   16046   17025#
EF57    106640     16039   16043   16045   17026#
EHT1    074400      3387    3508    3516    3524    3532    3540    3548    3564    3572    3580    3588    3596    3604
                    3612    3620    3628    3636    3644    3660    3668    3676    3684    3692    3700    3708    3716
                    3724    3732    3740    3748    3764    3772    3780    3788    3796    3820    3828    3868    3876
                    3884    3900    3908    3924    3932    3948    3956    3964    3972    3980    3988    4004    4012
                    4028    4036    4052    4060    4068    4116    4124    4132    4140    4156    4164    4172    4180
                    4196    4204    4220    4228    4236    4244    4252    4268    4284    4292    4300    4308    4324
                    4332    4356    4380    4388    4396    4404    4412    4420    4428    4436    4484    4508    4516
                     524    4564    4572    4580    4588    4596    4604    4637    4653    4669    4677    4686    4694
                    4702    4710    4718    4727    4735    4744    4752    4760    4768    4776    4784    4792    4801
                    4809    4817    4825    4833    4841    4849    4857    4865    4873    4881    4889    4897    4905
                    4914    4922    4930   16008#
EHT115  074440      3996    4020    4044    4076    4468    4476    4645   16016#
EHT130  074444      4084    4092   16017#
EHT132  074450      4100    4108    4148   16018#
EHT145  074454      4188   16019#
EHT150  074460      4212    4260    4276    4340    4348    4364    4444    4460    4621    4629    4661   16020#
EHT2    074404      3395   16009#
EHT213  074464      4492    4500    4613   16021#
EHT220  074470      4532    4540    4548    4556   16022#
EHT5    074410      3419    3427   16010#
EHT57   074420      3756   16012#
EHT65   074424      3804    3812   16013#
EHT7    074414      3436   16011#
EHT71   074430      3836    3844    3852    3892    3916    3940   16014#
EHT74   074434      3860   16015#
EH1     105231     16008   16873#
EH115   105636     16016   16923#
EH130   105734     16017   16935#
EH132   106053     16018   16949#
EH145   106152     16019   16961#
EH150   106270     16020   16976#
EH2     105310     16009   16882#
EH213   106367     16021   16988#
```

```
EH220   106465          16022   17000#
EH3     105317          16015   16884#
EH5     105326          16010   16886#
EH57    105404          16012   16895#
EH65    105503          16013   16907#
EH7     105365          16011   16892#
EH71    105560          16014   16915#
EMS1    074550          15245   15246   16048#
EMS10   075153          15422   15427   16094#
EMS11   075216          15432   16100#
EMS12   075307          15404   15407   15410   16110#
EMS13   075365          15440   15443   15446   15449   16118#
EMS14   075434          15453   15458   15466   16125#
EMS15   075501          15453   15458   15463   15581   15813   15817   15855   15860   15892   15924   15963   16132#
EMS16   075557          15468   15473   15478   15833   16140#
EMS17   075640          15468   15473   15481   16149#
EMS2    074621          15253   15545   15588   15639   15647   15656   15678   15839   15850   16055#
EMS20   075701          15483   15488   15492   15496   15501   15506   15513   16155#
EMS21   075745          15483   15488   15496   15501   15506   15510   15865   15897   15929   15968   16162#
EMS22   076024          15488   15492   15501   15506   16170#
EMS23   076071          15515   15520   15528   16177#
EMS24   076147          15515   15520   15525   15883   15915   15947   15999   16185#
EMS25   076226          15530   15535   15543   16193#
EMS250  101161          15291   15294   15297   15339   15562   15566   15570   15574   15702   15711   15745   15754   15792
                        16465#
EMS251  101217          15299   15302   15305   15765   15784   16471#
EMS252  101263          15307   15312   15314   15317   15320   15322   15325   15328   15330   15606   15610   15844   16478#
EMS253  101316          15307   15345   15348   15352   15356   15360   15364   15368   15606   15610   16483#
EMS254  101351          15307   15449   15463   15478   15510   15525   15540   15557   16488#
EMS255  101412          15332   15334   15337   15339   15578   15592   15702   15715   15731   15745   15758   16494#
EMS256  101454          15371   15374   15378   15382   15702   15707   16500#
EMS257  101504          15385   16504#
EMS26   076304          15530   15535   15540   15870   15902   15934   15994   16201#
EMS260  101540          15393   15397   15401   15432   16509#
EMS261  101571          15585   16514#
EMS262  101630          15765   15788   15800   15805   16520#
EMS27   076363          15545   15549   15553   15560   15736   15971   15973   15977   16002   16209#
EMS3    074670          15343   15621   15643   15651   15691   15698   15741   15747   15751   15833   16062#
EMS30   076427          15551   15553   15557   15570   15574   15592   15599   16216#
EMS300  101673          15245   15253   15312   15314   15317   15343   15348   15352   15404   15413   15417   15440   15545
                        15588   15639   15647   15656   15678   15702   15707   15711   15715   15839   15850   16527#
EMS301  101711          15246   15250   15257   15261   15264   15291   15294   15297   15299   15302   15305   15307   15320
                        15322   15325   15328   15330   15332   15334   15337   15339   15356   15360   15364   15368   15374
                        15378   15382   15393   15397   15401   15432   15449   15463   15478   15510   15525   15540   15557
                        15614   15694   16530#
EMS302  101734          15246   15250   15253   15264   16534#
EMS303  101761          15246   15432   16538#
EMS304  101772          15246   16540#
EMS306  102001          15250   15261   15307   15614   15694   15778   15781   16542#
EMS307  102015          15253   15545   15588   15639   15647   15656   15678   15839   15850   16545#
EMS31   076507          15562   16225#
EMS310  102056          15257   15603   15805   16551#
EMS311  102073          15257   15261   16554#
EMS312  102137          15260   15293   15296   15301   15304   15333   15336   15358   15362   15366   15376   15380   15395
                        15399   16561#
EMS313  102175          15264   15297   15305   15330   15337   15368   15382   15401   15449   15463   15478   15510   15525
```

```
                    15540   15557   16567#
EMS314  102226      15267   15273   15279   15285   15311   16572#
EMS315  102255      15270   15276   15282   15288   15488   15551   16576#
EMS316  102303      15267   15270   15273   15276   15279   15282   15285   15288   16580#
EMS317  102324      15267   15270   16583#
EMS32   076560      15566   16232#
EMS320  102344      15273   15276   16586#
EMS321  102364      15279   15282   16589#
EMS322  102404      15285   15288   16592#
EMS323  102424      15294   15302   15328   15334   15339   15364   15378   15397   16595#
EMS324  102443      15291   15299   15320   15322   15325   15332   15356   15360   15374   15393   16598#
EMS325  102463      15296   15304   15336   15366   15380   15399   16601#
EMS326  102474      15293   15301   15333   15358   15362   15376   15395   16603#
EMS327  102504      15307   15339   15417   15606   15672   15702   15745   15747   15765   16605#
EMS33   076640      15570   16240#
EMS330  102512      15311   16606#
EMS331  102531      15345   15371   15549   15578   16609#
EMS332  102561      15385   15466   15481   15513   15528   15543   15560   15824   15888   15920   15956   15990   16614#
EMS333  102577      15389   15415   15422   15492   15506   15654   15675   15683   15774   15784   15788   15792   15805
                    16617#
EMS334  102613      15427   15492   15501   15664   15765   15774   16620#
EMS335  102623      15407   15422   15664   15668   15683   15769   15868   15886   15900   15918   15932   15950   16622#
EMS336  102646      15410   15427   15860   15870   15892   15902   15924   15934   15963   15971   15994   16002   16626#
EMS337  102665      15446   15458   15473   15496   15501   15520   15535   15553   15599   15606   15625   15660   15664
                    15668   15672   15688   15719   15745   15762   15771   16629#
EMS34   076677      15574   15691   16246#
EMS340  102701      15422   15427   15683   15860   15870   15892   15902   15924   15934   15963   15994   16632#
EMS341  102712      15453   15458   15468   15473   15483   15496   15501   15515   15520   15530   15535   15553   15570
                    15574   15592   15599   15606   15704   15727   15711   15715   15719   15723   15727   15736   15796
                    16634#
EMS342  102721      15407   15410   15422   15427   15437   15824   16636#
EMS343  102753      15437   16641#
EMS344  102767      15437   16644#
EMS345  103016      15443   15446   15596   15731   15765   16648#
EMS346  103044      15443   15453   15468   15483   15515   15530   15596   15723   15727   15731   15736   15796   15800
                    16652#
EMS347  103062      15449   15463   15478   15510   15525   15540   15557   16655#
EMS35   076735      15581   15654   15664   15672   15688   16252#
EMS350  103103      15488   15492   15501   15506   15774   15805   15844   15855   16658#
EMS351  103110      15488   16659#
EMS352  103126      15506   16662#
EMS353  103142      15549   15621   15631   15635   15643   15651   15660   15698   15828   16664#
EMS354  103166      15562   15566   15610   15621   15628   15643   15651   15698   15741   15751   15754   15758   15833
                    16668#
EMS355  103221      15570   15574   15592   16673#
EMS356  103243      15813   15817   15821   15960   16677#
EMS357  103263      15581   15809   16680#
EMS36   076777      15588   15614   15625   15628   15631   15635   15683   15809   15868   15879   15886   15900   15911
                    15918   15932   15943   15950   15981   16258#
EMS360  103305      15581   15813   16684#
EMS361  103333      15817   16688#
EMS362  103361      15585   16692#
EMS363  103402      15603   15778   15781   15805   16695#
EMS364  103417      15610   15754   15758   16698#
EMS365  103440      15610   15621   15631   15635   15643   15651   15698   15824   15828   15833   16701#
EMS366  103445      15618   16702#
```

```
EMS367  103470          15621  15631  15635  15643  15651  15698  15828  16706#
EMS37   077027          15592  16263#
EMS370  103540          15631  15879  15911  15943  15973  16713#
EMS371  103555          15635  15977  16716#
EMS372  103604          15664  15668  15675  15809  15833  15844  15855  16720#
EMS373  103636          15731  15765  16725#
EMS374  103642          15745  16726#
EMS375  103673          15784  16731#
EMS376  103707          15788  15792  15800  15844  16734#
EMS377  103723          15824  15833  15844  15865  15883  15897  15915  15929  15947  15968  15999  16737#
EMS4    074735          15345  15371  15578  16069#
EMS40   077067          15596  15599  16269#
EMS400  103734          15855  16739#
EMS401  103751          15828  15846  15855  15875  15907  15939  15952  15985  16004  16742#
EMS402  103766          15865  15883  15897  15915  15929  15947  15968  15999  16745#
EMS403  103775          15865  15883  15897  15915  15929  15947  15968  15999  16747#
EMS404  104036          15870  15902  15934  15981  15994  16753#
EMS405  104055          15860  15870  15875  15879  15888  15892  15902  15907  15911  15920  15924  15934  15939
                        15943  15952  15956  15963  15973  15977  15981  15985  15990  15994  16004  16756#
EMS406  104065          15860  15892  15924  15963  16758#
EMS407  104100          15960  16760#
EMS41   077145          15603  15606  15610  15664  15668  15675  15821  15824  16277#
EMS42   077213          15618  15621  16284#
EMS43   077275          15628  16293#
EMS44   077346          15639  15643  16300#
EMS45   077407          15647  15651  16306#
EMS46   077464          15656  15660  16314#
EMS47   077546          15668  15672  15678  15698  16323#
EMS5    074766          15389  15778  16074#
EMS50   077606          15683  15691  16329#
EMS500  104120          15248  16764#
EMS501  104260          15248  15251  15255  15258  15262  15265  15268  15271  15274  15277  15280  15283  15286
                        15289  15300  15303  15306  15309  15313  15315  15318  15321  15323  15326  15329  15331
                        15341  15344  15346  15349  15353  15357  15361  15365  15369  15372  15375  15379  15383
                        15386  15390  15394  15398  15402  15405  15408  15411  15414  15419  15424  15429  15434
                        15455  15460  15470  15475  15485  15490  15494  15498  15504  15508  15517  15522  15532
                        15537  15583  15590  15604  15608  15612  15615  15626  15629  15653  15661  15666  15670
                        15674  15680  15685  15689  15692  15695  15700  15705  15709  15721  15725  15729  15733
                        15738  15748  15752  15768  15772  15776  15779  15782  15786  15790  15794  15798  15802
                        15807  15811  15815  15819  15822  15826  15830  15836  15847  15852  15857  16781#
EMS502  104305          15248  15251  15255  15258  15262  15265  15268  15271  15274  15277  15280  15283  15286
                        15289  15309  15424  15429  15434  15742  15841  15847  15987  16785#
EMS503  104354          15248  15255  15258  15262  15265  15309  15318  15326  15341  15349  15357  15441  15444
                        15447  15451  15455  15460  15465  15470  15475  15480  15485  15498  15512  15517  15522
                        15527  15532  15537  15542  15547  15550  15555  15559  15563  15567  15586  15615  15619
                        15623  15633  15637  15641  15645  15649  15653  15658  15661  15680  15685  15742  15802
                        15830  15836  15841  15847  15852  15857  15863  15867  15873  15877  15881  15885  15890
                        15895  15899  15905  15909  15913  15917  15922  15927  15931  15937  15941  15945  15949
                        15954  15958  15961  15966  15970  15975  15979  15983  15987  15992  15997  16001  16006
                        16792#
EMS504  104400          15258  15262  15265  15292  15295  15298  15315  15323  15335  15338  15341  15353  15361
                        15419  15563  15567  15572  15576  15579  15594  15597  15601  15695  15713  15717  15733
                        15738  15748  15756  15760  15763  16796#
EMS505  104424          15251  16800#
EMS506  104477          15386  16808#
EMS507  104534          15390  16813#
```

F12


```
EMS51   077655          15694   15727   16336#
EMS510  104624          15742   16823#
EMS511  104677          15248   15251   15255   15258   15262   15265   15268   15271   15274   15277   15280   15283   15286
                        15289   15292   15295   15298   15300   15303   15306   15309   15313   15315   15318   15321   15323
                        15326   15329   15331   15335   15338   15341   15344   15346   15349   15353   15357   15361   15365
                        15369   15372   15375   15379   15383   15386   15390   15394   15398   15402   15405   15408   15411
                        15414   15419   15424   15429   15434   15441   15444   15447   15451   15455   15460   15465   15470
                        15475   15480   15485   15490   15494   15498   15504   15508   15512   15517   15522   15527   15532
                        15537   15542   15547   15550   15555   15559   15563   15567   15572   15576   15579   15583   15586
                        15590   15594   15597   15601   15604   15608   15612   15615   15619   15623   15626   15629   15633
                        15637   15641   15645   15649   15653   15658   15661   15666   15670   15674   15680   15685   15689
                        15692   15695   15700   15705   15709   15713   15717   15721   15725   15729   15733   15738   15742
                        15748   15752   15756   15760   15763   15768   15772   15776   15779   15782   15786   15790   15794
                        15798   15802   15807   15811   15815   15819   15822   15826   15830   15836   15841   15847   15852
                        15857   15863   15867   15873   15877   15881   15885   15890   15895   15899   15905   15909   15913
                        15917   15922   15927   15931   15937   15941   15945   15949   15954   15958   15961   15966   15970
                        15975   15979   15983   15987   15992   15997   16001   16006   16831#
EMS52   077724          15719   15723   15731   15736   16343#
EMS53   077765          15741   15985   16349#
EMS54   100026          15745   15751   15754   15758   16355#
EMS55   100103          15762   16363#
EMS56   100161          15765   15769   15771   15844   16371#
EMS57   100254          15774   16382#
EMS6    075036          15413   15417   15422   15427   16081#
EMS60   100340          15774   16391#
EMS600  104776          15704   15707   15711   15715   16843#
EMS601  105026          15719   16847#
EMS602  105046          15723   15727   16850#
EMS603  105107          15796   16856#
EMS604  105130          15860   15865   15870   15875   15879   15883   15888   16859#
EMS605  105155          15892   15897   15902   15907   15911   15915   15920   16863#
EMS606  105173          15924   15929   15934   15939   15943   15947   15952   15956   16866#
EMS607  105213          15963   15968   15973   15977   15981   15985   15990   15994   15999   16004   16869#
EMS61   100412          15781   16398#
EMS62   100465          15784   15788   15792   15796   16406#
EMS63   100546          15800   15805   15809   15813   15817   15821   15824   15828   15875   15907   15939   16415#
EMS64   100606          15833   16421#
EMS65   100656          15839   15846   16428#
EMS66   100717          15850   15855   15860   15870   15892   15902   15924   15934   15960   15963   15994   16434#
EMS67   101001          15888   15920   15956   15990   16443#
EMS7    075102          15415   15417   16087#
EMS70   101113          15952   16004   16457#
EMTVEC= 000030           2802#   4956*   4957*   5200*
EMT1    066522          3386    15245#
EMT10   066710          3443    15267#
EMT100  070324          3891    15449#
EMT101  070344          3899    15453#
EMT102  070364          3907    15458#
EMT103  070404          3915    15463#
EMT104  070430          3923    15468#
EMT105  070450          3931    15473#
EMT106  070470          3939    15478#
EMT107  070514          3947    15483#
EMT11   066726          3451    15270#
EMT110  070534          3955    15488#
EMT111  070556          3963    15492#
```

```
EMT112  070576          3971   15496#
EMT113  070616          3979   15501#
EMT114  070642          3987   15506#
EMT115  070664          3995   15510#
EMT116  070710          4003   15515#
EMT117  070730          4011   15520#
EMT12   066744          3459   15273#
EMT120  070750          4019   15525#
EMT121  070774          4027   15530#
EMT122  071014          4035   15535#
EMT123  071034          4043   15540#
EMT124  071060          4051   15545#
EMT125  071076          4059   15549#
EMT126  071116          4067   15553#
EMT127  071134          4075   15557#
EMT13   066762          3467   15276#
EMT130  071160          4083   15562#
EMT131  071176          4091   15566#
EMT132  071214          4099   15570#
EMT133  071234          4107   15574#
EMT134  071254          4115   15578#
EMT135  071270          4123   15581#
EMT136  071306          4131   15585#
EMT137  071320          4139   15588#
EMT14   067000          3475   15279#
EMT140  071336          4147   15592#
EMT141  071356          4155   15596#
EMT142  071372          4163   15599#
EMT143  071410          4171   15603#
EMT144  071424          4179   15606#
EMT145  071446          4187   15610#
EMT146  071470          4195   15614#
EMT147  071506          4203   15618#
EMT15   067016          3483   15282#
EMT150  071520          4211   15621#
EMT151  071542          4219   15625#
EMT152  071554          4227   15628#
EMT153  071570          4235   15631#
EMT154  071610          4243   15635#
EMT155  071630          4251   15639#
EMT156  071646          4259   15643#
EMT157  071670          4267   15647#
EMT16   067034          3491   15285#
EMT160  071706          4275   15651#
EMT161  071736          4283   15656#
EMT162  071754          4291   15660#
EMT163  071772          4299   15664#
EMT164  072014          4307   15668#
EMT165  072036          4315   15672#
EMT166  072062          4323   15678#
EMT167  072102          4331   15683#
EMT17   067052          3499   15288#
EMT170  072124          4339   15688#
EMT171  072136          4347   15691#
EMT172  072152          4355   15694#
EMT173  072170          4363   15698#
```

```
EMT174  072212          4371   15702#
EMT175  072240          4379   15707#
EMT176  072256          4387   15711#
EMT177  072274          4395   15715#
EMT2    066530          3394   15246#
EMT20   067070          3507   15291#
EMT200  072312          4403   15719#
EMT201  072330          4411   15723#
EMT202  072346          4419   15727#
EMT203  072364          4427   15731#
EMT204  072406          4435   15736#
EMT205  072426          4443   15741#
EMT206  072446          4451   15745#
EMT207  072476          4459   15751#
EMT21   067110          3515   15294#
EMT210  072512          4467   15754#
EMT211  072530          4475   15758#
EMT212  072546          4483   15762#
EMT213  072560          4491   15765#
EMT214  072610          4499   15771#
EMT215  072622          4507   15774#
EMT216  072642          4515   15778#
EMT217  072656          4523   15781#
EMT22   067130          3523   15297#
EMT220  072672          4531   15784#
EMT221  072710          4539   15788#
EMT222  072726          4547   15792#
EMT223  072744          4555   15796#
EMT224  072762          4563   15800#
EMT225  073002          4571   15805#
EMT226  073024          4579   15809#
EMT227  073042          4587   15813#
EMT23   067144          3531   15299#
EMT230  073060          4595   15817#
EMT231  073076          4603   15821#
EMT232  073112          4612   15824#
EMT233  073134          4620   15828#
EMT234  073156          4628   15833#
EMT235  073204          4636   15839#
EMT236  073224          4644   15844#
EMT237  073256          4652    4660   15850#
EMT24   067164          3539   15302#
EMT240  073276         15855#
EMT241  073322          4668   15860#
EMT242  073346          4676   15865#
EMT243  073372          4685   15870#
EMT244  073416          4693   15875#
EMT245  073434          4701   15879#
EMT246  073452          4709   15883#
EMT247  073476          4717   15888#
EMT25   067204          3547   15305#
EMT250  073514          4726   15892#
EMT251  073540          4734   15897#
EMT252  073564          4743   15902#
EMT253  073610          4751   15907#
EMT254  073626          4759   15911#
```

```
EMT255  073644          4767   15915#
EMT256  073670          4775   15920#
EMT257  073706          4783   15924#
EMT26   067220          3555   15307#
EMT260  073732          4791   15929#
EMT261  073756          4800   15934#
EMT262  074002          4808   15939#
EMT263  074020          4816   15943#
EMT264  074036          4824   15947#
EMT265  074062          4832   15952#
EMT266  074100          4840   15956#
EMT267  074116          4848   15960#
EMT27   067252          3563   15312#
EMT270  074132          4856   15963#
EMT271  074156          4864   15968#
EMT272  074202          4872   15973#
EMT273  074220          4880   15977#
EMT274  074236          4888   15981#
EMT275  074254          4896   15985#
EMT276  074274          4904   15990#
EMT277  074312          4913   15994#
EMT3    066556          3402   15250#
EMT30   067264          3571   15314#
EMT300  074336          4921   15999#
EMT301  074362          4929   16004#
EMT31   067300          3579   15317#
EMT32   067314          3587   15320#
EMT33   067330          3595   15322#
EMT34   067346          3603   15325#
EMT35   067364          3611   15328#
EMT36   067400          3619   15330#
EMT37   067414          3627   15332#
EMT4    066576          3410   15253#
EMT40   067430          3635   15334#
EMT41   067450          3643   15337#
EMT42   067464          3651   15339#
EMT43   067510          3659   15343#
EMT44   067522          3667   15345#
EMT45   067536          3675   15348#
EMT46   067552          3683   15352#
EMT47   067566          3691   15356#
EMT5    066620          3418   15257#
EMT50   067610          3699   15360#
EMT51   067632          3707   15364#
EMT52   067652          3715   15368#
EMT53   067666          3723   15371#
EMT54   067702          3731   15374#
EMT55   067722          3739   15378#
EMT56   067742          3747   15382#
EMT57   067756          3755   15385#
EMT6    066644          3426   15261#
EMT60   067772          3763   15389#
EMT61   070006          3771   15393#
EMT62   070026          3779   15397#
EMT63   070046          3787   15401#
EMT64   070062          3795   15404#
```

```
EMT65   070074          3803   15407#
EMT66   070110          3811   15410#
EMT67   070124          3819   15413#
EMT7    066666          3435   15264#
EMT70   070142          3827   15417#
EMT71   070162          3835   15422#
EMT72   070206          3843   15427#
EMT73   070232          3851   15432#
EMT74   070253          3859   15437#
EMT75   070213          3867   15440#
EMT76   070274          3875   15443#
EMT77   070310          3883   15446#
ENRGDT  066522         15242#
EQUALS  065133         14975#
ERR   = 040000          2877#   8062   8066   8074   8094   8120   8121   9017
ERRNMB  060052         13732*  13733*  13736  13748  13753  13845#
ERRTYP  057314         13711#  14501
ERRVEC= 000004          2795#   4970   4971*  4982*  6935   6936   6937*  6938*  6944*  6945*  6950*  6951*  6976
                        6977   6978*  6979*  6988*  6989*  7000*  7001*  13865  13866  13867*  13868*  13883*  13904*
                       13905*  14338  14339*  14341*  14344*
ERTY00  060060         13718  13850#
ERTY01  060066         13726  13851#
ERTY02  060076         13735  13853#
ERTY03  060105         13741  13855#
ERTY04  060113         13837  13857#
ESRC  = 004000          2951#  12185  12194  12376  12381  13333  13338
FER   = 000020          2901#   2907   6204   6235
FMT16 = 010000          2976#   5345   6518   7704   7722   7742   7744   7873   7875   7958   7960   9631
FNCDTB  065764          8810   9178   9241   9863  15058#
FNCMSK= 000077          2819#   8178   8237   8374   8406   8549   8564   8645
F0    = 000002          2817#
F1    = 000004          2816#
F2    = 000010          2815#
F3    = 000020          2814#
F4    = 000040          2813#
GNS   = ****** U        3129   5012  13670  13677  14856  14857  14858  14859  14860  14861  14863  14865  14866
                       14867  14868  14869  14870
GO    = 000001          2818#   7025   7030   7795   7847   7935   8018   8170   8175   8236   8367   8372   8383
                        8412   8418   8420   8425   8541   8547   8567   8573   8575   8644   8800   8864   8906
                        8933   8960   9096   9165   9224   9296   9358   9411   9472   9531   9638   9685   9750
                        9847   9917  10019  10100  10159  10300  10347  10394  10445  10560  10762  10832  10947
                       11031  11093  11145  11169  11195  11235  11289  11314  11316  11334  11360  11448  11505
                       11529  11560  11605  11664  11689  11691  11709  11741  11837  11894  11918  11952  11998
                       12077  12102  12104  12122  12153  12213  12245  12342  12446  12501  12588  12618  12682
                       12774  12917  12975  13087  13170  13282  13407  13979  14012
GTSWR = 104407          5007   5196  14863#
HCE   = 000200          2898#   2907   6204   6235  15080  15081  15084  15088  15089
HCI   = 002000          2978#   5345   6518
HCRC  = 000400          2897#   2907   6204   6235
HELP    106652          5060  17036#
HELPQS  065143          5054  14978#
HT    = 000011          2705#  13774  14276  14317
IAE   = 002000          2895#   6204   6235   9766   9862   9864   9929   9931  10031  10033  15062  15072  15080
                       15081  15084  15085  15088  15089
IDXMSK= 000077          3057#
IE    = 000100          3071#   3104#
```

```
ILF    = 000001          2905#  6196   6227   9044   9177   9179  15061  15070  15071  15073  15074  15075  15076
                        15077  15078  15079  15082  15083  15086  15087  15090  15091
ILF02  = 000002          2824#  8448   8692
ILF24  = 000024          2834#  8463   8728   9041
ILF26  = 000026          2835#  8466   8732
ILF30  = 000030          2837#
ILF32  = 000032          2838#  8321   8595   8740
ILF34  = 000034          2839#  8469   8744
ILF36  = 000036          2840#  8472   8748
ILF40  = 000040          2841#
ILF42  = 000042          2842#  8475
ILF44  = 000044          2843#  8478
ILF46  = 000046          2844#  8481
ILF54  = 000054          2847#  8484
ILF56  = 000056          2848#  8487
ILF64  = 000064          2851#  8490
ILF66  = 000066          2852#  8493
ILF74  = 000074          2855#  8496
ILF76  = 000076          2856#  5339   5360   5392   5408   5482   5519   5555   5580   5820   5827   6103   6108
                         6118   6123   6135   6138   6139   6153   6160   8185   8366   8499   8540   8819   9095
                         9189   9254   9706   9749   9875  11053
ILR    = 000002          2904#  6196   6227   6923   6961   6965   6986   6992
ILRG50= 000050           3043#
ILRG52= 000052           3044#
ILRG54= 000054           3045#
ILRG56= 000056           3046#
ILRG60= 000060           3047#
ILRG62= 000062           3048#
ILRG64= 000064           3049#
ILRG66= 000066           3050#
ILRG70= 000070           3051#
ILRG72= 000072           3052#
ILRG74= 000074           3053#
ILRG76= 000076           3054#
IOTVEC= 000020           2800#  4954*  4955*
IPCK0  = 000001          3108#
IPCK1  = 000002          3107#
IPCK2  = 000004          3106#
IPCK3  = 000010          3105#
IR     = 000100          3085#  5260   5281
IVC    = 010000          3013#  6335   6359   8791   8809   8811  15061  15062  15063  15065  15066  15067  15070
                        15071  15072  15073  15074  15075  15076  15077  15078  15079  15080  15081  15082  15083
                        15084  15085  15086  15087  15088  15089  15090  15091
LBC    = 002000          3015#  6342   6366
LBT    = 002000          2881#  8000   8025   8027
LCLOCK  060374          13885  13914#
LCOUNT  060444          13887  13932#
LF     = 000012          2706#  13772  14311  14317  14976  14979  14985  14995  15001  15005  15009  15014  15019
                        15023  15028  15033  15038  15048  16606  16764  16769  16775  16781  16785  16792  16796
                        16800  16808  16813  16823  16831  16835  16873  16882  16886  16895  16907  16915  16923
                        16935  16949  16961  16976  16988  17000  17037  17038  17041  17044  17045  17048  17051
                        17056  17061  17066  17071  17075  17080  17087  17094  17100  17106  17112  17118  17123
                        17130  17137  17142  17148  17154  17159  17164  17169  17172  17176  17180  17184  17187
                        17190  17196  17200  17204  17207  17212  17216  17221  17225  17230  17234  17238  17241
                        17244  17250  17255  17259  17264  17268  17272  17277  17282  17287  17292  17296  17299
                        17305  17311  17316  17320  17324  17328  17332  17338  17344  17350  17354  17358  17363
```

```
                    17368   17372   17378   17382   17386   17390   17394   17398   17401   17405   17410   17415   17420
                    17425   17426   17427   17430   17436   17440   17444   17449   17454   17458   17462   17468   17471
                    17474   17477   17480   17482   17484   17486   17488
LS     = 000004      2960#   7702    7708    7712
LSC    = 004000      3014#   6335    6359    8857    8875    8877
LST    = 000002      2961#   7720    7726    7730
LSTOP    060506     13886   13946#
MCLK   = 004000      2933#  12529   12560   12731   13042   13243   13362
MCPE   = 020000      3065#   6692
MDF    = 000100      2938#   7341    7389   11157   11162   11164   11517   11522   11524   11906   11911   11913   12201
                    12206   12208   12513   12518   12520   12529   12531   12549   12551   12560   12562   12581   12583
                    12715   12720   12722   12731   12733   13042   13044   13227   13232   13234   13243   13245   13346
                    13351   13353   13362   13364
MDPE   = 000400      3082#
MI     = 000004      2941#  12012   12167
MOC    = 000400      2936#   7522    7556   11229   11240   11242   11249   11254   11256   11283   11294   11296   11354
                    11383   11595   11610   11612   11619   11624   11626   11654   11669   11671   11731   11765   11988
                    12003   12005   12015   12020   12022   12028   12030   12032   12037   12039   12067   12082   12084
                    12143   12158   12160   12170   12175   12177   12188   12190   12201   12206   12208   12235   12267
                    12269   12332   12347   12349   12356   12361   12363   12371   12373   12389   12436   12451   12453
                    12458   12463   12465   12491   12506   12508   12513   12518   12520   12529   12531   12549   12551
                    12560   12562   12581   12583   12608   12623   12625   12643   12645   12672   12708   12710   12715
                    12720   12722   12731   12733   12764   12797   12799   12806   12821   12823   12907   12940   12942
                    12965   12998   13000   13042   13044   13081   13110   13112   13120   13125   13127   13164   13193
                    13195   13203   13208   13210   13243   13245   13272   13305   13307   13318   13320   13328   13330
                    13346   13351   13353   13362   13364   13454   13492   13494   13508   13539   13541   13556   13566
                    13568
MOH    = 020000      2968#
MOL    = 010000      2879#   7141    7154    7156    7179    7183
MRD    = 002000      2934#
MS     = 000040      2939#  12030
MSC    = 000002      2942#  12028   12030   12032
MSE    = 100000      3022#
MSER   = 000200      2937#   7452    7477   11322   11327   11329   11697   11702   11704   12110   12115   12117   13026
                    13031   13033
MUR    = 001000      2935#   7151    7181    8163    8216    8246    8248    8359    8401    8403    8533    8559    8561
                     8628    8657    8659    8784    8804    8806    8910    8912    8937    8939    8964    8966    9007
                     9081    9103    9105    9163    9170    9172    9222    9231    9233    9282    9300    9302    9349
                     9362    9364    9400    9415    9417    9470    9476    9478    9529    9535    9537    9636    9640
                     9642    9747    9758    9760    9845    9854    9856    9915    9921    9923   10017   10023   10025
                    10088   10105   10107   10112   10114   10152   10162   10443   10449   10451   10501   10557   10601
                    10760   10768   10770   10827   10945   10951   10953   11087   11098   11100   11139   11150   11152
                    11157   11162   11164   11189   11200   11202   11229   11240   11242   11247   11249   11254   11256
                    11283   11294   11296   11301   11306   11308   11322   11327   11329   11354   11381   11383   11438
                    11453   11455   11495   11510   11512   11517   11522   11524   11550   11565   11567   11595   11610
                    11612   11617   11619   11624   11626   11654   11669   11671   11676   11681   11683   11697   11702
                    11704   11731   11763   11765   11827   11842   11844   11884   11899   11901   11906   11911   11913
                    11942   11957   11959   11988   12003   12005   12012   12015   12020   12022   12028   12030   12032
                    12037   12039   12067   12082   12084   12089   12094   12096   12110   12115   12117   12143   12158
                    12160   12167   12170   12175   12177   12188   12190   12201   12206   12208   12235   12267   12269
                    12332   12347   12349   12354   12356   12361   12363   12371   12373   12389   12436   12451   12453
                    12491   12506   12508   12513   12518   12520   12529   12531   12549   12551   12560   12562   12581
                    12583   12608   12623   12625   12643   12645   12672   12708   12710   12715   12720   12722   12731
                    12733   12764   12797   12799   12804   12806   12821   12823   12907   12940   12942   12965   12998
                    13000   13005   13010   13012   13019   13021   13026   13031   13033   13042   13044   13081   13110
                    13112   13118   13120   13125   13127   13164   13193   13195   13201   13203   13208   13210   13215
```

M12


```
                    13220   13222   13227   13232   13234   13243   13245   13272   13305   13307   13313   13318   13320
                    13328   13330   13346   13351   13353   13362   13364   13454   13492   13494   13506   13508   13539
                    13541   13554   13556   13566   13568   13973   14006
MMD   = 000010       2959#
MMP   = 000010       2940#   7237    7261    10871   10879   10883
MXF   = 001000       3081#
NOTMSK= 115760       2907#
NED   = 010000       3078#   5256    5271    5277    5290
NEM   = 004000       3079#
NOP   = 000000       2823#   8273    8277    8688
NSA   = 100000       2966#
OCC   = 100000       2947#   8971    9031    9239    9243
OFD   = 000200       2979#   5345    6448    6450    6518    9515    9551    13406   13522
OFFSET= 000014       2829#   8457    8712    9472    10794   14012
OM    = 000001       2886#   9458    9484    9543    9593    9645    14015   14017
ONES    066134      15119#
OPE   = 020000       3012#   6335    6359    12949
OPI   = 020000       2892#   6212    6244    11020   11044   11046   11117   11119   11208   11212   11472   11474   11573
                    11577   11861   11863   11965   11969   12395   12400   12470   12472   12947   13133   13135   15060
                    15061   15062   15063   15064   15065   15066   15067   15068   15069   15070   15071   15072   15073
                    15074   15075   15076   15077   15078   15079   15080   15081   15082   15083   15084   15085   15086
                    15087   15088   15089   15090   15091
OR    = 000200       3084#
PACACK= 000022       2833#   8906    13979   13989
PAKACK= 000022       2832#   2833    8305    8724    9129
PAR   = 000010       2902#   6196    6279    6281    6616    6635    6651
PAT   = 000020       3087#   6638    6647    6650
PCLOCK  060356      13872   13910#
PCOUNT  060444      13874   13931#
PDA   = 000400       2954#
PGE   = 002000       3080#
PGM   = 001000       2882#   10229   10312   10359   10406
PHA   = 000200       2955#
PIP   = 020000       2878#   7525    7536    7538    7554    7558
PIRQ  = 177772       2712#
PIRQVE= 000240       2806#
PLCLK   060402      13912   13916#
PLFS  = 002000       2952#
PLSTP   060512      13944   13948#
PROMPT  065135       5133    5141   14976#
PR0   = 000000       2729#
PR1   = 000040       2730#
PR2   = 000100       2731#
PR3   = 000140       2732#   4989    5200
PR4   = 000200       2733#
PR5   = 000240       2734#   13923
PR6   = 000300       2735#   6938    6979    13868   13875   13888   13921
PR7   = 000340       2736#   13920
PS    = 177776       2709#   2710
PSEL  = 002000       3067#
PSTOP   060500      13873   13943#
PSW   = 177776       2710#
PWRVEC= 000024       2801#   4960*   4961*   14875*  14876*  14885*  14891*  14903*  14904*
QSTMRK  065141       5049    5151   14977#
RD    = 000070       2853#   8317    9803    10878   12446   12501   12618   12682   12774   12917   12975   13087   13170
                    13282   13407
```

DZRMJA.P11 01-AUG-77 11:10 CROSS REFERENCE TABLE -- USER SYMBOLS

```
RDCHR = 104411          5041   5055   5065   5134   5142   14728  14866#
RDLIN = 104412         14798  14867#
RDOCT = 104413          5077   5093   5114  14868#
RDY   = 000200          3070#
READY   006212          5046   5201#  13645  13693
RECAL = 000006          2826#  8454   8700  11093  11145  11195  11235  11289  11360
RESREG= 104415         13841  14870#
RESVEC= 000010          2796#
REX   = 010000          2950#  10926  10964  10969
RG    = 040000          2948#  9693   9710  12631  12633  12689  12694  12781  12786  12924  12929  12982  12987
                       13094  13099  13177  13182  13289  13294  13467  13472
RGDTPT  066074         15102#
RH    = 000072          2854#  8325   9794
RIP   = 000020          2831#  8297   8720   8933   9126   9296   9358   9411  10882
RLEASE= 000012          2828#  8289   8589   8708  10300  10347  10394  10445
RMAS  = 000016          3031#  5304   5758*  10189  10355*  10508*  10522  10528  10598*  10613*  10653*  10693*  10718
RMASI   001344          3297#
RMASO   001420          3327#
RMBA  = 000004          3113#  5300   10096*  13451*
RMBAE = 000050          3116#  6974   6983
RMBAEI  001376          3310#  6983*  6984*
RMBAEO  001452          3340#
RMBAI   001332          3292#
RMBAO   001406          3322#  13409*  13451
RMCS1 = 000000          3027#  3111#  5339*  5350   5392*  5402   5482*  5493*  5506  5555*  5568  5820*  5996
                        6103*  6107   6112   6118*  6120*  6122   6127   6133*  6135*  6137   6143  6155*  6157*
                        6159   6165   6691   6981*  7025*  7029   7795*  7847*  7935*  8018*  8172*  8174   8239*
                        8369*  8371   8405   8544*  8546   8563   8647*  8802*  8864*  8906*  8933*  8960*  9041*
                        9098*  9167*  9226*  9296*  9358*  9411*  9472*  9531*  9638*  9689*  9752*  9849*  9917*
                       10019*  10100*  10159*  10257  10300*  10308  10347*  10394*  10445*  10560*  10724  10764*  10834*
                       10947*  11033*  11093*  11145*  11168  11195*  11235  11289*  11313  11333  11360*  11448*  11505*
                       11528  11560*  11605*  11664*  11688  11708  11741*  11837*  11894*  11917  11952*  11998*  12077*
                       12101  12121  12153*  12212  12245*  12342*  12446*  12501*  12587  12618*  12682*  12774*  12917*
                       12975*  13087*  13170*  13282*  13460*  13979*  14012*
RMCS1I  001326          3290#  5350*  5360*  5402*  5408*  5410  5506*  5519*  5523*  5529  5568*  5580*  5585
RMCS10  001402          3320#  13407*  13460
RMCS2 = 000010          3114#  5252*  5253*  5255  5262  5271  5273*  5274*  5276  5283  5290  5302  5335*
                        5336*  5388*  5389*  5398*  5399*  5436*  5437*  5478*  5479*  5543*  5544*  5604*  5605*
                        5726*  5727*  5749*  5750*  5781*  5782*  5813*  5814*  5837*  5838*  5869*  5870*  5892*
                        5893*  5914*  5915*  5993*  5994*  6020*  6021*  6098*  6099*  6104*  6105*  6191*  6192*
                        6273*  6274*  6321*  6322*  6403*  6404*  6442*  6443*  6564*  6565*  6610*  6611*  6630*
                        6683*  6684*  6720*  6721*  6763*  6764*  6832*  6833*  6919*  6920*  6940  7020*  7021*
                        7026*  7027*  7057*  7058*  7131*  7132*  7221*  7222*  7297*  7298*  7426*  7427*  7512*
                        7513*  7596*  7597*  7679*  7680*  7779*  7780*  7836*  7837*  7903*  7904*  7924*  7925*
                        7990*  7991*  8048*  8049*  8082*  8083*  8108*  8109*  8158*  8159*  8211*  8212*  8354*
                        8355*  8528*  8529*  8623*  8624*  8779*  8780*  8845*  8846*  9002*  9003*  9076*  9077*
                        9277*  9278*  9344*  9345*  9395*  9396*  9672*  9673*  10147*  10148*  10220*  10221*  10254*
                       10255*  10492*  10493*  10548*  10549*  10589*  10590*  10644*  10645*  10686*  10687*  10922*  10923*
                       11016*  11017*  13968*  13969*
RMCS2I  001336          3294#
RMCS20  001412          3324#  6603*  6604*  6630  6638  6647  6650*
RMCS3 = 000052          3117#  6990
RMCS3I  001400          3311#
RMCS30  001454          3341#
RMDA  = 000006          3028#  5341*  5352  5484*  5495*  5508  5548*  5557*  5570  5608*  5619  5752*  5764
                        5784*  5796  5872*  5880  5897*  5905  5927*  6030*  6032*  6044*  6046*  6064*  6066*
```

```
                    6725*  6729*  6733   6767*  6769*  6771   6776   6782*  6784*  6786   6791   6800*  6802*
                    6804   6809   7698*  7766   7788*  7804   7841*  7855   7929*  7995*  9289*  9307   9406*
                    9422   9426   9633*  9740*  9838*  9912*  10012* 10092* 10195  10402* 10721  10941* 11440*
                   11497* 11552* 11597* 11656* 11733* 11829* 11886* 11944* 11990* 12069* 12145* 12237* 12340*
                   12442* 12497* 12614* 12678* 12770* 12913* 12973* 13070* 13153* 13278* 13436*
RMDAI   001334      3293*  5352*  5362   5508*  5524*  5530   5570*  5586   5619*  5631   5764*  5769   5796*
                    5801   5880*  5883   5905*  5908   6733*  6736
RMDAO   001410      3323*  7825*  7841   7866*  7868*  7909*  7929   7961*  13403* 13404* 13436  13530
RMDB  = 000022      3115*  5306
RMDBI   001350      3299*
RMDBO   001424      3329*
RMDC  = 000034      3037*  5343*  5354   5501*  5514   5552*  5561*  5574   5612*  5623   5731*  5739   5818*
                    5826   5844*  5856   5899*  5919*  5931   6727*  6731*  6735   6836   6840*  6842   6850*
                    6852*  6654   6862*  6864*  6866   6881*  6883*  6887   7907   7912*  7942   7997*  9291*
                    9311   9408*  9431   9435   9467*  9590*  9742*  9840*  9908*  10014* 10094* 10943* 11442*
                   11499* 11554* 11599* 11658* 11735* 11831* 11888* 11946* 11992* 12071* 12147* 12239* 12338*
                   12444* 12499* 12616* 12680* 12772* 12915* 12971* 13072* 13155* 13280* 13438*
RMDCI   001362      3304*  5354*  5364*  5514*  5521*  5527*  5533   5574*  5582*  5588   5623*  5641   5739*
                    5742*  5743   5826*  5830*  5831   5856*  5862*  5863   5931*  5936   5937   6735*  6738*
                    6739
RMDCO   001436      3334*  13405* 13438  13488
RMDS  = 000012      3029*  5876*  5923*  7134   7140   7153   7165   7178   7219   7227   7239   7258   7516
                    7524   7535   7553   7983   7999   8024   8051   8061   8073   8093   8119   8388   8393
                    8430   8433   8916   8923   8943   8950   9014   9016   9023   9072   9087   9109   9457
                    9483   9506   9542   9576   9592   9617   9644   10225  10233  10[illegible]  10271  10311  10358
                   10405  10511  10515  10562  10567  10603  10607  10617  10621  10[illegible]  10663  10699  10706
                   10775  10779  11261  11266  11631  11636  12044  12049  13981  13985  14014  14019
RMDSI   001340      3295*
RMDSO   001414      3325*
RMDT  = 000026      3034*  5846*  5964   5972   10223  10456
RMDTI   001354      3301*
RMDTO   001430      3331*
RMEC1 = 000044      3041*  5850*
RMEC1I  001372      3308*
RMEC1O  001446      3338*
RMEC2 = 000046      3042*  5308   5794*  6952
RMEC2I  001374      3309*
RMEC2O  001450      3339*
RMER1 = 000014      3030*  5394*  5404   5440*  5447   5436*  5497*  5510   5563*  5576   5614*  5625   5729*
                    5737   5816*  5824   5840*  5852   5895*  5903   5925*  6190*  6194   6200   6208   6216
                    6221*  6223*  6225   6231   6239   6248   6254*  6256*  6258   6263   6276*  6278   6289
                    6613   6619   6634   6922   6927   6960   7310*  7322   7327   7359   7364   7784*  8014*
                    8057*  8069*  8087*  8091*  8113*  8165*  8218*  8361*  8535*  8630*  8786*  8852*  8904*
                    8931*  8958*  9009*  9043   9047   9083*  9176   9183   9284*  9351*  9402*  9404*  9679*
                    9765   9861   9868   9928   9933   10030  10035  10117  10121  10143  10154* 10165  10497*
                   10553* 10594* 10649* 10656* 10691* 10846  10852  10958* 11013  11019  11043  11089* 11116
                   11121  11141* 11191* 11207  11211  11231* 11285* 11356* 11444* 11471  11476  11501* 11556*
                   11572  11576  11601* 11660* 11737* 11833* 11860  11865  11890* 11948* 11964  11968  11994*
                   12073* 12149* 12241* 12334* 12394  12402  12438* 12469  12474  12493* 12610* 12674* 12766*
                   12909* 12946  12951  12967* 13083* 13132  13137  13166* 13274* 13456* 13975* 14008*
RMER1I  001342      3296*  5404*  5412   5447*  5452*  5510*  5525*  5531   5576*  5589   5625*  5647   5737*
                    5740   5824*  5827*  5828   5852*  5857   5903*  5906   6194*  6195   6203   6211   6225*
                    6226   6234   6243
RMER1O  001416      3326*  8898*  8904   8919   8931   8946   8958   8973   8982   8991*
RMER2 = 000042      3040*  5396*  5406   5442*  5449   5490*  5503*  5516   5565*  5578   5616*  5627   5756*
                    5768   5788*  5800   5848*  5874*  5882   5901*  5921*  5933   6315   6320*  6324   6351*
```

		6353*	6355	6375*	6377*	6379	6406*	6408	6615	7301	7308*	7312	7343	7347
		7382*	7386	7395	7430	7436*	7438	7450*	7454	7470*	7474	7786*	8016*	8059*
		8071*	8089*	8115*	8117*	8167*	8220*	8363*	8537*	8632*	8775	8788*	8790	8808
		8842	8854*	8856	8874	9011*	9085*	9286*	9353*	9681*	10156*	10499*	10555*	10596*
		10651*	11091*	11143*	11193*	11233*	11287*	11358*	11446*	11503*	11558*	11603*	11662*	11739*
		11835*	11892*	11950*	11996*	12075*	12151*	12243*	12336*	12440*	12495*	12612*	12676*	12768*
		12911*	12969*	13085*	13168*	13276*	13458*	13977*	14010*					
RMER2I	001370	3307#	5406*	5409*	5414	5449*	5454*	5516*	5522*	5528*	5534	5578*	5583*	5590
		5627*	5651	5768*	5774*	5775	5800*	5806*	5807	5882*	5885*	5886	5933*	5939*
		5940	6324*	6327	6334	6341	6355*	6358	6365	6379*	6381	6388	6615*	6620
RMER20	001444	3337#												
RMHR =	000036	3038#	5735*	5792*	6024	6034	6048	6068	6632*	6686*	6689			
RMHRI	001364	3305#	6689*											
RMHRO	001440	3335#	6605*	6607*	6632	6645	6652*	6675*	6680*	6686	6699	17015		
RMLA =	000020	3032#	5790*											
RMLAI	001346	3298#												
RMLAO	001422	3328#												
RMMR1 =	000024	3033#	5444*	5451	5733*	7054	7060	7068*	7070*	7072	7081*	7083*	7085	7099*
		7101*	7103	7138*	7149*	7151*	7174*	7176*	7225*	7235*	7237*	7254*	7256*	7306*
		7339*	7341*	7380*	7384*	7434*	7448*	7452*	7468*	7472*	7520*	7522*	7533*	7549*
		7551*	7593	7599	7607*	7609*	7611*	7613	7621*	7623*	7625	7639*	7641*	7643
		7683	7700	7782*	7793*	7800*	7802*	7843*	7845*	7850*	7852*	7931*	7933*	7938*
		7940*	8012*	8020*	8022*	8055*	8085*	8111*	8161*	8163*	8214*	8216*	8246*	8248*
		8357*	8359*	8401*	8403*	8531*	8533*	8559*	8561*	8619	8626*	8628*	8635	8657*
		8659*	8661	8782*	8784*	8804*	8806*	8848*	8850*	8872*	8910*	8912*	8937*	8939*
		8964*	8966*	8970	8977	9005*	9007*	9030	9034	9079*	9081*	9103*	9105*	9163*
		9170*	9172*	9222*	9231*	9233*	9238	9248	9280*	9282*	9300*	9302*	9347*	9349*
		9362*	9364*	9398*	9400*	9415*	9417*	9470*	9476*	9478*	9529*	9535*	9537*	9636*
		9640*	9642*	9669	9675*	9677*	9692	9747*	9758*	9760*	9845*	9854*	9856*	9915*
		9921*	9923*	10017*	10023*	10025*	10088*	10105*	10107*	10112*	10114*	10150*	10152*	10192
		10443*	10449*	10451*	10495*	10501*	10551*	10557*	10592*	10601*	10615*	10647*	10689*	10760*
		10768*	10770*	10829*	10838*	10841*	10925	10929	10945*	10951*	10953*	10963	10971	11027*
		11029*	11037*	11039*	11087*	11098*	11100*	11105*	11109*	11111*	11139*	11150*	11152*	11157*
		11162*	11164*	11189*	11200*	11202*	11229*	11240*	11242*	11247*	11249*	11254*	11256*	11283*
		11294*	11296*	11301*	11306*	11308*	11322*	11327*	11329*	11354*	11381*	11383*	11438*	11453*
		11455*	11460*	11464*	11466*	11495*	11510*	11512*	11517*	11522*	11524*	11550*	11565*	11567*
		11595*	11610*	11612*	11617*	11619*	11624*	11626*	11654*	11669*	11671*	11676*	11681*	11683*
		11697*	11702*	11704*	11731*	11763*	11765*	11827*	11842*	11844*	11[illegible]*	11853*	11855*	11884*
		11899*	11901*	11906*	11911*	11913*	11942*	11957*	11959*	11988*	12003*	12005*	12012*	12015*
		12020*	12022*	12028*	12030*	12032*	12037*	12039*	12067*	12082*	12084*	12089*	12094*	12096*
		12110*	12115*	12117*	12143*	12158*	12160*	12167*	12170*	12175*	12177*	12184	12188*	12190*
		12196	12201*	12206*	12208*	12235*	12267*	12269*	12332*	12347*	12349*	12354*	12356*	12361*
		12363*	12371*	12373*	12375	12383	12389*	12436*	12451*	12453*	12458*	12463*	12465*	12491*
		12506*	12508*	12513*	12518*	12520*	12529*	12531*	12533	12541	12549*	12551*	12560*	12562*
		12568	12573	12581*	12583*	12608*	12623*	12625*	12630	12635	12643*	12645*	12672*	12688
		12696	12708*	12710*	12715*	12720*	12722*	12731*	12733*	12735	12743	12764*	12780	12788
		12797*	12799*	12804*	12806*	12821*	12823*	12907*	12923	12931	12940*	12942*	12965*	12981
		12989	12998*	13000*	13005*	13010*	13012*	13019*	13021*	13026*	13031*	13033*	13042*	13044*
		13046	13054	13081*	13093	13101	13110*	13112*	13118*	13120*	13125*	13127*	13164*	13176
		13184	13193*	13195*	13201*	13203*	13208*	13210*	13215*	13220*	13222*	13227*	13232*	13234*
		13243*	13245*	13247	13255	13272*	13288	13296	13305*	13307*	13313*	13318*	13320*	13328*
		13330*	13332	13340	13346*	13351*	13353*	13362*	13364*	13367	13375	13454*	13466	13474
		13492*	13494*	13506*	13508*	13539*	13541*	13554*	13556*	13566*	13568*	13971*	13973*	14006*
RMMR1I	001352	3300#	5451*	5456	7700*	7701	7719							
RMMR10	001425	3330#												
RMMR2 =	000040	3039#	5762*	8207	8224	8254	10303	10315	10319	10350	10362	10366	10397	10409

```
                    10413  10458  10471  11365  11371  11746  11753  12250  12257  12649  12655  12811  12838
                    13485  13499  13513  13547  13561  13573
RMMR2I  001366       3306#
RMMR20  001442       3336#
RMOF  = 000032       3036#  5345*  5356   5488*  5499*  5512   5550*  5559*  5572   5610*  5621   5754*  5766
                     5786*  5798   5842*  5854   5878*  5917*  5929   6445*  6447   6452   6475   6480*  6482*
                     6484   6503*  6505*  6507   6532*  6534*  6536   7696*  7790*  7839*  7914*  7927*  7993*
                     9293*  9315   9340   9355*  9369   9515*  9550   9555   9631*  9744*  9842*  9910*  10090*
                    13440*
RMOFI   001360       3303#  5356*  5366*  5512*  5520*  5526*  5532   5572*  5581*  5587   5621*  5635   5766*
                     5771*  5772   5798*  5803*  5804   5854*  5859*  5860   5929*  5934*  6484*  6486   6493
                     6507*  6509   6516
RMOFO   001434       3333#  7684*  7696   7704   7716   7722   7742   7744*  7767*  7790   7791   7839   7873
                     7875*  7908*  7914   7915   7927   7958   7960*  13406*  13417*  13440  13522
RMR   = 000004       2903#  6196   6227   10166  10171  10196  10990
RMSN  = 000030       3035#  5760*  5822*  6567   6571   6575
RMSNI   001356       3302#
RMSNO   001432       3332#
RMWC  = 000002       3112#  5298   10098*  13449*
RMWCI   001330       3291#
RMWCO   001404       3321#  13408*  13449
RQA   = 100000       2990#  10304  10316  10351  10363  10398  10410  10459  10463  11366  11747  12251  12650
                    12812  13486  13514  13562
RQB   = 040000       2991#  10304  10316  10351  10363  10398  10410  10459  10465  11366  11747  12251  12650
                    12812  13486  13514  13562
RTC   = 000016       2830#  8460   8716   9531   10796
SADMSK= 000037       2872#
SAVREG= 104414      13712  14869#
SA1   = 000001       2867#
SA16  = 000020       2863#
SA2   = 000002       2866#
SA4   = 000004       2865#
SA8   = 000010       2864#
SC    = 100000       3063#
SCTMSK= 003700       2923#
SCO   = 000100       2921#
SC1   = 000200       2920#
SC2   = 000400       2919#
SC3   = 001000       2918#
SC4   = 002000       2917#
SEARCH= 000030       2836#  8313   8592   8736   9782   9917   10019  11837  11894  11952  11998  12077  12153
                    12245  12342
SEEK  = 000004       2825#  8451   8696   9785   11448  11505  11560  11605  11664  11741
SETOM   060656       9517   9584   9625   13074  13157  13442  14004#
SETVV   060526       9156   9215   9461   9509   9523   9578   9619   9733   9830   9900   10005  10080  10293
                    10340  10387  10436  10753  10820  10934  11080  11131  11182  11222  11276  11347  11430
                    11486  11542  11587  11646  11723  11819  11875  11934  11980  12059  12135  12227  12325
                    12429  12484  12601  12665  12757  12900  12958  13063  13146  13265  13429  13967#
SIZCLK  060116       5183  13864#
SKI   = 040000       3011#  5454   6409   6411   7439   7455   7457   7475   7479
SNGPRT= 020024       2971#  5965   5969
STACK = 001100       2700#  4952   5246   5332   5385   5433   5474   5688   5960   5990   6017   6094   6184
                     6310   6439   6470   6561   6598   6670   6717   6759   6828   6911   7017   7050   7128
                     7214   7294   7423   7509   7588   7676   7762   7900   7979   8045   8150   8203   8347
                     8520   8615   8771   8838   8895   9068   9150   9207   9273   9336   9389   9453   9502
                     9572   9613   9663   9727   9824   9893   9999   10076  10139  10216  10250  10289  10336
```

```
                    10383   10431   10488   10545   10585   10640   10680   10746   10814   10916   11008   11072   11422
                    11811   12321   12423   12894   13393
STANDA  005234       5020    5031#
START   004542       3149    4940#  14569   14652
STKLMT= 177774       2711#
STOP    001530       3362#   8869    9698    9703   10968   10977   12399   12406   12693   12700   12785   12792   12928
                    12935   12986   12993   13098   13105   13181   13188   13293   13300   13471   13478   13873*  13886*
SWR     001154       3202#   4950    4972*   4974    4980*   4987*   5005    5194   13713   14333   14347   14350   14365
                    14372   14492   14499   14511   14515   14572   14611   14666*  14883   14896*
SWREG   000176       3131#   4980    5005    5194   14572   14611   14634
SW0   = 000001       2764#
SW00  = 000001       2754#   2764
SW01  = 000002       2753#   2763
SW02  = 000004       2752#   2762
SW03  = 000010       2751#   2761
SW04  = 000020       2750#   2760
SW05  = 000040       2749#   2759
SW06  = 000100       2748#   2758
SW07  = 000200       2747#   2757
SW08  = 000400       2746#   2756
SW09  = 001000       2745#   2755
SW1   = 000002       2763#
SW10  = 002000       2744#
SW11  = 004000       2743#
SW12  = 010000       2742#
SW13  = 020000       2741#  13713
SW14  = 040000       2740#
SW15  = 100000       2739#
SW2   = 000004       2762#
SW3   = 000010       2761#
SW4   = 000020       2760#
SW5   = 000040       2759#
SW6   = 000100       2758#
SW7   = 000200       2757#
SW8   = 000400       2756#
SW9   = 001000       2755#
TADMSK= 003400       2871#
TAG   = 020000       2992#  11395   11396   11397   11398   11399   11400   11784   11785   11786   11787   11788   11789
                    12291   12292   12293   12294   12295   12296   12650   12847   12848   12849   12850   12851   12852
                    13588   13589   13590   13591   13592   13593   13599   13600   13601   13602   13603   13604   13631
                    13632   13633   13634   13635
TAGADR= 001114       3174#   3181
TAP   = 040000       2967#
TA1   = 000400       2862#
TA2   = 001000       2861#
TA4   = 002000       2860#
TBITVE= 000014       2797#
TIME    001522       3359#  13916*  13933*  13935*
TKVEC = 000060       2804#  14549*  14550*
TPVEC = 000064       2805#
TRAPVE= 000034       2803#   4958*   4959*
TRE   = 040000       3064#
TRTVEC= 000014       2798#
TST   = 010000       2993#   8225    8227    8228    8255    8283    8291    8299    8307    8315    8323   11366   11747
                    12251   12812   13486   1351-   13562
TSTNMB  060050      13724*  13725*  13727   13844#
```

F13

```
TST43   025562          8141#  14426
TST44   025770          8194#  14427
TST45   026406          8338#  14428
TST46   027154          8511#  14429
TST47   027506          8606#  14430
TST5    007424          5466#  14396
TST50   030146          8762#  14431
TST51   030440          8829#  14432
TST52   030704          8886#  14433
TST53   031574          9059#  14434
TST54   032104          9141#  14435
TST55   032336          9198#  14436
TST56   032600          9264#  14437
TST57   033032          9327#  14438
TST6    010562          5679#  14397
TST60   033250          9380#  14439
TST61   033524          9444#  14440
TST62   033722          9493#  14441
TST63   034176          9563#  14442
TST64   034342          9604#  14443
TST65   034544          9655#  14444
TST66   035032          9718#  14445
TST67   035300          9815#  14446
TST7    011742          5951#  14398
TST70   035556          9884#  14447
TST71   036044          9990#  14448
TST72   036306         10067#  14449
TST73   036550         10130#  14450
TST74   037034         10207#  14451
TST75   037200         10241#  14452
TST76   037362         10280#  14453
TST77   037542         10327#  14454
TYPBN = 104406         13834   14861#
TYPDS = 104405         13674   13681   13828   14860#
TYPE  = 104401          4999    5040    5043    5049    5054    5057    5060    5064    5067    5073    5076    5082    5087
                        5092    5098    5108    5113    5119    5132    5133    5136    5141    5144    5151   13668   13675
                       13682   13717   13718   13726   13735   13741   13750   13788   13794   13798   13811   13817   13819
                       13837   13839   14086   14152   14221   14281   14494   14502   14566   14578   14632   14633   14636
                       14647   14657   14668   14687   14735   14741   14746   14750   14755   14756   14758   14761   14765
                       14856#  14905
TYPOC = 104402          5075   13831   14635   14857#
TYPON = 104404         14859#
TYPOS = 104403          5089    5110   13721   13729   13738   13744   14858#
T1      006270          5243    5244    5245#
T10     012144          5987    5988    5989#
T100    037766         10380   10381   10382#
T101    040150         10428   10429   10430#
T102    040370         10485   10486   10487#
T103    040630         10542   10543   10544#
T104    041014         10582   10583   10584#
T105    041256         10637   10638   10639#
T106    041444         10677   10678   10679#
T107    041676         10743   10744   10745#
T11     012266          6014    6015    6016#
T110    042124         10811   10812   10813#
T111    042432         10913   10914   10915#
```

T112	042774	11005	11006	11007#
T113	043246	11069	11070	11071#
T114	044670	11419	11420	11421#
T115	046446	11808	11809	11810#
T116	050632	12318	12319	12320#
T117	051242	12420	12421	12422#
T12	012564	6091	6092	6093#
T120	053414	12891	12892	12893#
T121	055732	13390	13391	13392#
T13	013176	6181	6182	6183#
T14	014024	6307	6308	6309#
T15	014560	6436	6437	6438#
T16	014716	6467	6468	6469#
T17	015272	6558	6559	6560#
T2	006644	5329	5330	5331#
T20	015432	6595	6596	6597#
T21	015754	6667	6668	6669#
T22	016130	6714	6715	6716#
T23	016314	6756	6757	6758#
T24	016624	6825	6826	6827#
T25	017162	6908	6909	6910#
T26	017644	7014	7015	7016#
T27	020004	7047	7048	7049#
T3	007064	5382	5383	5384#
T30	020330	7125	7126	7127#
T31	020702	7211	7212	7213#
T32	021230	7291	7292	7293#
T33	022004	7420	7421	7422#
T34	022354	7506	7507	7508#
T35	022704	7585	7586	7587#
T36	023264	7673	7674	7675#
T37	023652	7759	7760	7761#
T4	007300	5430	5431	5432#
T40	024372	7897	7898	7899#
T41	024722	7976	7977	7978#
T42	025164	8042	8043	8044#
T43	025624	8147	8148	8149#
T44	026032	8200	8201	8202#
T45	026450	8344	8345	8346#
T46	027216	8517	8518	8519#
T47	027550	8612	8613	8614#
T5	007466	5471	5472	5473#
T50	030210	8768	8769	8770#
T51	030502	8835	8836	8837#
T52	030746	8892	8893	8894#
T53	031636	9065	9066	9067#
T54	032146	9147	9148	9149#
T55	032400	9204	9205	9206#
T56	032642	9270	9271	9272#
T57	033074	9333	9334	9335#
T6	010624	5685	5686	5687#
T60	033312	9386	9387	9388#
T61	033566	9450	9451	9452#
T62	033764	9499	9500	9501#
T63	034240	9569	9570	9571#
T64	034404	9610	9611	9612#

```
T65     034606          9660    9661    9662#
T66     035074          9724    9725    9726#
T67     035342          9821    9822    9823#
T7      012004          5957    5958    5959#
T70     035620          9890    9891    9892#
T71     036106          9996    9997    9998#
T72     036350         10073   10074   10075#
T73     036612         10136   10137   10138#
T74     037076         10213   10214   10215#
T75     037242         10247   10248   10249#
T76     037424         10286   10287   10288#
T77     037604         10333   10334   10335#
UBUSQS  065177          5064   14984#
UNS   = 040000          2891#   5452    6282    6284    7317    7323    7328    7356    7360    7365    8991
UNTMSK= 000007          3095#   5259    5280    6604   10504
UPE   = 020000          3077#
USE   = 040000          3023#
U0    = 000001          3092#
U1    = 000002          3091#
U2    = 000004          3090#
VV    = 000100          2885#   7166    8917    8944    9024    9088    9110    9113   13982   13986
WATCH   001524          3360#   8865*   8867    9686*   9696   10959*  10966   12390*  12397   12684*  12691   12776*  12783
                       12919*  12926   12977*  12984   13089*  13096   13172*  13179   13284*  13291   13462*  13469   13936*
                       13938*
WC    = 000040          2957#
WCD   = 000050          2845#   8285    8960    9708    9800
WCE   = 040000          3076#  15080   15081
WCEHI = 010000          3101#
WCELO = 004000          3102#
WCF   = 000040          2900#   2907    6204    6235
WCH   = 000052          2846#   8293    9791
WD    = 000060          2849#   8301    9797   10100   10870   10874   10947
WH    = 000062          2850#   8309    9788
WLE   = 004000          2894#   2907   10847   10872   15084   15085
WRL   = 004000          2880#   7228    7240    7242    7259    7263
XNUDC = 176000          2985#   5364    5521    5561    5582    5642    5645    5742    5830    5862    5936    6738    6843
                        6855    6867    6885    9312    9432
XNUER2= 001567          3018#   5409    5522    5565    5583    5652    5655    5774    5806    5885    5939    6328    6382
                        6389    6390    6416    8122
XNUOF = 161577          2980#   5366    5520    5559    5581    5636    5639    5771    5803    5859    5934    6487    6494
                        6510    6517    6538    9316    9370
ZEROS   066176         15136#
$APTHD  001100          3161    3167#
$ASTAT= ****** U       14948   14963
$ATYC   064710         14919   14921#
$ATY1   064664         14917#
$ATY3   064672         14266   14918#
$ATY4   064702         14507   14920#
$AUTOB  001150          3199#   5009*   5198#  14625   14782
$BASE   001276          3268#   5074    5084*   5247    5314    5333    5386    5434    5475    5689    5961    5991    6018
                        6095    6185    6311    6440    6471    6562    6599    6671    6718    6760    6829    6912    7018
                        7051    7129    7215    7295    7424    7510    7589    7677    7763    7901    7980    8046    8151
                        8204    8348    8521    8616    8772    8839    8896    9069    9151    9208    9274    9337    9390
                        9454    9503    9573    9614    9664    9728    9825    9894   10000   10077   10140   10217   10251
                       10290   10337   10384   10432   10489   10546   10586   10641   10681   10747   10815   10917   11009
                       11073   11423   11812   12322   12424   12895   13394
```

$BDADR 001136
3194# 5261* 5262* 5282* 5283* 5314* 5971* 5972* 6000* 6023* 6024* 6111* 6112*
6126* 6127* 6142* 6143* 6164* 6165* 6199* 6200* 6207* 6208* 6215* 6216* 6230*
6231* 6238* 6239* 6247* 6248* 6262* 6263* 6288* 6289* 6314* 6315* 6451* 6452*
6474* 6475* 6566* 6567* 6618* 6619* 6775* 6776* 6790* 6791* 6808* 6809* 6835*
6836* 6926* 6927* 6954* 7022* 7053* 7054* 7133* 7134* 7218* 7219* 7300* 7301*
7326* 7327* 7346* 7347* 7363* 7364* 7394* 7395* 7429* 7430* 7515* 7516* 7592*
7593* 7682* 7683* 7765* 7766* 7906* 7907* 7982* 7983* 8050* 8051* 8153* 8206*
8207* 8376* 8392* 8393* 8407* 8432* 8433* 8524* 8618* 8619* 8774* 8775* 8841*
8842* 8922* 8923* 8949* 8950* 8976* 8977* 9013* 9014* 9033* 9034* 9046* 9047*
9071* 9072* 9182* 9183* 9247* 9248* 9339* 9340* 9425* 9426* 9434* 9435* 9456*
9457* 9505* 9506* 9554* 9555* 9575* 9576* 9616* 9617* 9668* 9669* 9867* 9868*
9932* 9933* 10034* 10035* 10120* 10121* 10142* 10143* 10232* 10261* 10270* 10271* 10318*
10319* 10365* 10366* 10412* 10413* 10470* 10471* 10514* 10515* 10527* 10528* 10566* 10567*
10606* 10607* 10620* 10621* 10662* 10663* 10705* 10706* 10778* 10779* 10851* 10852* 10928*
10929* 10970* 10971* 11012* 11013* 11120* 11121* 11171* 11210* 11211* 11265* 11266* 11317*
11336* 11370* 11371* 11475* 11476* 11531* 11575* 11576* 11635* 11636* 11692* 11711* 11752*
11753* 11864* 11865* 11920* 11967* 11968* 12048* 12049* 12105* 12124* 12195* 12196* 12216*
12256* 12257* 12382* 12383* 12401* 12402* 12473* 12474* 12540* 12541* 12572* 12573* 12591*
12634* 12635* 12654* 12655* 12695* 12696* 12742* 12743* 12787* 12788* 12837* 12838* 12930*
12931* 12950* 12951* 12988* 12989* 13053* 13054* 13101* 13136* 13137* 13183* 13184* 13254*
13255* 13295* 13296* 13339* 13340* 13374* 13375* 13473* 13474* 13498* 13499* 13546* 13547*
13572* 13573* 13984* 13985* 14018* 14019* 17007 17009 17011 17017 17019 17021

$BDDAT 001142
3196# 5255* 5256 5276* 5277 5537* 5594* 5665* 5964* 5965 5967 5996* 5997*
6034* 6035* 6048* 6049* 6051 6068* 6069* 6070 6107* 6108* 6122* 6123* 6137*
6138* 6140 6159* 6160* 6161 6195* 6196* 6203* 6204* 6211* 6212* 6226* 6227*
6234* 6235* 6243* 6244* 6258* 6260 6278* 6281* 6284* 6285 6327* 6328* 6334*
6335* 6341* 6342* 6358* 6359* 6365* 6366* 6381* 6382* 6388* 6389* 6391 6408*
6411* 6414* 6417 6447* 6448* 6486* 6487* 6493* 6494* 6509* 6510* 6516* 6517*
6519 6536* 6539 6575* 6576 6613* 6616* 6634* 6635* 6636 6691* 6692* 6771*
6772 6786* 6787 6804* 6805 6842* 6843* 6854* 6855* 6866* 6867* 6869 6887*
6888 6922* 6923* 6960* 6961* 6962 7029* 7030* 7060* 7061* 7072* 7073* 7085*
7086* 7103* 7104* 7140* 7141* 7153* 7154* 7165* 7166* 7178* 7179* 7184 7227*
7228* 7239* 7240* 7258* 7259* 7264 7312* 7313* 7322* 7323* 7324 7343* 7344*
7359* 7360* 7361 7386* 7387* 7392 7438* 7439* 7454* 7455* 7474* 7475* 7480
7524* 7525* 7535* 7536* 7553* 7554* 7559 7599* 7600* 7613* 7614* 7625* 7626*
7643* 7644* 7647 7701* 7702* 7713 7719* 7720* 7731 7804* 7805 7813 7817
7855* 7856 7863 7866 7942* 7943 7953 7999* 8000* 8024* 8025* 8061* 8062*
8073* 8074* 8093* 8094* 8119* 8120* 8125 8174* 8175 8178* 8224* 8225 8227*
8254* 8255* 8257 8371* 8372 8374* 8383 8388* 8389* 8390 8405* 8406* 8412
8418 8425 8430* 8431* 8434 8546* 8547 8549* 8563* 8564* 8567 8573 8635*
8636* 8661* 8662* 8667 8790* 8791* 8808* 8809* 8812 8856* 8857* 8874* 8875*
8916* 8917* 8943* 8944* 8970* 8971* 9016* 9017* 9023* 9024* 9030* 9031* 9043*
9044* 9087* 9088* 9109* 9110* 9176* 9177* 9180 9238* 9239* 9244 9369* 9370*
9422* 9423 9431* 9432* 9483* 9484* 9542* 9543* 9550* 9551* 9592* 9593* 9644*
9645* 9692* 9693* 9694 9861* 9862* 9865 9928* 9929* 10030* 10031* 10117* 10118*
10165* 10166* 10168 10225* 10229* 10257* 10258* 10266* 10267* 10315* 10316* 10362* 10363*
10409* 10410* 10458* 10459* 10463 10465 10511* 10512* 10522* 10524* 10562* 10563* 10603*
10604* 10617* 10618* 10658* 10659* 10699* 10700* 10702 10775* 10776* 10846* 10847* 10848
10925* 10926* 10963* 10964* 11019* 11020* 11043* 11044* 11116* 11117* 11168* 11169* 11207*
11208* 11261* 11262* 11313* 11314* 11333* 11334* 11365* 11366* 11367 11471* 11472* 11528*
11529* 11572* 11573* 11631* 11632* 11688* 11689* 11708* 11709* 11746* 11747* 11750 11850*
11861* 11917* 11918* 11964* 11965* 12044* 12045* 12101* 12102* 12121* 12122* 12184* 12185*
12212* 12213* 12250* 12251* 12254 12375* 12376* 12394* 12395* 12469* 12470* 12533* 12534*
12568* 12569* 12587* 12588* 12630* 12631* 12649* 12650* 12652 12688* 12689* 12735* 12736*
12780* 12781* 12811* 12812* 12818 12923* 12924* 12946* 12947* 12981* 12982* 13046* 13047*
13093* 13094* 13132* 13133* 13176* 13177* 13247* 13248* 13288* 13289* 13332* 13333* 13367*

```
                    13368* 13466* 13467* 13485* 13486* 13489  13513* 13514* 13536  13561* 13562* 13563  13981*
                    13982* 14014* 14015* 17007  17010  17011  17013  17015  17017  17019  17021
$BELL   001212       3219* 14494  14527  14578  14775
$BIN    061126      14082* 14085* 14086  14093*
$CDW1   001302       3270*
$CDW2   001304       3271*
$CHARC  062060      14283* 14293* 14300  14309* 14314*
$CKSWR  063336      14611* 14865
$CMTAG  001114       3182*  4947   4948   4956   4962   4963   4964
$CM3  = 000000       3212*
$CM4  = 000005       3212*  3213*  3214*  3215*  3216*  3217*
$CNTLC  064234      14566  14647  14775*
$CNTLG  064246      14632  14777*
$CNTLU  064241      14657  14750  14776*
$CPUOP  001250       3242*
$CRLF   001217       3221* 13682  13717  13750  13788  13798  13811  13819  13839  14282  14317  14502  14527
                    14668  14755  14775
$DBLK   061344      14118  14152  14160*
$DDW0   001306       3272*
$DDW1   001310       3273*
$DDW2   001312       3274*
$DDW3   001314       3275*
$DDW4   001316       3276*
$DDW5   001320       3277*
$DDW6   001322       3278*
$DDW7   001324       3279*
$DEVCT  001232       3233*
$DEVM   001300       3269*  5026*  5131*  5139*  5157*  5158   5165
$DOAGN  057304      13664  13685  13691*
$DTBL   061334      14121  14156*
$ENDAD  057274       3140   4997  13687*  14522
$ENDCT  057140       4962  13666*
$ENULL  057310      13694*
$ENV    001242       3238*  5003   5019   5192  14261  14504  14926  14950
$ENVM   001243       3239*  4985   5024  14263  14268  14928
$EOP    057104      13656*
$EOPCT  057132       4962* 13663* 13667
$EOSP   057050       5316   5974  13638*
$ERFLG  001117       3185* 14322  14361  14363  14369* 14391  14489* 14527
$ERMAX  001131       3191*  4965*  5242*  5328*  5381*  5429*  5470*  5684*  5956*  5986*  6013*  6090*  6180*
                     6306*  6435*  6466*  6557*  6594*  6666*  6713*  6755*  6824*  6907*  7013*  7046*  7124*
                     7210*  7290*  7419*  7505*  7584*  7672*  7758*  7896*  7975*  8041*  8146*  8199*  8343*
                     8516*  8611*  8767*  8834*  8891*  9064*  9146*  9203*  9269*  9332*  9385*  9449*  9498*
                     9568*  9609*  9659*  9723*  9820*  9889*  9995* 10072* 10135* 10212* 10246* 10285* 10332*
                    10379* 10427* 10484* 10541* 10581* 10636* 10676* 10742* 10810* 10912* 11004* 11068* 11418*
                    11807* 12317* 12419* 12890* 13389* 14363  14386* 14391
$ERROR  062640       4956  14487*
$ERRPC  001132       3192* 13742  14496* 14497* 14498  14527
$ERRTB  001532       3380* 13757
$ERTTL  001126       3189* 13679  13683* 14495* 14527
$ESCAP  001210       3218*  4964* 14385* 14518  14520  14527
$ETABL  001242       3237*
$ETEND  001326       3173   3280*
$FATAL  001224       3230* 14954*
$FFLG   065130      14917* 14920* 14948  14957* 14965*
$FILLC  001172       3210* 14286  14317
```

```
$FILLS  001171          3209#  14317
$GDADR  001134          3193#  10233*
$GDDAT  001140          3195#   5258*  5259*  5260*  5279*  5280*  5281*  5538*  5595*  5664*  5999*  6037*  6050*
                        6051    6072*  6110*  6125*  6139*  6140   6163*  6198*  6206*  6214*  6229*  6237*  6246*
                        6259*   6260   6287*  6316*  6390*  6391   6415*  6416*  6417   6450*  6476*  6518*  6519
                        6541*   6571*  6576   6602*  6636   6651*  6676*  6774*  6789*  6807*  6834*  6868*  6869
                        6890*   6914*  6925*  6962   6965   6986*  6992*  7032*  7063*  7075*  7088*  7108*  7143*
                        7156*   7168*  7137*  7220*  7242*  7266*  7302*  7317*  7324   7328   7337*  7350*  7354
                        7356*   7361   7365   7388*  7391*  7392   7441*  7457*  7476*  7479*  7480   7527*  7538*
                        7555*   7558*  7559   7594*  7628*  7649*  7703*  7708*  7712*  7713   7721*  7726*  7730*
                        7731    7769*  7805   7812*  7815*  7816*  7826*  7856   7862*  7865*  7877*  7910*  7943
                        7949*   7950   7952*  8002*  8027*  8064*  8066*  8121*  8124*  8125   8179*  8228*  8256*
                        8257    8375*  8382*  8385*  8390   8408*  8420*  8424*  8427*  843?   8542*  8575*  8638*
                        8663*   8666*  8667   8793*  8810*  8811*  8812   8859*  8877*  8921*  8948*  8975*  9012*
                        9090*   9113*  9178*  9179*  9180   9240*  9243*  9244   9341*  9392*  9458*  9545*  9553*
                        9595*   9647*  9666*  9694   9710*  9963*  9864*  9865   9931*  10033*  10122*  10167*  10168
                       10231*  10260*  10269*  10516*  10526*  10565*  10608*  10622*  10661*  10701*  10702  10780*  10850*
                       10930*  10969*  11022*  11046*  11119*  11172*  11212*  11264*  11316*  11337*  11369*  11474*  11532*
                       11577*  11634*  11691*  11712*  11748*  11749*  11750  11863*  11921*  11969*  12047*  12104*  12125*
                       12194*  12215*  12252*  12253*  12254  12381*  12400*  12472*  12539*  12571*  12590*  12633*  12651*
                       12652   12694*  12741*  12786*  12813*  12817*  12818  12929*  12949*  12987*  13052*  13099*  13100*
                       13135*  13182*  13253*  13294*  13338*  13373*  13472*  13487*  13488*  13489  13515*  13521*  13524*
                       13525*  13533*  13536  13563  13986*  14017*  17007  17010  17013  17015  17017  17019
$GET42  057264          13684#
$GTSWR  063426          14633#  14863
$HD     = 000001         2677    2678
$HIBTS  001100          3168#
$HIOCT  064376          14815*  14820#
$ICNT   001120          3186#   5241*  5327*  5380*  5428*  5469*  5683*  5955*  5985*  6012*  6089*  6179*  6305*
                        6434*   6465*  6556*  6593*  6665*  6712*  6754*  6823*  6906*  7012*  7045*  7123*  7209*
                        7289*   7418*  7504*  7583*  7671*  7757*  7895*  7974*  8040*  8145*  8198*  8342*  8515*
                        8610*   8766*  8833*  8890*  9063*  9145*  9202*  9268*  9331*  9384*  9448*  9497*  9567*
                        9608*   9658*  9722*  9819*  9888*  9994*  10071*  10134*  10211*  10245*  10284*  10331*  10378*
                       10426*  10483*  10540*  10580*  10635*  10675*  10741*  10809*  10911*  11003*  11067*  11417*  11806*
                       12316*  12418*  12889*  13388*  14376*  14377  14379*  14390
$ILLUP  064646          14875   14891  14908#
$INTAG  001151          3200#  14630*  14649  14669  14782
$ITEMB  001130          3190#  13733  14498*  14506  14527
$LF     001220          3222#  14317  14527  14765  14775
$LFLG   065127          14958*  14964#
$LLCSR  001512          3355#  13884  13914*  13946*
$LLVEC  001514          3356#  13876*  13877*  13887*  13888*  13899*  13900*
$LPADR  001122          3187#   4966*  5243*  5329*  5382*  5430*  5471*  5685*  5957*  5987*  6014*  6091*  6181*
                        6307*   6436*  6467*  6558*  6595*  6667*  6714*  6756*  6825*  6908*  7014*  7047*  7125*
                        7211*   7291*  7420*  7506*  7585*  7673*  7759*  7897*  7976*  8042*  8147*  8200*  8344*
                        8517*   8612*  8768*  8835*  8892*  9065*  9147*  9204*  9270*  9333*  9386*  9450*  9499*
                        9569*   9610*  9660*  9724*  9821*  9890*  9996*  10073*  10136*  10213*  10247*  10286*  10333*
                       10380*  10428*  10485*  10542*  10582*  10637*  10677*  10743*  10811*  10913*  11005*  11069*  11419*
                       11808*  12318*  12420*  12891*  13390*  14358*  14367*  14383*  14388  14390
$LPCSB  001504          3352#  13910*
$LPCSR  001502          3351#  13871  13911*  13943*
$LPERR  001124          3188#   4967*  5244*  5330*  5383*  5431*  5472*  5686*  5958*  5988*  6015*  6092*  6182*
                        6308*   6437*  6468*  6559*  6596*  6668*  6715*  6757*  6826*  6909*  7015*  7048*  7126*
                        7212*   7292*  7421*  7507*  7586*  7674*  7760*  7898*  7977*  8043*  8148*  8201*  8345*
                        8518*   8613*  8769*  8836*  8893*  9066*  9148*  9205*  9271*  9334*  9387*  9451*  9500*
                        9572*   9611*  9661*  9725*  9822*  9891*  9997*  10074*  10137*  10214*  10248*  10287*  10334*
```

```
                    10381*  10429*  10486*  10543*  10583*  10638*  10678*  10744*  10812*  10914*  11006*  11070*  11124*
                    11175*  11215*  11269*  11340*  11420*  11479*  11535*  11580*  11639*  11715*  11809*  11868*  11927*
                    11972*  12052*  12128*  12219*  12319*  12421*  12477*  12594*  12658*  12750*  12892*  12953*  13056*
                    13139*  13258*  13391*  14367   14384*  14390   14517
$LPVEC  001506       3353*  13874*  13875*  13889*  13890*  13897*  13898*
$MADR1  001254       3255*
$MADR2  001260       3259*
$MADR3  001264       3262*
$MADR4  001270       3265*
$MAIL   001222       3169    3173    3228*   4984    5003    5192    5238    5325    5377    5425    5467    5680    5952
                     5982    6009    6086    6176    6302    6431    6462    6553    6590    6662    6709    6752    6820
                     6904    7009    7042    7120    7206    7286    7415    7501    7580    7669    7754    7892    7971
                     8037    8142    8195    8339    8512    8607    8763    8830    8887    9060    9142    9199    9265
                     9328    9381    9445    9494    9564    9605    9656    9719    9816    9885    9991    10068   10131
                    10208   10242   10281   10328   10375   10423   10480   10537   10577   10632   10673   10738   10806
                    10896   11000   11064   11414   11803   12313   12415   12886   13385   14261   14382   14504
$MAMS1  001252       3249*
$MAMS2  001256       3257*
$MAMS3  001262       3260*
$MAMS4  001266       3263*
$MBADR  001102       3169*
$MFLG   065126      14918*  14924   14959*  14963*
$MNEW   064264      14636   14780*
$MSGAD  001236       3235*  14934*  14937
$MSGLG  001240       3236*  14939*
$MSGTY  001222       3229*  14932   14940*  14952   14956*
$MSWR   064253      14633   14778*
$MTYP1  001253       3250*
$MTYP2  001257       3258*
$MTYP3  001263       3261*
$MTYP4  001267       3264*
$MXCNT  062374      14380   14390*
$NULL   001170       3208*  14288   14317
$NWTST= 000001       5233*   5235    5320*   5322    5372*   5374    5420*   5422    5462*   5464    5675*   5677    5947*
                     5949    5977*   5979    6004*   6006    6081*   6083    6171*   6173    6297*   6299    6426*   6428
                     6457*   6459    6548*   6550    6585*   6587    6657*   6659    6704*   6706    6747*   6749    6815*
                     6817    6899*   6901    7004*   7006    7037*   7039    7115*   7117    7201*   7203    7281*   7283
                     7410*   7412    7496*   7498    7575*   7577    7664*   7666    7749*   7751    7887*   7889    7966*
                     7968    8032*   8034    8137*   8139    8190*   8192    8334*   8336    8507*   8509    8602*   8604
                     8758*   8760    8825*   8827    8882*   8884    9055*   9057    9137*   9139    9194*   9196    9260*
                     9262    9323*   9325    9376*   9378    9440*   9442    9489*   9491    9559*   9561    9600*   9602
                     9651*   9653    9714*   9716    9811*   9813    9880*   9882    9986*   9988   10063*  10065   10126*
                    10128   10203*  10205   10237*  10239   10276*  10278   10323*  10325   10370*  10372   10418*  10420
                    10475*  10477   10532*  10534   10572*  10574   10627*  10629   10668*  10670   10733*  10735   10801*
                    10803   10891*  10893   10995*  10997   11059*  11061   11409*  11411   11798*  11800   12308*  12310
                    12410*  12412   12881*  12883   13380*  13382
$OCNT   061576      14193*  14222*  14235*
$OMODE  061600      14188*  14192*  14197   14200*  14211*  14237*
$OVER   062360      14334   14359   14368   14378   14387*
$PASS   001230       3232*   4984*  13660*  13661*  13672   13694   14374   14391
$PASTM  001106       3171*
$POWER  064654      14906   14911*
$PSW    001520       3358*  13919*  13949
$PWRDN  064506       4960   14875*  14903
$PWRMG  064642      14906*
$PWRUP  064560      14885   14891*
```

DZRMJA.P11 01-AUG-77 11:10 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$QUES   001216          3220#  14317  14527  14687  14758  14775
$RDCHR  063700         14700#  14866
$RDDEC= ****** U       14869
$RDLIN  063770         14723#  14867
$RDOCT  064276         14793#  14868
$RCSZ = 000010         14716#
$RESRE  061016         14058#  14870
$RTNAD  057306         13693#
$R2A  = ****** U       14871
$SAVRE  060760         14042#  14869
$SAVR6  064652         14884#  14892  14893*  14894*  14910#
$SCOPE  062064          4954   14331#
$SETUP= 000137          3122#   4953   4954   4956   4958   4960   4962   4963   4964   4966   4997   5000   5189
                       13658  14332  14488  14514  14522  14570  14575  14576  14606  14782
$STUP = 177777          3122#
$SVLAD  062324         14342  14381#
$SVPC = 000200          3138#   3143
$SWR  = 167400          2666#   2677   2682   2683   2684   2685   2686   2687   2688   3217   3218   3219   4963
                        4964    4966   4967   5238   5325   5377   5425   5467   5680   5952   5982   6009   6086
                        6176    6302   6431   6462   6553   6590   6662   6709   6752   6820   6904   7009   7042
                        7120    7206   7286   7415   7501   7580   7669   7754   7892   7971   8037   8142   8195
                        8339    8512   8607   8763   8830   8887   9060   9142   9199   9265   9328   9381   9445
                        9494    9564   9605   9656   9719   9816   9885   9991  10068  10131  10208  10242  10281
                       10328   10375  10423  10480  10537  10577  10632  10673  10738  10806  10896  11000  11064
                       11414   11803  12313  12415  12886  13385  13653  13659  13686  13692  13694  14323  14324
                       14325   14326  14327  14333  14345  14347  14348  14361  14362  14363  14370  14371  14372
                       14384   14387  14390  14479  14480  14481  14482  14483  14492  14499  14511  14515  14527
                       14907
$SWREG  001244          3240#   4987
$SWRMK= 000000          2688    2689   14327  14328  14351
$SWOBT  062376         14357  14391#
$TESTN  001226          3231#   5238*  5325*  5377*  5425*  5467*  5680*  5952*  5982*  6009*  6086*  6176*  6302*
                        6431*   6462*  6553*  6590*  6662*  6709*  6752*  6820*  6904*  7009*  7042*  7120*  7206*
                        7286*   7415*  7501*  7580*  7669*  7754*  7892*  7971*  8037*  8142*  8195*  8339*  8512*
                        8607*   8763*  8830*  8887*  9060*  9142*  9199*  9265*  9328*  9381*  9445*  9494*  9564*
                        9605*   9656*  9719*  9816*  9885*  9991* 10068* 10131* 10208* 10242* 10281* 10328* 10375*
                       10423*  10480* 10537* 10577* 10632* 10673* 10738* 10806* 10896* 11000* 11064* 11414* 11803*
                       12313*  12415* 12886* 13385* 13725  14382*
$TIMES  001206          3217#   4963*  13659*  14370*  14377  14380*  14390
$TKB    001162          3205#  14530   14551   14562   14587  14615   14642
$TKCNT  063040         14531#  14546*  14576   14593*  14707  14709*
$TKINT  063050          5033    5202   14546#  14567   14628
$TKQEN= 063047         14535#  14601   14712
$TKQIN  063042         14532#  14547*  14548   14599*  14600* 14601   14603*
$TKQOU  063044         14533#  14548*  14710   14711*  14712  14714*
$TKQSR  063046         14534#  14547   14603   14714
$TKS    001160          3204#  14530   14552*  14583*  14585  14591*  14613  14629*  14639  14651*  14671*
$TKSRV  063120         14549  14562#
$TMP0   001174          3212#   5969*  6964*  7106*  7107*  7186*  7267*  7396*  7482*  7561*  7650*  7715*  7733*
                        7791*   7915*  8096*  8127*  8240*  8648*  8814*  9112*  9184*  9246*  9699*  9869*  9934*
                       10036*  10170* 10223* 10227  10456* 10461  10704* 10781* 10853* 10972* 11047* 13989* 17011
                       17013   17017  17019
$TMP1   001176          3213#   5042*  5043   5044   5047   5056*  5057   5058   5066*  5067   5068   5078*  5080
                        5084    5085*  5086*  5088   5094*  5096   5100   5101*  5102*  5103*  5104*  5105*  5106*
                        5107*   5109   5115*  5117   5122*  5123*  5124*  5125*  5126*  5127   5135*  5136   5137
                        5143*   5144   5145   5147   5149   5153   5970*  7716*  8097*  8128*  10854* 17011  17019
```

```
$TMP2   001200          3214#
$TMP3   001202          3215#
$TMP4   001204          3216#
$TN   = 000122          2677#   5233    5238#   5243    5320    5325#   5329    5372    5377#   5382    5420    5425#   5430
                        5462    5467#   5471    5675    5680#   5685    5947    5952#   5957    5977    5982#   5987    6004
                        6009#   6014    6081    6086#   6091    6171    6176#   6181    6297    6302#   6307    6426    6431#
                        6436    6457    6462#   6467    6548    6553#   6558    6585    6590#   6595    6657    6662#   6667
                        6704    6709#   6714    6747    6752#   6756    6815    6820#   6825    6899    6904#   6908    7004
                        7009#   7014    7037    7042#   7047    7115    7120#   7125    7201    7206#   7211    7281    7286#
                        7291    7410    7415#   7420    7496    7501#   7506    7575    7580#   7585    7664    7669#   7673
                        7749    7754#   7759    7887    7892#   7897    7966    7971#   7976    8032    8037#   8042    8137
                        8142#   8147    8190    8195#   8200    8334    8339#   8344    8507    8512#   8517    8602    8607#
                        8612    8758    8763#   8768    8825    8830#   8835    8882    8887#   8892    9055    9060#   9065
                        9137    9142#   9147    9194    9199#   9204    9260    9265#   9270    9323    9328#   9333    9376
                        9381#   9386    9440    9445#   9450    9489    9494#   9499    9559    9564#   9569    9600    9605#
                        9610    9651    9656#   9660    9714    9719#   9724    9811    9816#   9821    9880    9885#   9890
                        9986    9991#   9996    10063   10068#  10073   10126   10131#  10136   10203   10208#  10213   10237
                        10242#  10247   10276   10281#  10286   10323   10328#  10333   10370   10375#  10380   10418   10423#
                        10428   10475   10480#  10485   10532   10537#  10542   10572   10577#  10582   10627   10632#  10637
                        10668   10673#  10677   10733   10738#  10743   10801   10806#  10811   10891   10896#  10913   10995
                        11000#  11005   11059   11064#  11069   11409   11414#  11419   11798   11803#  11808   12308   12313#
                        12318   12410   12415#  12420   12881   12886#  12891   13380   13385#  13390   14352   14392
$TPB    001166          3207#  14306*  14317
$TPFLG  001173          3211#  14255   14317
$TPS    001164          3206#  14304   14317
$TRAP   064400          4958   14829#
$TRAP2  064440         14844#  14855
$TRP  = 000016         14848#  14857#  14858#  14859#  14860#  14861#  14862#  14863   14864#  14865   14866#  14867#  14868#
                       14869#  14870#  14871#
$TRPAD  064452         14838   14855#
$TSTM   001104          3170#
$TSTNM  001116          3184#  13658*  14322   14354*  14381*  14382   14387   14391   14491   14527
$TTYIN  064224         14725   14726   14738   14756   14770   14774#
$TYPBN  061054         14079#  14861
$TYPDS  061130         14106#  14860
$TYPE   061602         14255#  14848   14856   14945
$TYPEC  062014         14285   14292   14299   14304#  14305   14673
$TYPEX  062062         14310   14312   14315#
$TYPOC  061400         14191#  14857
$TYPON  061414         14190   14193#  14859
$TYPOS  061354         14186#  14858
$UNIT   001234          3234#   5203*  13719
$UNITM  001110          3172#
$USWR   001246          3241#
$VECT1  001272          3266#   5085    5100*   5101    5127*
$VECT2  001274          3267#
$XTSTR  062076         14336#
$$GET4= 000000         13686#
$$SW08= 000122         14391#  14392   14393#  14394#  14395#  14396#  14397#  14398#  14399#  14400#  14401#  14402#  14403#
                       14404#  14405#  14406#  14407#  14408#  14409#  14410#  14411#  14412#  14413#  14414#  14415#  14416#
                       14417#  14418#  14419#  14420#  14421#  14422#  14423#  14424#  14425#  14426#  14427#  14428#  14429#
                       14430#  14431#  14432#  14433#  14434#  14435#  14436#  14437#  14438#  14439#  14440#  14441#  14442#
                       14443#  14444#  14445#  14446#  14447#  14448#  14449#  14450#  14451#  14452#  14453#  14454#  14455#
                       14456#  14457#  14458#  14459#  14460#  14461#  14462#  14463#  14464#  14465#  14466#  14467#  14468#
                       14469#  14470#  14471#  14472#  14473#
$OFILL  061577         14187*  14191*  14201   14236#
```

DZRMJA.P11 01-AUG-77 11:10 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$40CAT= ****** U        14333   14501
.      = 113370          3125#   3129#   3138    3139#   3141#   3143#   3148#   3151#   3157    3158#   3160#   3162#   3174
                         3181#   3223    3349#   4951    4966    4967   13671#  13694   13695#  14160#  14317   14390   14391
                        14527   14530   14534#  14535   14536#  14774#  14775   14782#  14887   14909   14966#  15056#  17031#
                        17033#  17034#  17035#
.$ASTA= ****** U        14918   14921
.$X    = 001100          3157#   3162
```

CLEAR 2666# 5252 5273 5335 5388 5398 5436 5478 5543 5603 5726 5749 5781 5813 5837
5869 5892 5914 5993 6020 6098 6104 6191 6272 6321 6403 6442 6564 6610 6683
6720 6763 6832 6919 7020 7026 7057 7131 7221 7297 7426 7512 7596 7679 7779
7836 7903 7924 7990 8048 8082 8108 8158 8210 8353 8528 8623 8779 8845 9002
9076 9277 9344 9395 9672 10147 10220 10254 10492 10548 10589 10644 10685 10922 11016
13968

CLKOFF 2666# 8869 9698 9703 10968 10977 12399 12405 12693 12699 12785 12791 12928 12934 12986
12992 13098 13104 13181 13187 13293 13299 13471 13477

CLKON 2666# 8865 9686 10959 12390 12684 12776 12919 12977 13089 13172 13284 13462

COMMEN 2807#

ENDCOM 2807#

ERR 2666# 3385 3393 3401 3409 3417 3425 3434 3442 3450 3458 3466 3474 3482 3490
3498 3506 3514 3522 3530 3538 3546 3554 3562 3570 3578 3586 3594 3602 3610
3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730
3738 3746 3754 3762 3770 3779 3786 3794 3802 3810 3818 3826 3834 3842 3850
3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3962 3970
3978 3986 3994 4002 4010 4018 4026 4034 4042 4050 4058 4066 4074 4082 4090
4098 4106 4114 4122 4130 4138 4146 4154 4162 4170 4178 4186 4194 4202 4210
4218 4226 4234 4242 4250 4258 4266 4274 4282 4290 4298 4306 4314 4322 4330
4338 4346 4354 4362 4370 4378 4386 4394 4402 4410 4418 4426 4434 4442 4450
4458 4466 4474 4482 4490 4498 4506 4514 4522 4530 4538 4546 4554 4562 4570
4578 4586 4594 4602 4611 4619 4627 4635 4643 4651 4659 4667 4675 4684 4692
4700 4708 4716 4725 4733 4742 4750 4758 4766 4774 4782 4790 4799 4807 4815
4823 4831 4839 4847 4855 4863 4871 4879 4887 4895 4903 4912 4920 4928

ERROR 2701# 5185 5263 5284 5315 5369 5417 5458 5539 5596 5666 5745 5777 5809 5833
5865 5888 5910 5942 5973 6001 6038 6053 6073 6113 6128 6144 6166 6201 6209
6217 6232 6240 6249 6264 6290 6330 6337 6344 6361 6368 6384 6393 6419 6453
6489 6496 6512 6521 6542 6578 6622 6624 6640 6642 6694 6741 6777 6792 6810
6845 6857 6871 6891 6928 6955 6967 6969 7033 7064 7076 7089 7109 7144 7157
7169 7188 7230 7243 7268 7315 7330 7333 7348 7367 7370 7397 7442 7483 7528
7539 7562 7602 7616 7629 7651 7717 7734 7807 7858 7945 8003 8028 8065 8076
8098 8129 8180 8229 8259 8377 8394 8414 8421 8436 8550 8569 8576 8639 8669
8794 8815 8860 8878 8924 8951 8978 8985 9019 9026 9035 9048 9091 9114 9158
9185 9217 9249 9319 9372 9427 9436 9463 9486 9511 9519 9525 9546 9556 9580
9586 9596 9621 9627 9648 9700 9735 9775 9832 9870 9902 9935 10007 10037 10082
10123 10173 10175 10234 10262 10272 10295 10320 10342 10367 10389 10414 10438 10472 10517
10529 10568 10609 10623 10664 10707 10755 10782 10822 10855 10931 10936 10973 11023 11048
11082 11122 11133 11173 11184 11213 11224 11267 11278 11318 11338 11349 11372 11432 11477
11488 11533 11544 11578 11589 11637 11648 11693 11713 11725 11754 11821 11866 11877 11922
11936 11970 11982 12050 12061 12106 12126 12137 12197 12217 12229 12258 12327 12384 12403
12431 12475 12486 12542 12574 12592 12603 12636 12656 12667 12697 12744 12759 12789 12839
12902 12932 12952 12960 12990 13055 13065 13076 13102 13138 13148 13159 13185 13256 13267
13297 13341 13376 13431 13444 13475 13500 13548 13574

ESCAPE 2807#

GETAS 2666# 10521

GETBA 2666#

GETBAE 2666# 6982

GETCS1 2666# 5348 5401 5505 5567 5995 6106 6121 6136 6158 6690 7028 8173 8370 8404
8545 8562 10256 10307 11167 11312 11332 11527 11687 11707 11916 12100 12120 12211 12586

GETCS2 2666# 5254 5275

GETDA 2666# 5351 5507 5569 5618 5763 5795 5879 5904 6732 6770 6785 6803 7803 7854
9306 9421

GETDB 2666#

GETDC 2666# 5353 5513 5573 5622 5738 5825 5855 5930 6734 6841 6853 6865 6886 7941
9310 9430

GETDS 2666# 7139 7152 7164 7177 7226 7238 7257 7523 7534 7552 7998 8023 8060 8072

E14


```
        8092    8118    8386    8428    8915    8942    9015    9021    9086    9108    9482    9541    9591    9643   10224
       10265   10310   10357   10404   10510   10561   10602   10616   10657   10698   10774   11260   11630   12043   13980
       14013
GETDT   2666#   5963   10222   10455
GETEC1  2666#
GETEC2  2666#
GETER1  2666#   5403    5446    5509    5575    5624    5736    5823    5851    5902    6193    6224    6257    6277    6612
        6633    6921    6958    7321    7358    9042    9175    9764    9860    9927   10029   10116   10164   10845   11018
       11042   11115   11206   11470   11571   11859   11963   12392   12468   12945   13131
GETER2  2666#   5405    5448    5515    5577    5626    5767    5799    5881    5932    6323    6354    6378    6407    6614
        7311    7342    7385    7437    7453    7473    8789    8807    8855    9873
GETHR   2666#   6033    6047    6067    6688
GETLA   2666#
GETMR1  2666#   5450    7059    7071    7084    7102    7598    7612    7624    7642    7699    8634    8660    8969    9028
        9237    9690   10924   10961   12183   12374   12532   12567   12627   12686   12734   12778   12921   12979   13045
       13091   13174   13246   13286   13331   13366   13464
GETMR2  2666#   8223    8253   10302   10314   10349   10361   10396   10408   10457   11363   11744   12248   12648   12809
       13483   13511   13559
GETOF   2666#   5355    5511    5571    5620    5765    5797    5853    5928    6446    6483    6506    6535    9314    9368
        9549
GETPRI  2807#  13917
GETSN   2666#   6570    6574
GETSWR  2666#   2807#   5000#   5189
GETWC   2666#
GETX    2666#
MSG     2666#   5235    5322    5374    5422    5464    5677    5949    5979    6006    6083    6173    6299    6428    6459
        6550    6587    6659    6706    6749    6817    6901    7006    7039    7117    7203    7283    7412    7498    7577
        7666    7751    7889    7968    8034    8139    8192    8336    8509    8604    8760    8827    8884    9057    9139
        9196    9262    9325    9378    9442    9491    9561    9602    9653    9716    9813    9882    9988   10065   10128
       10205   10239   10278   10325   10372   10420   10477   10534   10574   10629   10670   10735   10803   10893   10997
       11061   11411   11800   12310   12412   12883   13382
MULT    2807#
NEWTST  2807#   5233    5320    5372    5420    5462    5675    5947    5977    6004    6081    6171    6297    6426    6457
        6548    6585    6657    6704    6747    6815    6899    7004    7037    7115    7201    7281    7410    7496    7575
        7664    7749    7887    7966    8032    8137    8190    8334    8507    8602    8758    8825    8882    9055    9137
        9194    9260    9323    9376    9440    9489    9559    9600    9651    9714    9811    9880    9986   10063   10126
       10203   10237   10276   10323   10370   10418   10475   10532   10572   10627   10668   10733   10801   10891   10995
       11059   11409   11798   12308   12410   12881   13380
POP     2807#   6944    6945    6949    6951    6987    6989    6999    7001   13903   13905   14063   14147   14816   14896
       14897   14960   14961
PUSH    2807#   6935    6936    6976    6977   13865   13866   14043   14106   14795   14877   14883   14921   14923   14944
PUTAS   2666#   5757   10354   10507   10597   10612   10652   10692
PUTBA   2666#  10095   13450
PUTBAE  2666#
PUTCS1  2666#   5338    5391    5481    5492    5554    5819    6102    6117    6119    6132    6134    6154    6156    6980
        7024    7794    7846    7934    8017    8171    8238    8368    8543    8646    8801    8862    8905    8932    8959
        9040    9097    9166    9225    9295    9357    9410    9471    9530    9637    9688    9751    9848    9916   10018
       10099   10158   10299   10346   10393   10444   10559   10763   10833   10946   11032   11092   11144   11194   11234
       11288   11359   11447   11504   11559   11604   11663   11740   11836   11893   11951   11997   12076   12152   12244
       12341   12445   12500   12617   12681   12773   12916   12974   13086   13169   13281   13459   13978   14011
PUTCS2  2666#   6629
PUTDA   2666#   5340    5483    5494    5547    5556    5607    5751    5783    5871    5896    5926    6029    6031    6043
        6045    6063    6065    6724    6728    6766    6768    6780    6783    6798    6801    7697    7787    7840    7928
        7994    9288    9405    9632    9739    9837    9911   10011   10091   10401   10940   11439   11496   11551   11596
       11655   11732   11828   11885   11943   11989   12068   12144   12236   12339   12441   12496   12613   12677   12769
       12912   12972   13069   13152   13277   13435
```

F14


```
PUTDB   2666#
PUTDC   2666#  5342  5500  5551  5560  5611  5730  5817  5843  5898  5918  6726  6730  6839  6849
        6851  6861  6863  6880  6882  7911  7996  9290  9407  9465  9589  9741  9839  9907 10013
       10093 10942 11441 11498 11553 11598 11657 11734 11830 11887 11945 11991 12070 12146 12238
       12337 12443 12498 12615 12679 12771 12914 12970 13071 13154 13279 13437
PUTDS   2666#  5875  5922
PUTDT   2666#  5845
PUTEC1  2666#  5849
PUTEC2  2666#  5793
PUTER1  2666#  5393  5439  5485  5496  5562  5613  5728  5815  5839  5894  5924  6189  6220  6222
        6253  6255  6275  7309  7783  8013  8056  8068  8086  8090  8112  8164  8217  8360  8534
        8629  8785  8851  8903  8930  8957  9008  9082  9283  9350  9401  9403  9678 10153 10496
       10552 10593 10648 10655 10690 10957 11088 11140 11190 11230 11284 11355 11443 11500 11555
       11600 11659 11736 11832 11889 11947 11993 12072 12148 12240 12333 12437 12492 12609 12673
       12765 12908 12966 13082 13165 13273 13455 13974 14007
PUTER2  2666#  5395  5441  5489  5502  5564  5615  5755  5787  5847  5873  5900  5920  6319  6350
        6352  6374  6376  6405  7307  7381  7435  7449  7469  7785  8015  8058  8070  8088  8114
        8116  8166  8219  8362  8536  8631  8787  8853  9010  9084  9285  9352  9680 10155 10498
       10554 10595 10650 11090 11142 11192 11232 11286 11357 11445 11502 11557 11602 11661 11738
       11834 11891 11949 11995 12074 12150 12242 12335 12439 12494 12611 12675 12767 12910 12968
       13084 13167 13275 13457 13976 14009
PUTHR   2666#  5734  5791  6631  6685
PUTLA   2666#  5789
PUTMR1  2666#  5443  5732  7067  7069  7080  7082  7097  7100  7137  7148  7150  7172  7175  7224
        7234  7236  7253  7255  7305  7338  7340  7378  7383  7433  7447  7451  7467  7471  7519
        7521  7532  7548  7550  7606  7608  7610  7620  7622  7638  7640  7781  7792  7799  7801
        7842  7844  7849  7851  7930  7932  7937  7939  8011  8019  8021  8054  8084  8110  8160
        8162  8213  8215  8245  8247  8356  8358  8400  8402  8530  8532  8558  8560  8625  8627
        8656  8658  8781  8783  8803  8805  8847  8849  8871  8906  8911  8935  8938  8962  8965
        9004  9006  9078  9080  9101  9104  9162  9168  9171  9221  9229  9232  9279  9281  9299
        9301  9346  9348  9360  9363  9397  9399  9413  9416  9469  9474  9477  9527  9533  9536
        9635  9639  9641  9674  9676  9746  9757  9759  9844  9852  9855  9914  9920  9922 10016
       10021 10024 10087 10103 10106 10111 10113 10149 10151 10442 10447 10450 10494 10500 10550
       10556 10591 10600 10614 10646 10688 10759 10767 10769 10828 10837 10840 10944 10949 10952
       11026 11028 11036 11038 11086 11096 11099 11104 11107 11110 11138 11149 11151 11156 11160
       11163 11188 11199 11201 11228 11238 11241 11246 11248 11253 11255 11282 11293 11295 11300
       11305 11307 11320 11325 11328 11353 11380 11382 11437 11451 11454 11459 11462 11465 11494
       11509 11511 11516 11520 11523 11549 11564 11566 11594 11608 11611 11616 11618 11623 11625
       11653 11668 11670 11675 11680 11682 11695 11700 11703 11730 11762 11764 11826 11840 11843
       11848 11851 11854 11883 11898 11900 11905 11909 11912 11941 11956 11958 11987 12001 12004
       12011 12014 12019 12021 12027 12029 12031 12035 12038 12066 12081 12083 12088 12093 12095
       12108 12113 12116 12142 12156 12159 12166 12169 12174 12176 12187 12189 12200 12204 12207
       12234 12266 12268 12331 12345 12348 12353 12355 12359 12362 12369 12372 12388 12435 12449
       12452 12457 12461 12464 12490 12504 12507 12512 12516 12519 12527 12530 12548 12550 12558
       12561 12579 12582 12607 12621 12624 12641 12644 12671 12706 12709 12714 12718 12721 12729
       12732 12763 12795 12798 12803 12805 12820 12822 12906 12938 12941 12964 12996 12999 13004
       13008 13011 13017 13020 13025 13029 13032 13040 13043 13080 13108 13111 13117 13119 13123
       13126 13162 13191 13194 13200 13202 13206 13209 13214 13218 13221 13226 13230 13233 13241
       13244 13271 13303 13306 13312 13316 13319 13326 13329 13345 13349 13352 13360 13363 13453
       13491 13493 13505 13507 13538 13540 13553 13555 13565 13567 13970 13972 14005
PUTMR2  2666#  5761
PUTOF   2666#  5344  5487  5498  5549  5558  5609  5753  5785  5841  5877  5916  6444  6479  6481
        6502  6504  6531  6533  7695  7789  7838  7913  7926  7992  9292  9354  9513  9630  9743
        9841  9909 10089 13439
PUTSN   2666#  5759  5821
PUTWC   2666# 10097 13448
```

```
PUTX      2666#
REPORT    2807#
RGBFMC    2666#  3283  3313
SCOPE     2702#  5237  5324  5376  5424  5466  5679  5951  5981  6008  6085  6175  6301  6430  6461
          6552   6589  6661  6708  6751  6819  6903  7008  7041  7119  7205  7285  7414  7500  7579
          7668   7753  7891  7970  8036  8141  8194  8338  8511  8606  8762  8829  8886  9059  9141
          9198   9264  9327  9380  9444  9493  9563  9604  9655  9718  9815  9884  9990  10067 10130
          10207  10241 10280 10327 10374 10422 10479 10536 10576 10631 10672 10737 10805 10895 10999
          11063  11413 11802 12312 12414 12885 13384
SETOM     2666#  9516  9583  9624  13073 13156 13441
SETPRI    2807#  4989  13923 13949 14703
SETTRA    14848# 14857 14858 14859 14860 14861 14863 14865 14866 14867 14868 14869 14870
SETUP     2807#  4946
SETVV     2666#  9155  9214  9460  9508  9522  9577  9618  9732  9829  9899  10004 10079 10292 10339
          10386  10435 10752 10819 10933 11079 11130 11181 11221 11275 11346 11429 11485 11541 11586
          11645  11722 11818 11874 11933 11979 12058 12134 12226 12324 12428 12482 12599 12663 12755
          12899  12957 13062 13144 13263 13427
SKIP      2807#
SLASH     2807#
SPACE     2807#
STARS     2807#  3136  3154  3156  3163  3177  3223  3226  5233  5236  5320  5323  5372  5375  5420
          5423   5462  5465  5675  5678  5947  5950  5977  5980  6004  6007  6041  6056  6081  6084
          6171   6174  6297  6300  6317  6348  6372  6397  6426  6429  6457  6460  6477  6500  6525
          6548   6551  6585  6588  6657  6660  6704  6707  6747  6750  6815  6818  6899  6902  6917
          6931   7004  7007  7037  7040  7115  7118  7201  7204  7281  7284  7410  7413  7496  7499
          7575   7578  7664  7667  7749  7752  7770  7822  7871  7887  7890  7966  7969  8032  8035
          8137   8140  8190  8193  8334  8337  8446  8507  8510  8602  8605  8758  8761  8825  8828
          8882   8885  9055  9058  9137  9140  9194  9197  9260  9263  9323  9326  9376  9379  9440
          9443   9489  9492  9559  9562  9600  9603  9651  9654  9714  9717  9811  9814  9880  9883
          9986   9989  10063 10066 10126 10129 10203 10206 10237 10240 10276 10279 10323 10326 10370
          10373  10418 10421 10475 10478 10532 10535 10572 10575 10627 10630 10668 10671 10733 10736
          10801  10804 10891 10894 10995 10998 11059 11062 11076 11126 11177 11217 11271 11342 11409
          11412  11426 11481 11537 11582 11641 11718 11798 11801 11815 11870 11929 11974 12054 12130
          12222  12308 12311 12410 12413 12426 12479 12596 12660 12752 12881 12884 12897 12954 13058
          13141  13260 13380 13383 13396 13399 13414 13424 13649 13698 13863 13908 14027 14072 14096
          14163  14240 14319 14475 14529 14606 14621 14692 14716 14785 14823 14873 14889 14916
SWRSU     2807#  4968#
TAGS      2666#  3282
TRMTRP    14848#
TSTSET    2666#  5240  5326  5379  5427  5468  5682  5954  5984  6011  6088  6178  6304  6433  6464
          6555   6592  6664  6711  6753  6822  6905  7011  7044  7122  7208  7288  7417  7503  7582
          7670   7756  7894  7973  8039  8144  8197  8341  8514  8609  8765  8832  8889  9062  9144
          9201   9267  9330  9383  9447  9496  9566  9647  9657  9721  9818  9887  9993  10070 10133
          10210  10244 10283 10330 10377 10425 10482 10539 10579 10634 10674 10740 10808 10910 11002
          11066  11416 11805 12315 12417 12888 13387
TYPBIN    2807#
TYPDEC    2807#  13672 13679
TYPNAM    2666#  2807#  4993
TYPNUM    2807#
TYPOCS    2807#  5088  5109  13719 13727 13736 13742
TYPOCT    2807#  5074  14634
TYPTXT    2807#  13668 13675
$$CMRE    3175#
$$CMTM    3175#  3212  3213  3214  3215  3216
$$ESCA    2807#
$$NEWT    2807#  5233  5320  5372  5420  5462  5675  5947  5977  6004  6081  6171  6297  6426  6457
```

```
          6548    6585    6657    6704    6747    6815    6899    7004    7037    7115    7201    7281    7410    7496    7575
          7664    7749    7887    7966    8032    8137    8190    8334    8507    8602    8758    8825    8882    9055    9137
          9194    9260    9323    9376    9440    9489    9559    9600    9651    9714    9811    9880    9986   10063   10126
         10203   10237   10276   10323   10370   10418   10475   10532   10572   10627   10668   10733   10801   10891   10995
         11059   11409   11798   12308   12410   12881   13380
SSSET    14848#  14857   14858   14859   14860   14861   14863   14865   14866   14867   14868   14869   14870
SSSETM    4984#
SSSKIP    2807#
.EQUAT    2666#   2697
.GETPR    2666#
.HEADE    2666#   2667
.SETUP    2666#   3122
.SWRHI    2666#   2678
.SWRLO    2666#   2689#
.$ACTI    2666#   3134
.$APTB    2666#   3224#
.$APTH    2666#   3152
.$APTY    2666#  14914
.$CATC    2666#   3123
.$CMTA    2666#   3175
.$DIV     2666#
.$EOP     2666#  13647
.$ERRO    2666#  14473
.$ERRT    2666#
.$MULT    2666#
.$POWE    2666#  14871
.$RAND    2666#
.$RDDE    2666#
.$RDDC    2666#  14783
.$READ    2666#  14527
.$SAVE    2666#  14025
.$SCOP    2666#  14317
.$SIZE    2666#
.$TRAP    2666#  14821
.$TYPB    2666#  14070
.$TYPD    2666#  14054
.$TYPE    2666#  14238
.$TYPO    2666#  14161


. ABS.   113370     000


 ERRORS DETECTED:  0

 DSKZ:DZRMJA.BIN,DSKZ:DZRMJA.SEQ/DOC/SOL/CRF=DSKM:DZRMJA.P11
 RUN-TIME: 49 47 5 SECONDS
 RUN-TIME RATIO: 576/102=5.6
 CORE USED:  33K  (65 PAGES)

 DOCUMENT PAGES:  386
```