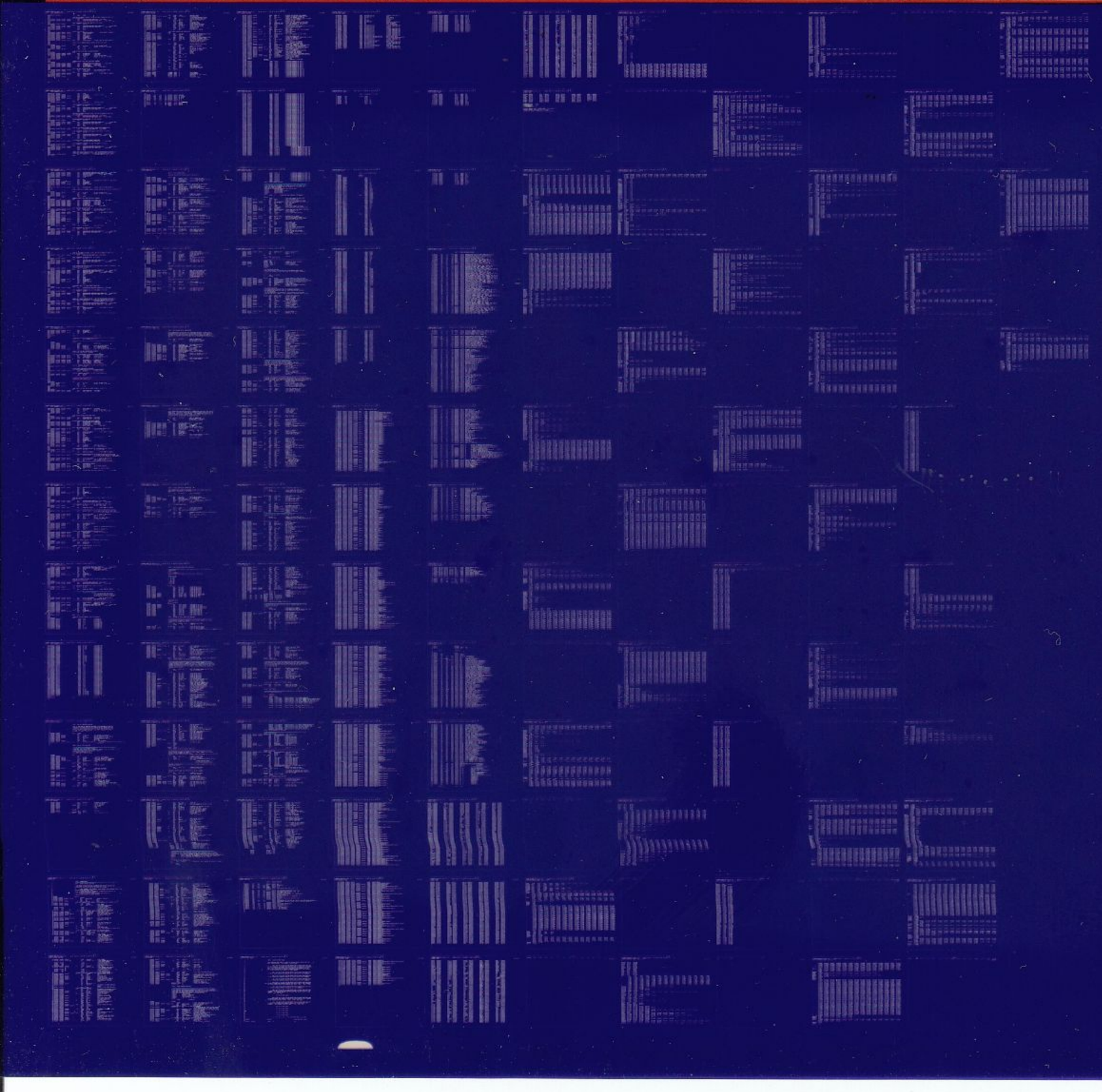
RM05/3/2

RM05/3/2 DSKLS TST 1 AH-F931A-MC
CZRMPA0
FICHE 1 OF 2

RM05/3/2

RM05/3/2 DSKLS TST 1 AH-F931A-MC
CZRMPA0
FICHE 2 OF 2



```
.REM    \
```

IDENTIFICATION

PRODUCT CODE: AC-F930A-MC

PRODUCT NAME: CZRMPA0 RM05/3/2 DSKLS TST 1

DATE CREATED: APRIL 1980

MAINTAINER: CX DIAGNOSTIC GROUP

AUTHOR: MIKE LEAVITT

CONTENTS

1.0 INTRODUCTION

1.1 ABSTRACT

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

TO DETECT ERRORS AND FAULTS IN THE RH MASSBUS CONTROLLER;

TO DETECT ERRORS AND FAULTS IN THE RM MASSBUS ADAPTER;

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

1.2 UNIT UNDER TEST

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

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

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

PDP-11 PROCESSOR
24K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 MEDIA REQUIREMENTS

NONE

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

NONE

3.0 OPERATING PROCEDURE

3.1 LOADING

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

3.2 SWITCH OPTIONS

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

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

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

3.3 STARTING PROGRAM

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

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

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

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

5.2 DUAL PORT CONFIGURATIONS

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

5.3 MEMORY PARITY HARDWARE

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

5.4 MEMORY MANAGEMENT HARDWARE

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

DISKLESS DIAGNOSTIC.

5.5 ACT11, APT11 COMPATIBILITY

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

5.6 XXDP COMPATIBILITY

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

5.7 OPERATING SYSTEM COMPATIBILITY

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

6.0 TEST DESCRIPTION

THE PROGRAM IS DESIGNED IN A BOTTOM UP MANNER SUCH THAT EACH TEST GENERALLY USES A MORE COMPLEX SUBSET OF HARDWARE THAN THE PREVIOUS TEST.

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

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

PROCEDURE:

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

PROBABLE FAULT:

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

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

3. CS MODULE

CTOD TEST

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

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.

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

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 OR DUAL PORT RM05, RM03 OR RM02 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. '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

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

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

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

PROCEDURE:

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

DATA PATTERN	PAT	PAR
000000	1	1
075266	0	0
163753	0	0
116535	1	1

PROBABLE FAULT:

1. IF MODULE

2. ASSYNCHRONOUS MASSBUS MODULE

CONTROL BUS PARITY GENERATION TEST

PURPOSE:

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

PROCEDURE:

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

SHIFTING ONE BIT PATTERN.

DATA PATTERN	MCPE
000000	0
056747	0
135672	0
163135	0

PROBABLE FAULT:

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

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 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.

PROCEDURE:

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

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

FINALLY, A SHIFTING 1 BIT PATTERN IS 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 RM05/3/2.

                             PROCEDURE:

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

                             PROBABLE FAULT:

                             1.  IF MODULE

                             2.  ASSYNCHRONOUS MASSBUS MODULE



                        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  RMMR1,  IS  NOT
                             STUCK AT ONE OR ZERO.

                             PROCEDURE:

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

STUCK AT ONE.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

MOL TEST

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

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 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

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 ''LST'' 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 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 TRANSFERRED, '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 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

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 RM05/3/2.

PROCEDURE:

WITH DIAGNOSTIC CLOCK ENABLED TO INHIBIT THE COMMAND SEQUENCER, THIS TEST VERIFIES THAT "ILF", BIT 0 OF RMER1, IS OFF FOR LEGAL FUNCTION CODES AND ON FOR ILLEGAL FUNCTIONCODES. THE STATUS OF THE "GO" BIT IS IGNORED.

PROBABLE FAULT:

1. IF MODULE

OCCUPIED TEST

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

READ IN PRESET TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" COMMAND IS DECODED, AND IN PARTICULAR, TO VERIFY THAT "IF5 READ IN CMD L" IS NOT STUCK AT ONE.

PROCEDURE:

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

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

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

. THE DISK ADDRESS REGISTER, RMDA, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-07, AND BITS 08-15 ARE ZERO AFTER THE RIP COMMAND.

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

PROBABLE FAULT:

1. IF MODULE =

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

```
1426
1427                        RMDA/RMDC/RIP TEST
1428
1429                             PURPOSE:
1430
1431                                  TO VERIFY THAT 'READ IN PRESET' RESETS THE DESIRED  CYLINDER
1432                             ADDRESS, RMDC, AND THE DISK ADDRESS, RMDA.
1433
1434                             PROCEDURE:
1435
1436                                  RMDA AND RMDC ARE PRESET THEN TESTED FOR ZERO AFTER THE  RIP
1437                             COMMAND.
1438
1439                             PROBABLE FAULT:
1440
1441                             1.  DS MODULE
1442
1443
1444
1445
1446
1447                        OFFSET COMMAND TEST
1448
1449                             PURPOSE:
1450
1451                                  TO  VERIFY  THAT  'OFFSET MODE'  SETS  WITH  OFFSET COMMAND.
1452
1453                             PROCEDURE:
1454
1455                                  THE TEST EXECUTES OFFSET COMMAND AND VERIFIES THAT 'OM', BIT
1456                             00 OF RMDS IS ONE.
1457
1458                             PROBABLE FAULT:
1459
1460                             1.  IF MODULE
1461
1462
1463
1464
1465
1466                        RETURN TO CENTER TEST
1467
1468                             PURPOSE:
1469
1470                                  TO VERIFY THAT 'RETURN TO CENTER' RESETS OFFSET MODE.
1471
1472                             PROCEDURE:
1473
1474                                  OFFSET MODE, BIT 00 OF RMDS, IS  SET  WITH  OFFSET  COMMAND,
1475                             THEN  THE  TEST  EXECUTES A RETURN TO CENTER COMMAND AND VERIFIES
1476                             THAT OFFSET MODE  RESETS.  OFFSET  DIRECTION  IS  ALSO  SET  AND
1477                             CHECKED FOR ZERO AFTER THE COMMAND.
1478
1479                             PROBABLE FAULT:
1480
1481                             1.  IF MODULE
1482
```

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

PROBABLE FAULT:

1. CS MODULE

2. SYNCHRONOUS MASSBUS MODULE

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 TRACK/SECTOR TEST

PURPOSE:

TO VERIFY THAT INVALID TRACK AND SECTOR ADDRESSES ARE DETECTED.

PROCEDURE:

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

PROBABLE FAULT:

1. DS MODULE

2. TRACK ADDRESS OPTION JUMPER

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

PURPOSE:

TO VERIFY THAT ADDRESS OVERFLOW ERROR IS DETECTED.

PROCEDURE:

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

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:

"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 DEVICE AVAILABLE STATUS AND DRIVE PRESENT STATUS ARE SET.

PROCEDURE:

DVA AND DPR ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

1. IF MODULE

PORT REQUEST TEST, PART 1

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE

PORT REQUEST TEST, PART 2

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE

PORT REQUEST TEST, PART 3

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

RELEASE TEST

PURPOSE:

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

PROCEDURE:

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

PROBABLE FAULT:

1. IF MODULE

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.

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

PURPOSE:

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

P SET ATA TEST

PURPOSE:

TO VERIFY THAT ATA IS SET AT THE COMPLETETION OF AN OFFSET AND RETURN TO CENTER LINE COMMAND.

PROCEDURE:

THE PROGRAM EXECUTES THE COMMANDS USING THE MAINTANCE DEBUG CLOCK AND EXPECTS ATA TO BE SET ON COMPLETETION.

PROBABLE FAULT:

1. IF MODULE

SET WLE TEST

PURPOSE:

TO VERIFY THAT 'WLE' IS SET OR RESET WHEN IT SHOULD BE.

PROCEDURE:

THE PROGRAM EXECUTES THE FOLLOWING COMMANDS USING THE MAINTANCE DEBUG CLOCK AND EXPECTS WLE SET OR RESET.

EXECUTE WRITE DATA COMMAND WITH MAINTANCE WRITE PROTECT SET, SHOULD EXPECT WLE TO BE SET.

EXECUTE WRITE DATA COMMAND WITHOUT MAINTANCE WRITE PROTECT SET, SHOULD EXPECT WLE TO BE RESET.

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

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

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

DATA COMMAND TESTS (1, 2, 3)

PURPOSE:

TO VERIFY THE COMMAND SEQUENCER DURING DATA COMMANDS.

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

\

SEQ 0038

```
                                ;PROGRAM VERSION #001

                                .TITLE  CZRMPA0 RM05/3/2 DSKLS TST 1
                                ;*COPYRIGHT (C) 1980
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY MIKE LEAVITT
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C4), 1980.
                                ;*
                                .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
104000                          ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
000004                          SCOPE   = IOT           ;;BASIC DEFINITION OF SCOPE CALL

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

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

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

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

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

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

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

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC  = 4             ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC  = 10            ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC = 14            ;;''T'' BIT
000014                  TRTVEC  = 14            ;;TRACE TRAP
000014                  BPTVEC  = 14            ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC  = 20            ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC  = 24            ;;POWER FAIL
000030                  EMTVEC  = 30            ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC = 34            ;;''TRAP'' TRAP
000060                  TKVEC   = 60            ;;TTY KEYBOARD VECTOR
000064                  TPVEC   = 64            ;;TTY PRINTER VECTOR
000240                  PIRQVEC = 240           ;;PROGRAM INTERRUPT REQUEST VECTOR
689
690                     .SBTTL  RM REGISTER BIT DEFINITIONS
691
692                     ;*RMCS1 CONTROL STATUS REGISTER
693
694     004000          DVA     = BIT11                 ;DEVICE AVAILABLE-READ ONLY
695     000040          F4      = BIT05                 ;FUNCTION CODE
696     000020          F3      = BIT04                 ;FUNCTION CODE
697     000010          F2      = BIT03                 ;FUNCTION CODE
698     000004          F1      = BIT02                 ;FUNCTION CODE
699     000002          F0      = BIT01                 ;FUNCTION CODE
700     000001          GO      = BIT00                 ;GO BIT
701     000077          FNCMSK  = 000077                ;FUNCTION CODE MASK
702
703                     ;FUNCTION CODES (BITS 01-05 OF RMCS1)
704     000000          NOP     = 000000                ;NOP COMMAND
705     000002          ILF02   = 000002                ;ILLEGAL COMMAND
706     000004          SEEK    = 000004                ;SEEK COMMAND
707     000006          RECAL   = 000006                ;RECALIBRATE COMMAND
708     000010          DRVCLR  = 000010                ;DRIVE CLEAR COMMAND
709     000012          RLEASE  = 000012                ;RELEASE COMMAND
710     000014          OFFSET  = 000014                ;OFFSET COMMAND
711     000016          RTC     = 000016                ;RETURN TO CENTERLINE COMMAND
712     000020          RIP     = 000020                ;READ IN PRESET COMMAND
713     000022          PAKACK  = 000022                ;PACK ACKNOWLEDGE COMMAND
714     000022          PACACK  = PAKACK
715     000024          ILF24   = 000024                ;ILLEGAL COMMAND
716     000026          ILF26   = 000026                ;ILLEGAL COMMAND
717     000030          SEARCH  = 000030                ;SEARCH COMMAND
```

```
                        ILF30   = 000030        ;ILLEGAL COMMAND
                        ILF32   = 000032        ;ILLEGAL COMMAND
                        ILF34   = 000034        ;ILLEGAL COMMAND
                        ILF36   = 000036        ;ILLEGAL COMMAND
                        ILF40   = 000040        ;ILLEGAL COMMAND
                        ILF42   = 000042        ;ILLEGAL COMMAND
                        ILF44   = 000044        ;ILLEGAL COMMAND
                        ILF46   = 000046        ;ILLEGAL COMMAND
                        WCD     = 000050        ;WRITE CHECK DATA COMMAND
                        WCH     = 000052        ;WRITE CHECK HEADER AND DATA
                        ILF54   = 000054        ;ILLEGAL COMMAND
                        ILF56   = 000056        ;ILLEGAL COMMAND
                        WD      = 000060        ;WRITE DATA COMMAND
                        WH      = 000062        ;WRITE HEADER AND DATA COMMAND
                        ILF64   = 000064        ;ILLEGAL COMMAND
                        ILF66   = 000066        ;ILLEGAL COMMAND
                        RD      = 000070        ;READ DATA COMMAND
                        RH      = 000072        ;READ HEADER AND DATA COMMAND
                        ILF74   = 000074        ;ILLEGAL COMMAND
                        ILF76   = 000076        ;ILLEGAL COMMAND

                        ;*RMDA  DISK ADDRESS REGISTER

                        ;TRACK ADDRESS DEFINITIONS
010000                  TA16    = BIT12         ;TRACK ADDRESS 16.
004000                  TA8     = BIT11         ;TRACK ADDRESS 8.
002000                  TA4     = BIT10         ;TRACK ADDRESS 4
001000                  TA2     = BIT09         ;TRACK ADDRESS 2
000400                  TA1     = BIT08         ;TRACK ADDRESS 1

                        ;SECTOR ADDRESS DEFINITIONS
000020                  SA16    = BIT04         ;SECTOR ADDRESS 16.
000010                  SA8     = BIT03         ;SECTOR ADDRESS 8.
000004                  SA4     = BIT02         ;SECTOR ADDRESS 4
000002                  SA2     = BIT01         ;SECTOR ADDRESS 2
000001                  SA1     = BIT00         ;SECTOR ADDRESS 1

                        ;TRACK & SECTOR MASKS
177400                  TADMSK  = 177400        ;TRACK ADDRESS MASK
000377                  SADMSK  = 000377        ;SECTOR ADDRESS MASK

                        ;*RMDS  DRIVE STATUS REGISTER

100000                  ATA     = BIT15         ;ATTENTION ACTIVE
040000                  ERR     = BIT14         ;COMPOSITE ERROR
020000                  PIP     = BIT13         ;POSITIONING IN PROGRESS
010000                  MOL     = BIT12         ;MEDIUM ON LINE
004000                  WRL     = BIT11         ;WRITE LOCK
002000                  LBT     = BIT10         ;LAST BLOCK TRANSFERRED
001000                  PGM     = BIT09         ;PROGRAMMABLE
000400                  DPR     = BIT08         ;DRIVE PRESENT
000200                  DRY     = BIT07         ;DRIVE READY
000100                  VV      = BIT06         ;VOLUME VALID
000001                  OM      = BIT00         ;OFFSET MODE ACTIVE

                        ;*RMER1 ERROR REGISTER #1
```

```
                        DCK     = BIT15                 ;DATA CHECK ERROR
                        UNS     = BIT14                 ;DRIVE UNSAFE
                        OPI     = BIT13                 ;OPERATION INCOMPLETE
                        DTE     = BIT12                 ;DRIVE TIMING ERROR
                        WLE     = BIT11                 ;WRITE LOCK ERROR
                        IAE     = BIT10                 ;INVALID ADDRESS ERROR
                        AOE     = BIT09                 ;ADDRESS OVERFLOW ERROR
                        HCRC    = BIT08                 ;HEADER CRC ERROR
                        HCE     = BIT07                 ;HEADER COMPARE ERROR
                        ECH     = BIT06                 ;ECC "HARD" ERROR
                        WCF     = BIT05                 ;WRITE CLOCK FAILURE
                        FER     = BIT04                 ;FORMAT ERROR
                        PAR     = BIT03                 ;PARITY ERROR
                        RMR     = BIT02                 ;REGISTER MODIFICATION REFUSED
                        ILR     = BIT01                 ;ILLEGAL REGISTER
                        ILF     = BIT00                 ;ILLEGAL FUNCTION

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

                        ;*RMAS  ATTENTION SUMMARY REGISTER

000377                  ATNMSK  = 377                   ;MASK FOR ATTENTION BITS

                        ;*RMLA  LOOK AHEAD REGISTER

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

003700                  SCTMSK  = 003700                ;SECTOR COUNT MASK

                        ;*RMMR1 MAINTENANCE REGISTER #1

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

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

```
827          020000                  EBL     = BIT13                 ;END OF BLOCK
828          010000                  REX     = BIT12                 ;EXCEPTION
829          004000                  ESRC    = BIT11                 ;ENABLE SEARCH
830          002000                  PLFS    = BIT10                 ;LOOKING FOR SYNC
831          001000                  ECRC    = BIT09                 ;ENABLE CRC OUT
832          000400                  PDA     = BIT08                 ;DATA AREA
833          000200                  PHA     = BIT07                 ;HEADER AREA
834          000100                  CONT    = BIT06                 ;CONTINUE
835          000040                  WC      = BIT05                 ;WORD CLOCK
836          000020                  EECC    = BIT04                 ;ENABLE ECC OUT
837          000010                  MWD     = BIT03                 ;WRITE DATA BIT
838          000004                  LS      = BIT02                 ;LAST SECTOR
839          000002                  LST     = BIT01                 ;LAST SECTOR AND TRACK
840          000001                  DMD     = BIT00                 ;DIAGNOSTIC MODE
841          051401                  MR1AAA  = DMD!MUR!DBEN!MOC!DTO
842
843                                  ;*RMDT  DRIVE TYPE REGISTER
844
845          100000                  NSA     = BIT15                 ;NOT SECTOR ADDRESSED = 0
846          040000                  TAP     = BIT14                 ;TAPE DRIVE = 0
847          020000                  MOH     = BIT13                 ;MOVING HEAD = 1
848          004000                  DRQ     = BIT11                 ;DRIVE REQUEST REQUIRED
849
850          020024                  SNGPRT  = 020024                ;SINGLE PORT DRIVE TYPE
851          024024                  DULPRT  = 024024                ;DUAL PORT DRIVE TYPE
852
853                                  ;*RMOF  OFFSET REGISTER
854
855          010000                  FMT16   = BIT12                 ;16 BIT WORD FORMAT
856          004000                  ECI     = BIT11                 ;ECC INHIBIT
857          002000                  HCI     = BIT10                 ;HEADER COMPARE INHIBIT
858          000200                  OFD     = BIT07                 ;OFFSET FORWARD
859          161577                  XNUOF   = 161577                ;UNSED BITS OF RMOF
860
861                                  ;*RMDC  DESIRED CYLINDER ADDRESS REGISTER
862
863          001777                  CYLMSK  = 001777                ;MASK FOR CYLINDER ADDRESS
864          176000                  XNUDC   = 176000                ;UNSED BITS OF RMDC
865
866                                  ;*RMMR2 MAINTENANCE REGISTER #2
867
868                                  ;READ ONLY BITS
869          100000                  RQA     = BIT15                 ;PORT A REQUEST
870          040000                  RQB     = BIT14                 ;PORT B REQUEST
871          020000                  TAG     = BIT13                 ;TAG CONTROL
872          010000                  TST     = BIT12                 ;COMMAND SEQUENCE TEST BIT
873          004000                  CC      = BIT11                 ;CONTROL OR CYLINDER TAG
874          002000                  CH      = BIT10                 ;CONTROL OR HEAD TAG
877          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
878
879                             ;*RMER2 ERROR REGISTER 2
880
881             100000          BSE     = BIT15                 ;BAD SECTOR ERROR
882             040000          SKI     = BIT14                 ;SEEK INCOMPLETE
883             020000          OPE     = BIT13                 ;OPERATOR PLUG ERROR
884             010000          IVC     = BIT12                 ;INVALID COMMAND ERROR
885             004000          LSC     = BIT11                 ;LOSS OF SYSTEM CLOCK
886             002000          LBC     = BIT10                 ;LOSS OF BIT CLOCK
887             000200          DVC     = BIT07                 ;DEVICE CHECK
888             000010          DPE     = BIT03                 ;DATA PARITY ERROR
889             001567          XNUER2  = 001567                ;UNSED BITS OF RMER2
890
891                             .SBTTL  PROGRAM MNEMONICS
892
893             100000          MSE     = BIT15                 ;MANUFACTURING DETECTED SECTOR ERROR
894             040000          USE     = BIT14                 ;USER DETECTED SECTOR ERROR
895
896                             .SBTTL  RM REGISTER INDEX VALUES
897
898             000000          RMCS1   = 00                    ;CONTROL STATUS REGISTER #1
899             000006          RMDA    = 06                    ;DISK ADDRESS REGISTER
900             000012          RMDS    = 12                    ;DRIVE STATUS REGISTER
901             000014          RMER1   = 14                    ;ERROR REGISTER #1
902             000016          RMAS    = 16                    ;ATTENTION SUMMARY REGISTER
903             000020          RMLA    = 20                    ;LOOK AHEAD REGISTER
904             000024          RMMR1   = 24                    ;MAINTENANCE REGISTER
905             000026          RMDT    = 26                    ;DRIVE TYPE REGISTER
906             000030          RMSN    = 30                    ;SERIAL NUMBER REGISTER
907             000032          RMOF    = 32                    ;OFFSET REGISTER
908             000034          RMDC    = 34                    ;DESIRED CYLINDER REGISTER
909             000036          RMHR    = 36                    ;HOLDING REGISTER
910             000040          RMMR2   = 40                    ;MAINTENANCE REGISTER #2
911             000042          RMER2   = 42                    ;ERROR REGISTER #2
912             000044          RMEC1   = 44                    ;ECC POSITION REGISTER
913             000046          RMEC2   = 46                    ;ECC PATTERN REGISTER
916             000050          ILRG50  = 50                    ;ILLEGAL REGISTER 50
                000052          ILRG52  = 52                    ;ILLEGAL REGISTER 52
                000054          ILRG54  = 54                    ;ILLEGAL REGISTER 54
                000056          ILRG56  = 56                    ;ILLEGAL REGISTER 56
                000060          ILRG60  = 60                    ;ILLEGAL REGISTER 60
                000062          ILRG62  = 62                    ;ILLEGAL REGISTER 62
                000064          ILRG64  = 64                    ;ILLEGAL REGISTER 64
                000066          ILRG66  = 66                    ;ILLEGAL REGISTER 66
                000070          ILRG70  = 70                    ;ILLEGAL REGISTER 70
                000072          ILRG72  = 72                    ;ILLEGAL REGISTER 72
                000074          ILRG74  = 74                    ;ILLEGAL REGISTER 74
                000076          ILRG76  = 76                    ;ILLEGAL REGISTER 76
917
918
919             000077          IDXMSK  = 77                    ;MASK FOR REGISTER INDEX NUMBER
920
921                             .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
922
923                             ;*RMCS1 CONTROL STATUS REGISTER #1
924
```

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

                        ;*RMCS2 RH CONTROL STATUS REGISTER #2

                        DLT     = BIT15                 ;DATA LATE-READ ONLY
                        WCE     = BIT14                 ;WRITE CHECK ERROR-READ ONLY
                        UPE     = BIT13                 ;UNIBUS PARITY ERROR
                        NED     = BIT12                 ;NONEXISTANT DRIVE-READ ONLY
                        NEM     = BIT11                 ;NONEXISTANT MEMORY-READ ONLY
                        PGE     = BIT10                 ;PROGRAM ERROR-READ ONLY
                        MXF     = BIT09                 ;MISSED TRANSFER
                        MDPE    = BIT08                 ;MASSBUS DATA BUS PARITY ERROR-READ ONLY
                        OR      = BIT07                 ;OUTPUT READY-READ ONLY
                        IR      = BIT06                 ;INPUT READY-READ ONLY
                        CLR     = BIT05                 ;CONTROLLER CLEAR
                        PAT     = BIT04                 ;PARITY TEST
                        BAI     = BIT03                 ;UNIBUS ADDRESS INCREMENT INHIBIT
                        U2      = BIT02                 ;UNIT SELECT
                        U1      = BIT01                 ;UNIT SELECT
                        U0      = BIT00                 ;UNIT SELECT

                        ;UNIT SELECT MASK

000007                  UNTMSK  = 7                     ;UNIT SELECT MASK

                        ;*RMCS3 RH70 CONTROL STATUS REGISTER #3

                        APE     = BIT15                 ;ADDRESS PARITY ERROR
                        DPEHI   = BIT14                 ;DATA PARITY ERROR HIGH WORD
                        DPELO   = BIT13                 ;DATA PARITY ERROR LOW WORD
                        WCEHI   = BIT12                 ;WRITE CHECK ERROR HIGH WORD
                        WCELO   = BIT11                 ;WRITE CHECK ERROR LOW WORD
                        DBL     = BIT10                 ;DOUBLE WORD TRANSFER
                        IE      = BIT06                 ;INTERRUPT ENABLE
                        IPCK3   = BIT03                 ;INVERT PARITY CHECK
                        IPCK2   = BIT02                 ;INVERT PARITY CHECK
                        IPCK1   = BIT01                 ;INVERT PARITY CHECK
                        IPCK0   = BIT00                 ;INVERT PARITY CHECK

                        .SBTTL  RH11/RH70 CONTROLLER REGISTER INDEX VALUES

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

176700                  ABASE   = 176700                ;UNIBUS ADDRESS
```

```
                        120254          AVECT1  = 120254                ;UNIBUS VECTOR ADDRESS AND PRIORITY
                                        .SBTTL  TRAP CATCHER

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

                                        .SBTTL  STARTING ADDRESS(ES)

000200  000137  004626                          JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM

                                        .SBTTL  ACT11 HOOKS

                                        ;;**************************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                        000204                  $SVPC=.                 ;SAVE PC
                        000046                  .=46
000046  054026                          $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052                  .=52
000052  000000                          '.WORD   0                      ;;2)SET LOC.52 TO ZERO
                        000204                  .=$SVPC                 ;; RESTORE PC

                        001100          .=1100
                                        .SBTTL  APT PARAMETER BLOCK

                                        ;;**************************************************************************
                                        ;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-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.

001100                                  $APTHD:
001100  000000                          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001222                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  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 ADDED UNIT
001112  000042                                  .WORD   $ETEND-$MAIL/2  ;;LENCTH MAILBOX-ETABLE(WORDS)
                        001114          TAGADR=.
```

```
      0                               .SBTTL  COMMON TAGS

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

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

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

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

```
      0                         .SBTTL  USER DEFINED TAGS

    001326  000000              XXDP:   .WORD   0       ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                                        ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
                                                        ;'XXDP' DEVICE CODE FOR THE RM05/3/2.
    001330      000             LSTRK:  .BYTE   0       ;LO BYTE = 0
    001331      000                     .BYTE   0       ;HI BYTE, CONTAINS LAST TRACK ADDRESS OF UNIT
                                                        ;UNDER TEST. RM02/3 = 4., RM05 = 18.

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

    001332                      GETBUF:

                                ;REGISTER INPUT BUFFER
    001332  000000              RMCS1I: .WORD   0       ;CONTROL, STATUS REGISTER #1
    001334  000000              RMWCI:  .WORD   0       ;WORD COUNT REGISTER
    001336  000000              RMBAI:  .WORD   0       ;BUS ADDRESS REGISTER
    001340  000000              RMDAI:  .WORD   0       ;DISK ADDRESS REGISTER
    001342  000000              RMCS2I: .WORD   0       ;CONTROL, STATUS REGISTER #2
    001344  000000              RMDSI:  .WORD   0       ;DRIVE STATUS REGISTER
    001346  000000              RMER1I: .WORD   0       ;ERROR REGISTER #1
    001350  000000              RMASI:  .WORD   0       ;ATTENTION SUMMARY REGISTER
    001352  000000              RMLAI:  .WORD   0       ;LOOK AHEAD REGISTER
    001354  000000              RMDBI:  .WORD   0       ;DATA BUFFER
    001356  000000              RMMR1I: .WORD   0       ;MAINTENANCE REGISTER #1
    001360  000000              RMDTI:  .WORD   0       ;DRIVE TYPE REGISTER
    001362  000000              RMSNI:  .WORD   0       ;SERIAL NUMBER REGISTER
    001364  000000              RMOFI:  .WORD   0       ;OFFSET REGISTER
    001366  000000              RMDCI:  .WORD   0       ;DESIRED CYLINDER REGISTER
    001370  000000              RMHRI:  .WORD   0       ;HOLDING REGISTER
    001372  000000              RMMR2I: .WORD   0       ;MAINTENANCE REGISTER #2
    001374  000000              RMER2I: .WORD   0       ;ERROR REGISTER #2
    001376  000000              RMEC1I: .WORD   0       ;ECC POSITION REGISTER
    001400  000000              RMEC2I: .WORD   0       ;ECC PATTERN REGISTER
    001402  000000              RMBAEI: .WORD   0       ;BUS ADDRESS EXTENSION REGISTER
    001404  000000              RMCS3I: .WORD   0       ;CONTROL, STATUS REGISTER #3

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

    001406                      PUTBUF:

                                ;REGISTER OUTPUT BUFFER
    001406  000000              RMCS1O: .WORD   0       ;CONTROL, STATUS REGISTER #1
    001410  000000              RMWCO:  .WORD   0       ;WORD COUNT REGISTER
    001412  000000              RMBAO:  .WORD   0       ;BUS ADDRESS REGISTER
    001414  000000              RMDAO:  .WORD   0       ;DISK ADDRESS REGISTER
    001416  000000              RMCS2O: .WORD   0       ;CONTROL, STATUS REGISTER #2
    001420  000000              RMDSO:  .WORD   0       ;DRIVE STATUS REGISTER
    001422  000000              RMER1O: .WORD   0       ;ERROR REGISTER #1
    001424  000000              RMASO:  .WORD   0       ;ATTENTION SUMMARY REGISTER
    001426  000000              RMLAO:  .WORD   0       ;LOOK AHEAD REGISTER
    001430  000000              RMDBO:  .WORD   0       ;DATA BUFFER
    001432  000000              RMMR1O: .WORD   0       ;MAINTENANCE REGISTER #1
    001434  000000              RMDTO:  .WORD   0       ;DRIVE TYPE REGISTER
    001436  000000              RMSNO:  .WORD   0       ;SERIAL NUMBER REGISTER
```

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

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

001462  000000                  TSTQUE: .WORD   0               ;CONTAINS DEVICE POINTER
                                        .BLKW   8.              ;TEST QUE FOR DEVICES UNDER TEST
001504  000000                          .WORD   0               ;TABLE TERMINATOR GOES HERE WHEN
                                                                ;ALL 8. DEVICES ARE UNDER TEST.

001506  172540                  $LPCSR: .WORD   172540          ;KW11-P CONTROL + STATUS REGISTER
001510  172542                  $LPCSB: .WORD   172542          ;KW11-P COUNT SET BUFFER
001512  000104                  $LPVEC: .WORD   104             ;KW11-P INTERRUPT VECTOR
001514  000106                          .WORD   106
001516  177546                  $LLCSR: .WORD   177546          ;KW11-L CONTROL + STATUS REGISTER
001520  000100                  $LLVEC: .WORD   100             ;KW11-L INTERRUPT VECTOR
001522  000102                          .WORD   102
001524  000000                  $PSW:   .WORD                   ;STORAGE FOR PRIORITY
001526  000000                  TIME:   .WORD                   ;STORAGE FOR ELAPSED TIME
001530  000000                  WATCH:  .WORD                   ;STORAGE FOR REMAINING TIME
001532  000000                  CLOCK:  .WORD                   ;ADDRESS OF START CLOCK SUB
001534  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


  001536                        $ERRTB:
    1
    2                           ;ERROR  1       CANNOT CLEAR NED STATUS
    3
  001536  063510                        EMT1
  001540  071366                        EHT1
  001542  071466                        EDT1
  001544  071514                        EFT1

    4
    5                           ;ERROR  2       CANNOT READ OR WRITE ANY DEVICE REG WITHOUT NED
    6
  001546  063516                        EMT2
  001550  071372                        EHT2
  001552  071470                        EDT2
  001554  071516                        EFT2

    7
    8                           ;ERROR  3       CANNOT WRITE/READ ONES TO ANY DEVICE REGISTER
    9
  001556  063544                        EMT3
  001560  000000                        0
  001562  000000                        0
  001564  000000                        0

   10
   11                           ;ERROR  4       CANNOT CLEAR ANY DEVICE REGISTER BITS W/MASSBUS INIT
   12
  001566  063564                        EMT4
  001570  000000                        0
  001572  000000                        0
  001574  000000                        0

   13
   14                           ;ERROR  5       CANNOT WRITE/READ ZEROS TO ALL BIT POSITIONS
   15
  001576  063606                        EMT5
  001600  071376                        EHT5
  001602  071472                        EDT5
  001604  071520                        EFT5

   16
   17                           ;ERROR  6       CANNOT WRITE/READ ONES TO ALL BIT POSITIONS
```

```
     18
     001606  063632                          EMT6
     001610  071376                          EHT5
     001612  071472                          EDT5
     001614  071520                          EFT5

     19
     20                             ;ERROR  7       CANNOT WRITE/READ SHIFTING ONE BIT TO ALL BIT POSITIONS
     21                             ;               OF DEVICE REGISTERS
     22
     001616  063654                          EMT7
     001620  071402                          EHT7
     001622  071472                          EDT5
     001624  071520                          EFT5

     23
     24                             ;ERROR  10      REGISTER SELECT 1 APPEARS S-A-0
     25
     001626  063676                          EMT10
     001630  000000                          0
     001632  000000                          0
     001634  000000                          0

     26
     27                             ;ERROR  11      REGISTER SELECT 1 APPEARS S-A-1
     28
     001636  063714                          EMT11
     001640  000000                          0
     001642  000000                          0
     001644  000000                          0

     29
     30                             ;ERROR  12      REGISTER SELECT 2 APPEARS S-A-0
     31
     001646  063732                          EMT12
     001650  000000                          0
     001652  000000                          0
     001654  000000                          0

     32
     33                             ;ERROR  13      REGISTER SELECT 2 APPEARS S-A-1
     34
     001656  063750                          EMT13
     001660  000000                          0
     001662  000000                          0
     001664  000000                          0

     35
     36                             ;ERROR  14      REGISTER SELECT 4 APPEARS S-A-0
     37
     001666  063766                          EMT14
     001670  000000                          0
     001672  000000                          0
     001674  000000                          0

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

```
 40
    001676  064004                          EMT15
    001700  000000                          0
    001702  000000                          0
    001704  000000                          0

 41
 42                                 ;ERROR  16      REGISTER SELECT 8 APPEARS S-A-0
 43
    001706  064022                          EMT16
    001710  000000                          0
    001712  000000                          0
    001714  000000                          0

 44
 45                                 ;ERROR  17      REGISTER SELECT 8 APPEARS S-A-1
 46
    001716  064040                          EMT17
    001720  000000                          0
    001722  000000                          0
    001724  000000                          0

 47
 48                                 ;ERROR  20      CANT WRITE ZEROS RMDA
 49
    001726  064056                          EMT20
    001730  071366                          EHT1
    001732  071466                          EDT1
    001734  071514                          EFT1

 50
 51                                 ;ERROR  21      CANT WRITE ONES RMDA
 52
    001736  064076                          EMT21
    001740  071366                          EHT1
    001742  071466                          EDT1
    001744  071514                          EFT1

 53
 54                                 ;ERROR  22      BIT INTERFERENCE IN WRITING/READING RMDA
 55
    001746  064116                          EMT22
    001750  071366                          EHT1
    001752  071466                          EDT1
    001754  071514                          EFT1

 56
 57                                 ;ERROR  23      CANT WRITE ZEROS RMCS1
 58
    001756  064132                          EMT23
    001760  071366                          EHT1
    001762  071466                          EDT1
    001764  071514                          EFT1

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

```
001766  064152                          EMT24
001770  071366                          EHT1
001772  071466                          EDT1
001774  071514                          EFT1

62
63                              ;ERROR  25      BIT INTERFERENCE IN WRITING/READING RMCS1
64
001776  064172                          EMT25
002000  071366                          EHT1
002002  071466                          EDT1
002004  071514                          EFT1

65
66                              ;ERROR  26      MBA CLR L IS STUCK ACTIVE
67
002006  064206                          EMT26
002010  000000                          0
002012  000000                          0
002014  000000                          0

68
69                              ;ERROR  27      CANNOT CLEAR RMER1-PAR,RMR,ILF,ILR
70
002016  064240                          EMT27
002020  071366                          EHT1
002022  071466                          EDT1
002024  071514                          EFT1

71
72                              ;ERROR  30      CANNOT CLEAR RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
73
002026  064252                          EMT30
002030  071366                          EHT1
002032  071466                          EDT1
002034  071514                          EFT1

74
75                              ;ERROR  31      CANNOT CLEAR RMER1-OPI,DTE
76
002036  064266                          EMT31
002040  071366                          EHT1
002042  071466                          EDT1
002044  071514                          EFT1

77
78                              ;ERROR  32      CANNOT WRITE 0 IN RMER1-PAR,RMR,ILF,ILR
79
002046  064302                          EMT32
002050  071366                          EHT1
002052  071466                          EDT1
002054  071514                          EFT1

80
81                              ;ERROR  33      CANNOT WRITE 0 IN RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
82
002056  064316                          EMT33
```

```
002060  071366                          EHT1
002062  071466                          EDT1
002064  071514                          EFT1

 83
 84                             ;ERROR  34      CANNOT WRITE 0 IN RMER1-OPI,DTE
 85
002066  064334                          EMT34
002070  071366                          EHT1
002072  071466                          EDT1
002074  071514                          EFT1

 86
 87                             ;ERROR  35      CANNOT WRITE 1 IN RMER1
 88
002076  064352                          EMT35
002100  071366                          EHT1
002102  071466                          EDT1
002104  071514                          EFT1

 89
 90                             ;ERROR  36      CANNOT WRITE SHIFTING 1 IN RMER1
 91
002106  064366                          EMT36
002110  071366                          EHT1
002112  071466                          EDT1
002114  071514                          EFT1

 92
 93                             ;ERROR  37      CANNOT WRITE ZEROS IN RMDC
 94
002116  064402                          EMT37
002120  071366                          EHT1
002122  071466                          EDT1
002124  071514                          FFT1

 95
 96                             ;ERROR  40      CANNOT WRITE ONES IN RMDC
 97
002126  064416                          EMT40
002130  071366                          EHT1
002132  071466                          EDT1
002134  071514                          EFT1

 98
 99                             ;ERROR  41      BIT INTERFERENCE IN WRITING/READING RMDC
100
002136  064436                          EMT41
002140  071366                          EHT1
002142  071466                          EDT1
002144  071514                          EFT1

101
102                            ;ERROR  42      CANNOT WRITE 1'S IN RMDC OR RMDA
103
002146  064452                          EMT42
002150  000000                          0
```

```
002152  000000                          0
002154  000000                          0

                        ;ERROR  43      CANNOT CLEAR RMCS1-FUNCTION CODE

002156  064476                          EMT43
002160  071366                          EHT1
002162  071466                          EDT1
002164  071514                          EFT1

                        ;ERROR  44      UNUSED BITS OF RMER2 NOT ZERO

002166  064510                          EMT44
002170  071366                          EHT1
002172  071466                          EDT1
002174  071514                          EFT1

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

002176  064524                          EMT45
002200  071366                          EHT1
002202  071466                          EDT1
002204  071514                          EFT1

                        ;ERROR  46      CANNOT CLEAR RMER2-LBC,DPE

002206  064540                          EMT46
002210  071366                          EHT1
002212  071466                          EDT1
002214  071514                          EFT1

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

002216  064554                          EMT47
002220  071366                          EHT1
002222  071466                          EDT1
002224  071514                          EFT1

                        ;ERROR  50      CANNOT WRITE ZEROS RMER2-LBC,DPE

002226  064576                          EMT50
002230  071366                          EHT1
002232  071466                          EDT1
002234  071514                          EFT1

                        ;ERROR  51      CANNOT WRITE ONES RMER2

002236  064620                          EMT51
002240  071366                          EHT1
002242  071466                          EDT1
```

```
        002244  071514                          EFT1

    125
    126                             ;ERROR  52      CANNOT WRITE SHIFTING ONES RMER2
    127
        002246  064640                          EMT52
        002250  071366                          EHT1
        002252  071466                          EDT1
        002254  071514                          EFT1

    128
    129                             ;ERROR  53      UNUSED BITS OF RMOF ARE NOT ZERO
    130
        002256  064654                          EMT53
        002260  071366                          EHT1
        002262  071466                          EDT1
        002264  071514                          EFT1

    131
    132                             ;ERROR  54      CANNOT WRITE ZEROS RMOF-FMT,ECI,HCI,OFD
    133
        002266  064670                          EMT54
        002270  071366                          EHT1
        002272  071466                          EDT1
        002274  071514                          EFT1

    134
    135                             ;ERROR  55      CANNOT WRITE ONES RMOF-FMT,ECI,HCI,OFD
    136
        002276  064710                          EMT55
        002300  071366                          EHT1
        002302  071466                          EDT1
        002304  071514                          EFT1

    137
    138                             ;ERROR  56      CANNOT WRITE SHIFTING ONES RMOF
    139
        002306  064730                          EMT56
        002310  071366                          EHT1
        002312  071466                          EDT1
        002314  071514                          EFT1

    140
    141                             ;ERROR  57      DEVICE IS NOT AN RM05/3/2
    142
        002316  064744                          EMT57
        002320  071406                          EHT57
        002322  071474                          EDT57
        002324  071522                          EFT57

    143
    144                             ;ERROR  60      DEVICE AVAILABLE IS NOT SET
    145
        002326  064760                          EMT60
        002330  071366                          EHT1
        002332  071466                          EDT1
        002334  071514                          EFT1
```

```
                                ;ERROR  61      CANNOT WRITE ZEROS RMHR

002336  064774                          EMT61
002340  071366                          EHT1
002342  071466                          EDT1
002344  071514                          EFT1

                                ;ERROR  62      CANNOT WRITE ONES RMHR

002346  065014                          EMT62
002350  071366                          EHT1
002352  071466                          EDT1
002354  071514                          EFT1

                                ;ERROR  63      CANNOT WRITE SHIFTING ONES RMHR

002356  065034                          EMT63
002360  071366                          EHT1
002362  071466                          EDT1
002364  071514                          EFT1

                                ;ERROR  64      CANNOT CLEAR ILR STATUS

002366  065050                          EMT64
002370  071366                          EHT1
002372  071466                          EDT1
002374  071514                          EFT1

                                ;ERROR  65      ILR ERROR SHOULD NOT BE SET

002376  065062                          EMT65
002400  071412                          EHT65
002402  071476                          EDT65
002404  071524                          EFT65

                                ;ERROR  66      ILR ERROR SHOULD BE SET

002406  065076                          EMT66
002410  071412                          EHT65
002412  071476                          EDT65
002414  071524                          EFT65

                                ;ERROR  67      CANNOT CLEAR PAR STATUS-DPE IS RESET

002416  065112                          EMT67
002420  071366                          EHT1
002422  071466                          EDT1
002424  071514                          EFT1
```

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

002426  065130                          EMT70
002430  071366                          EHT1
002432  071466                          EDT1
002434  071514                          EFT1

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

002436  065150                          EMT71
002440  071416                          EHT71
002442  071500                          EDT71
002444  071526                          EFT71

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

002446  065174                          EMT72
002450  071416                          EHT71
002452  071500                          EDT71
002454  071526                          EFT71

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

002456  065220                          EMT73
002460  071416                          EHT71
002462  071500                          EDT71
002464  071526                          EFT71

                                ;ERROR  74      UNEXPECTED BUS TIMEOUT

002466  065240                          EMT74
002470  071422                          EHT74
002472  071502                          EDT74
002474  071530                          EFT74

                                ;ERROR  75      CANT CLEAR 'DMD''

002476  065250                          EMT75
002500  071366                          EHT1
002502  071466                          EDT1
002504  071514                          EFT1

                                ;ERROR  76      CANT WRITE ZERO 'DMD''

002506  065262                          EMT76
002510  071366                          EHT1
002512  071466                          EDT1
002514  071514                          EFT1
```

```
                                        ;ERROR  77      CANT WRITE ONE 'DMD''

002516  065276                                  EMT77
002520  071366                                  EHT1
002522  071466                                  EDT1
002524  071514                                  EFT1

                                        ;ERROR  100     DMD SET BY WRONG BIT

002526  065312                                  EMT100
002530  071416                                  EHT71
002532  071476                                  EDT65
002534  071524                                  EFT65

                                        ;ERROR  101     CANT CLEAR 'MOL'' IN DIAGNOSTIC MODE

002536  065332                                  EMT101
002540  071366                                  EHT1
002542  071466                                  EDT1
002544  071514                                  EFT1

                                        ;ERROR  102     CANT SET 'MOL'' IN DIAGNOSTIC MODE

002546  065352                                  EMT102
002550  071366                                  EHT1
002552  071466                                  EDT1
002554  071514                                  EFT1

                                        ;ERROR  103     'MUR'' SET BY WRONG BIT

002556  065372                                  EMT103
002560  071416                                  EHT71
002562  071476                                  EDT65
002564  071524                                  EFT65

                                        ;ERROR  104     CANT RESET 'WRL'' IN DIAGNOSTIC MODE

002566  065416                                  EMT104
002570  071366                                  EHT1
002572  071466                                  EDT1
002574  071514                                  EFT1

                                        ;ERROR  105     CANT SET 'WRL'' IN DIAGNOSTIC MODE

002576  065436                                  EMT105
002600  071366                                  EHT1
002602  071466                                  EDT1
002604  071514                                  EFT1

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

```
211
     002606  065456                          EMT106
     002610  071416                          EHT71
     002612  071476                          EDT65
     002614  071524                          EFT65

212
213                                 ;ERROR  107     CANT RESET 'DVC' USING 'MDVC'
214
     002616  065502                          EMT107
     002620  071366                          EHT1
     002622  071466                          EDT1
     002624  071514                          EFT1

215
216                                 ;ERROR  110     'DVC' IS RESET BUT 'UNS' IS SET
217
     002626  065522                          EMT110
     002630  071366                          EHT1
     002632  071466                          EDT1
     002634  071514                          EFT1

218
219                                 ;ERROR  111     'DVC' IS SET BUT 'UNS' IS NOT SET
220
     002636  065544                          EMT111
     002640  071366                          EHT1
     002642  071466                          EDT1
     002644  071514                          EFT1

221
222                                 ;ERROR  112     CANT SET 'DVC' USING MDVC'
223
     002646  065564                          EMT112
     002650  071366                          EHT1
     002652  071466                          EDT1
     002654  071514                          EFT1

224
225                                 ;ERROR  113     'DVC' IS RESET BUT 'UNS' IS SET
226
     002656  065604                          EMT113
     002660  071366                          EHT1
     002662  071466                          EDT1
     002664  071514                          EFT1

227
228                                 ;ERROR  114     'DVC' IS SET BUT 'UNS' IS NOT SET
229
     002666  065630                          EMT114
     002670  071366                          EHT1
     002672  071466                          EDT1
     002674  071514                          EFT1

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

```
002676  065652                          EMT115
002700  071426                          EHT115
002702  071504                          EDT115
002704  071532                          EFT115

233
234                             ;ERROR  116     CANT RESET ''SKI'' USING 'MSER''
235
002706  065676                          EMT116
002710  071366                          EHT1
002712  071466                          EDT1
002714  071514                          EFT1

236
237                             ;ERROR  117     CANT SET ''SKI'' USING 'MSER''
238
002716  065716                          EMT117
002720  071366                          EHT1
002722  071466                          EDT1
002724  071514                          EFT1

239
240                             ;ERROR  120     ''SKI'' SET BY WRONG BIT
241
002726  065736                          EMT120
002730  071426                          EHT115
002732  071504                          EDT115
002734  071532                          EFT115

242
243                             ;ERROR  121     CANT RESET 'PIP'' USING 'MOC''
244
002736  065762                          EMT121
002740  071366                          EHT1
002742  071466                          EDT1
002744  071514                          EFT1

245
246                             ;ERROR  122     CANT SET 'PIP'' USING 'MOC''
247
002746  066002                          EMT122
002750  071366                          EHT1
002752  071466                          EDT1
002754  071514                          EFT1

248
249                             ;ERROR  123     'MOC'' SET BY WRONG BIT
250
002756  066022                          EMT123
002760  071426                          EHT115
002762  071504                          EDT115
002764  071532                          EFT115

251
252                             ;ERROR  124     CANT CLEAR 'EBL''
253
002766  066046                          EMT124
```

```
        002770  071366                          EHT1
        002772  071466                          EDT1
        002774  071514                          EFT1

    254
    255                             ;ERROR  125     'EBL' NOT ZERO IN DIAGNOSTIC MODE
    256
        002776  066064                          EMT125
        003000  071366                          EHT1
        003002  071466                          EDT1
        003004  071514                          EFT1

    257
    258                             ;ERROR  126     CANT SET 'EBL' USING 'DEBL'
    259
        003006  066104                          EMT126
        003010  071366                          EHT1
        003012  071466                          EDT1
        003014  071514                          EFT1

    260
    261                             ;ERROR  127     'DEBL' SET BY WRONG BIT
    262
        003016  066122                          EMT127
        003020  071426                          EHT115
        003022  071504                          EDT115
        003024  071532                          EFT115

    263
    264                             ;ERROR  130     'LS' NOT CORRECT ACCORDING TO RMDA
    265
        003026  066146                          EMT130
        003030  071432                          EHT130
        003032  071506                          EDT130
        003034  071534                          EFT130

    266
    267                             ;ERROR  131     'LST' NOT CORRECT ACCORDING TO RMDA
    268
        003036  066164                          EMT131
        003040  071432                          EHT130
        003042  071506                          EDT130
        003044  071534                          EFT130

    269
    270                             ;ERROR  132     CANNOT INCREMENT SECTOR ADDRESS USING 'DEBL'
    271
        003046  066202                          EMT132
        003050  071436                          EHT132
        003052  071510                          EDT132
        003054  071536                          EFT132

    272
    273                             ;ERROR  133     CANNOT INCREMENT TRACK ADDRESS USING 'DEBL'
    274
        003056  066222                          EMT133
        003060  071436                          EHT132
```

```
        003062  071510                          EDT132
        003064  071536                          EFT132

    275
    276                             ;ERROR  134     UNUSED BITS OF RMDC NOT ZERO
    277
        003066  066242                          EMT134
        003070  071366                          EHT1
        003072  071466                          EDT1
        003074  071514                          EFT1

    278
    279                             ;ERROR  135     'VV'' NOT RESET BY UNIT READY
    280
        003076  066256                          EMT135
        003100  071366                          EHT1
        003102  071466                          EDT1
        003104  071514                          EFT1

    281
    282                             ;ERROR  136     SERIAL NUMBER IS INCONSISTENT
    283
        003106  066274                          EMT136
        003110  071366                          EHT1
        003112  071466                          EDT1
        003114  071514                          EFT1

    284
    285                             ;ERROR  137     CANT CLEAR ''GO'' BIT
    286
        003116  066306                          EMT137
        003120  071366                          EHT1
        003122  071466                          EDT1
        003124  071514                          EFT1

    287
    288                             ;ERROR  140     CANT INCREMENT CYLINDER USING 'DEBL''
    289
        003126  066324                          EMT140
        003130  071436                          EHT132
        003132  071510                          EDT132
        003134  071536                          EFT132

    290
    291                             ;ERROR  141     CANT RESET 'LBT'' BY WRITING RMDA
    292
        003136  066344                          EMT141
        003140  071442                          EHT142
        003142  071510                          EDT132
        003144  071536                          EFT132

    293
    294                             ;ERROR  142     CANT SET 'LBT'' USING 'DEBL''
    295
        003146  066360                          EMT142
        003150  071442                          EHT142
        003152  071510                          EDT132
```

```
        003154  071536                          EFT132

    296
    297                                 ;ERROR  143     CANT READ ZERO FROM COMP ERROR
    298
        003156  066376                          EMT143
        003160  071366                          EHT1
        003162  071466                          EDT1
        003164  071514                          EFT1

    299
    300                                 ;ERROR  144     CANT SET COMP ERROR WITH RMER1 OR RMER2
    301
        003166  066412                          EMT144
        003170  071366                          EHT1
        003172  071466                          EDT1
        003174  071514                          EFT1

    302
    303                                 ;ERROR  145     COMP ERROR DID NOT SET
    304
        003176  066434                          EMT145
        003200  071446                          EHT145
        003202  071506                          EDT130
        003204  071534                          EFT130

    305
    306                                 ;ERROR  146     CANT SET ''GO'' BIT
    307
        003206  066456                          EMT146
        003210  071366                          EHT1
        003212  071466                          EDT1
        003214  071514                          EFT1

    308
    309                                 ;ERROR  147     CANT READ A ONE FROM ''TST''
    310
        003216  066474                          EMT147
        003220  071366                          EHT1
        003222  071466                          EDT1
        003224  071514                          EFT1

    311
    312                                 ;ERROR  150     ''TST'' IS INCORRECT FOR THE FUNCTION CODE
    313
        003226  066506                          EMT150
        003230  071452                          EHT150
        003232  071504                          EDT115
        003234  071532                          EFT115

    314
    315                                 ;ERROR  151     CANT SET THE ''GO'' BIT
    316
        003236  066530                          EMT151
        003240  071366                          EHT1
        003242  071466                          EDT1
        003244  071514                          EFT1
```

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

003246  066542                          EMT152
003250  071366                          EHT1
003252  071466                          EDT1
003254  071514                          EFT1

                                ;ERROR  153     ''GO'' RESET EARLY

003256  066556                          EMT153
003260  071366                          EHT1
003262  071466                          EDT1
003264  071514                          EFT1

                                ;ERROR  154     ''GO'' DIDNT RESET ON TIME

003266  066576                          EMT154
003270  071366                          EHT1
003272  071466                          EDT1
003274  071514                          EFT1

                                ;ERROR  155     CANT CLEAR CONTINUE

003276  066616                          EMT155
003300  071366                          EHT1
003302  071466                          EDT1
003304  071514                          EFT1

                                ;ERROR  156     CONTINUE IS INCORRECT FOR THE FUNCTION CODE

003306  066634                          EMT156
003310  071452                          EHT150
003312  071504                          EDT115
003314  071532                          EFT115

                                ;ERROR  157     CANT CLEAR IVC

003316  066656                          EMT157
003320  071366                          EHT1
003322  071466                          EDT1
003324  071514                          EFT1

                                ;ERROR  160     IVC IS INCORRECT FOR THE FUNCTION CODE

003326  066674                          EMT160
003330  071452                          EHT150
003332  071504                          EDT115
003334  071532                          EFT115
```

```
                                ;ERROR  161     CANT CLEAR LSC

003336  066724                          EMT161
003340  071366                          EHT1
003342  071466                          EDT1
003344  071514                          EFT1

                                ;ERROR  162     CANT SET LSC

003346  066742                          EMT162
003350  071366                          EHT1
003352  071466                          EDT1
003354  071514                          EFT1

                                ;ERROR  163     COMMAND DECODE WAS ENABLED WITH COMP ERROR SET

003356  066760                          EMT163
003360  071366                          EHT1
003362  071466                          EDT1
003364  071514                          EFT1

                                ;ERROR  164     COMMAND DECODE WAS ENABLED WITH COMP ERROR SET

003366  067002                          EMT164
003370  071366                          EHT1
003372  071466                          EDT1
003374  071514                          EFT1

                                ;ERROR  165     DECODE DOES NOT SET

003376  067024                          EMT165
003400  000000                          0
003402  000000                          0
003404  000000                          0

                                ;ERROR  166     CANT CLEAR OCCUPIED

003406  067050                          EMT166
003410  071366                          EHT1
003412  071466                          EDT1
003414  071514                          EFT1

                                ;ERROR  167     ILF SET WITHOUT GO BIT

003416  067070                          EMT167
003420  071366                          EHT1
003422  071466                          EDT1
003424  071514                          EFT1
```

```
360                                     ;ERROR  170     CANT SET VOLUME VALID
361
003426  067112                                  EMT170
003430  071452                                  EHT150
003432  071504                                  EDT115
003434  071532                                  EFT115

362
363                                     ;ERROR  171     ILF IS INCORRECT
364
003436  067124                                  EMT171
003440  071452                                  EHT150
003442  071504                                  EDT115
003444  071532                                  EFT115

365
366                                     ;ERROR  172     CANT SET OFFSET DIRECTION BIT
367
003446  067140                                  EMT172
003450  071366                                  EHT1
003452  071466                                  EDT1
003454  071514                                  EFT1

368
369                                     ;ERROR  173     OCCUPIED IS INCORRECT FOR FUNCTION CODE
370
003456  067156                                  EMT173
003460  071452                                  EHT150
003462  071504                                  EDT115
003464  071532                                  EFT115

371
372                                     ;ERROR  174     READ IN PRESET DIDNT CLEAR RMDA, RMDC OR RMOF
373
003466  067200                                  EMT174
003470  000000                                  0
003472  000000                                  0
003474  000000                                  0

374
375                                     ;ERROR  175     READ IN PRESET DIDNT CLEAR RMOF
376
003476  067226                                  EMT175
003500  071366                                  EHT1
003502  071466                                  EDT1
003504  071514                                  EFT1

377
378                                     ;ERROR  176     READ IN PRESET DIDNT CLEAR RMDA
379
003506  067244                                  EMT176
003510  071366                                  EHT1
003512  071466                                  EDT1
003514  071514                                  EFT1

380
381                                     ;ERROR  177     READ IN PRESET DIDNT CLEAR RMDC
```

```
382
    003516  067262                          EMT177
    003520  071366                          EHT1
    003522  071466                          EDT1
    003524  071514                          EFT1

383
384                             ;ERROR  200     CANT SET OFFSET MODE BY OFFSET COMMAND
385
    003526  067300                          EMT200
    003530  071366                          EHT1
    003532  071466                          EDT1
    003534  071514                          EFT1

386
387                             ;ERROR  201     CANT RESET OFFSET MODE BY RTC COMMAND
388
    003536  067316                          EMT201
    003540  071366                          EHT1
    003542  071466                          EDT1
    003544  071514                          EFT1

389
390                             ;ERROR  202     CANT RESET OFD BY RTC COMMAND
391
    003546  067334                          EMT202
    003550  071366                          EHT1
    003552  071466                          EDT1
    003554  071514                          EFT1

392
393                             ;ERROR  203     CANT RESET OM BY RMDC
394
    003556  067352                          EMT203
    003560  071366                          EHT1
    003562  071466                          EDT1
    003564  071514                          EFT1

395
396                             ;ERROR  204     CANT RESET OM BY EBL
397
    003566  067374                          EMT204
    003570  071366                          EHT1
    003572  071466                          EDT1
    003574  071514                          EFT1

398
399                             ;ERROR  205     RUN AND GO NOT CORRECT FOR FUNCTION CODE
400
    003576  067414                          EMT205
    003600  071452                          EHT150
    003602  071504                          EDT115
    003604  071532                          EFT115

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

```
003606  067434                          EMT206
003610  000000                          0
003612  000000                          0
003614  000000                          0

404
405                             ;ERROR  207     IAE IS INCORRECT FOR FUNCTION CODE
406
003616  067464                          EMT207
003620  071452                          EHT150
003622  071504                          EDT115
003624  071532                          EFT115

407
408                             ;ERROR  210     IAE IS INCORRECT FOR RMDA
409
003626  067500                          EMT210
003630  071426                          EHT115
003632  071504                          EDT115
003634  071532                          EFT115

410
411                             ;ERROR  211     IAE IS INCORRECT FOR RMDC
412
003636  067516                          EMT211
003640  071426                          EHT115
003642  071504                          EDT115
003644  071532                          EFT115

413
414                             ;ERROR  212     CANT SET AOE
415
003646  067534                          EMT212
003650  071442                          EHT142
003652  071510                          EDT132
003654  071536                          EFT132

416
417                             ;ERROR  213     RMR SET WHEN WRITING RMAS OR RMCS
418
003656  067546                          EMT213
003660  071456                          EHT213
003662  071504                          EDT115
003664  071532                          EFT115

419
420                             ;ERROR  214     CANT SET RMR
421
003666  067576                          EMT214
003670  071456                          EHT213
003672  071504                          EDT115
003674  071532                          EFT115

422
423                             ;ERROR  215     DRQ IS 0 AND PGM IS 1
424
003676  067610                          EMT215
```

```
003700  071366                          EHT1
003702  071466                          EDT1
003704  071514                          EFT1

425
426                             ;ERROR  216     DVA IS NOT SET
427
003706  067630                          EMT216
003710  071366                          EHT1
003712  071466                          EDT1
003714  071514                          EFT1

428
429                             ;ERROR  217     DPR IS NOT SET
430
003716  067644                          EMT217
003720  071366                          EHT1
003722  071466                          EDT1
003724  071514                          EFT1

431
432                             ;ERROR  220     CANT SET PORT REQUEST BY READING RMCS1
433
003726  067660                          EMT220
003730  071462                          EHT220
003732  071512                          EDT220
003734  071540                          EFT220

434
435                             ;ERROR  221     CANT SET PORT REQUEST BY WRITING RMAS
436
003736  067676                          EMT221
003740  071462                          EHT220
003742  071512                          EDT220
003744  071540                          EFT220

437
438                             ;ERROR  222     CANT SET PORT REQUEST BY WRITING RMDA
439
003746  067714                          EMT222
003750  071462                          EHT220
003752  071512                          EDT220
003754  071540                          EFT220

440
441                             ;ERROR  223     CANT RESET PORT REQUEST BY RELEASE COMMAND
442
003756  067732                          EMT223
003760  071462                          EHT220
003762  071512                          EDT220
003764  071540                          EFT220

443
444                             ;ERROR  224     CANT CLEAR ATA BY RMAS
445
003766  067750                          EMT224
003770  071366                          EHT1
```

```
003772  071466                          EDT1
003774  071514                          EFT1

446
447                             ;ERROR  225     ATA IS RESET BUT RMAS NOT ZERO
448
003776  067770                          EMT225
004000  071366                          EHT1
004002  071466                          EDT1
004004  071514                          EFT1

449
450                             ;ERROR  226     CANT RESET ATA BY GO
451
004006  070012                          EMT226
004010  071366                          EHT1
004012  071466                          EDT1
004014  071514                          EFT1

452
453                             ;ERROR  227     ATA NOT SET BY UNIT READY
454
004016  070030                          EMT227
004020  071366                          EHT1
004022  071466                          EDT1
004024  071514                          EFT1

455
456                             ;ERROR  230     ATA NOT SET BY UNIT READY
457
004026  070046                          EMT230
004030  071366                          EHT1
004032  071466                          EDT1
004034  071514                          EFT1

458
459                             ;ERROR  231     ATA NOT SET BY COMP ERROR
460
004036  070064                          EMT231
004040  071366                          EHT1
004042  071466                          EDT1
004044  071514                          EFT1

461
462                             ;ERROR  232     ATA SET/DID NOT SET WHEN REGISTER WRITTEN
463                             ;               WHILE COMP ERROR WAS SET
464
004046  070100                          EMT232
004050  071456                          EHT213
004052  071504                          EDT115
004054  071532                          EFT115

465
466                             ;ERROR  233     ATA NOT SET BY COMMAND SEQUENCER
467
004056  070122                          EMT233
004060  071452                          EHT150
```

```
004062  071504                          EDT115
004064  071532                          EFT115

468
469                             ;ERROR  234     WLE INCORRECT ACCORDING TO FUNCTION CODE
470
004066  070144                          EMT234
004070  071452                          EHT150
004072  071504                          EDT115
004074  071532                          EFT115

471
472                             ;ERROR  235     CANT CLEAR EXCEPTION
473
004076  070172                          EMT235
004100  071366                          EHT1
004102  071466                          EDT1
004104  071514                          EFT1

474
475                             ;ERROR  236     CANT SET EXCEPTION
476
004106  070212                          EMT236
004110  071426                          EHT115
004112  071504                          EDT115
004114  071532                          EFT115

477
478                             ;ERROR  237     CANT CLEAR IVC
479
004116  070244                          EMT237
004120  071366                          EHT1
004122  071466                          EDT1
004124  071514                          EFT1

480
481                             ;ERROR  240     CANT SET IVC
482
004126  070244                          EMT237
004130  071452                          EHT150
004132  071504                          EDT115
004134  071532                          EFT115

483
484                             ;ERROR  241     OPI NOT SET DURING RECALIBRATE
485
004136  070310                          EMT241
004140  071366                          EHT1
004142  071466                          EDT1
004144  071514                          EFT1

486
487                             ;ERROR  242     RECALIBRATE DID NOT ABORT WHEN DRIVE FAULT SET
488
004146  070334                          EMT242
004150  071366                          EHT1
004152  071466                          EDT1
```

```
        004154  071514                          EFT1

    489
    490                                 ;ERROR  243     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
    491                                 ;               DROPPED DURING RECALIBRATE
    492
        004156  070360                          EMT243
        004160  071366                          EHT1
        004162  071466                          EDT1
        004164  071514                          EFT1

    493
    494                                 ;ERROR  244     ATA NOT SET DURING RECALIBRATE
    495
        004166  070404                          EMT244
        004170  071366                          EHT1
        004172  071466                          EDT1
        004174  071514                          EFT1

    496
    497                                 ;ERROR  245     GO RESET EARLY DURING RECALIBRATE
    498
        004176  070422                          EMT245
        004200  071366                          EHT1
        004202  071466                          EDT1
        004204  071514                          EFT1

    499
    500                                 ;ERROR  246     GO NOT RESET DURING RECALIBRATE
    501
        004206  070440                          EMT246
        004210  071366                          EHT1
        004212  071466                          EDT1
        004214  071514                          EFT1

    502
    503                                 ;ERROR  247     INCORRECT TAG BUS DURING RECALIBRATE
    504
        004216  070464                          EMT247
        004220  071366                          EHT1
        004222  071466                          EDT1
        004224  071514                          EFT1

    505
    506                                 ;ERROR  250     OPI SHOULD HAVE SET DURING SEEK BECAUSE UNIT
    507                                 ;               READY DROPPED
    508
        004226  070502                          EMT250
        004230  071366                          EHT1
        004232  071466                          EDT1
        004234  071514                          EFT1

    509
    510                                 ;ERROR  251     SEEK DID NOT ABORT WHEN DRIVE FAULT SET
    511
        004236  070526                          EMT251
        004240  071366                          EHT1
```

```
        004242  071466                          EDT1
        004244  071514                          EFT1

    512
    513                             ;ERROR  252     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
    514                             ;               DROPPED DURING SEEK
    515
        004246  070552                          EMT252
        004250  071366                          EHT1
        004252  071466                          EDT1
        004254  071514                          EFT1

    516
    517                             ;ERROR  253     ATA NOT SET DURING SEEK
    518
        004256  070576                          EMT253
        004260  071366                          EHT1
        004262  071466                          EDT1
        004264  071514                          EFT1

    519
    520                             ;ERROR  254     GO RESET EARLY DURING SEEK
    521
        004266  070614                          EMT254
        004270  071366                          EHT1
        004272  071466                          EDT1
        004274  071514                          EFT1

    522
    523                             ;ERROR  255     GO DID NOT RESET DURING SEEK
    524
        004276  070632                          EMT255
        004300  071366                          EHT1
        004302  071466                          EDT1
        004304  071514                          EFT1

    525
    526                             ;ERROR  256     INCORRECT TAG BUS DURING SEEK
    527
        004306  070656                          EMT256
        004310  071366                          EHT1
        004312  071466                          EDT1
        004314  071514                          EFT1

    528
    529                             ;ERROR  257     OPI NOT SET DURING SEARCH
    530
        004316  070674                          EMT257
        004320  071366                          EHT1
        004322  071466                          EDT1
        004324  071514                          EFT1

    531
    532                             ;ERROR  260     SEARCH DID NOT ABORT WHEN DRIVE FAULT SET
    533
        004326  070720                          EMT260
        004330  071366                          EHT1
```

```
004332  071466                          EDT1
004334  071514                          EFT1

534
535                             ;ERROR  261     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
536                             ;               DROPPED DURING SEARCH
537
004336  070744                          EMT261
004340  071366                          EHT1
004342  071466                          EDT1
004344  071514                          EFT1

538
539                             ;ERROR  262     ATA NOT SET DURING SEARCH
540
004346  070770                          EMT262
004350  071366                          EHT1
004352  071466                          EDT1
004354  071514                          EFT1

541
542                             ;ERROR  263     GO RESET EARLY DURING SEARCH
543
004356  071006                          EMT263
004360  071366                          EHT1
004362  071466                          EDT1
004364  071514                          EFT1

544
545                             ;ERROR  264     GO DID NOT RESET DURING SEARCH
546
004366  071024                          EMT264
004370  071366                          EHT1
004372  071466                          EDT1
004374  071514                          EFT1

547
548                             ;ERROR  265     SEARCH ENABLE DIDNT SET DURING SEARCH
549
004376  071050                          EMT265
004400  071366                          EHT1
004402  071466                          EDT1
004404  071514                          EFT1

550
551                             ;ERROR  266     INCORRECT TAG BUS DURING SEARCH
552
004406  071066                          EMT266
004410  071366                          EHT1
004412  071466                          EDT1
004414  071514                          EFT1

553
554                             ;ERROR  267     OPI NOT SET BY SEARCH TIMEOUT
555
004416  071104                          EMT267
004420  071366                          EHT1
```

```
        004422  071466                          EDT1
        004424  071514                          EFT1

    556
    557                                 ;ERROR  270     OPI NOT SET DURING DATA COMMAND
    558
        004426  071120                          EMT270
        004430  071366                          EHT1
        004432  071466                          EDT1
        004434  071514                          EFT1

    559
    560                                 ;ERROR  271     DATA COMMAND DID NOT ABORT WHEN DRIVE FAULT SET
    561
        004436  071144                          EMT271
        004440  071366                          EHT1
        004442  071466                          EDT1
        004444  071514                          EFT1

    562
    563                                 ;ERROR  272     EBL RESET EARLY DURING DATA COMMAND
    564
        004446  071170                          EMT272
        004450  071366                          EHT1
        004452  071466                          EDT1
        004454  071514                          EFT1

    565
    566                                 ;ERROR  273     EBL DIDNT RESET ON TIME DURING DATA COMMAND
    567
        004456  071206                          EMT273
        004460  071366                          EHT1
        004462  071466                          EDT1
        004464  071514                          EFT1

    568
    569                                 ;ERROR  274     GO NOT RESET DURING DATA COMMAND
    570
        004466  071224                          EMT274
        004470  071366                          EHT1
        004472  071466                          EDT1
        004474  071514                          EFT1

    571
    572                                 ;ERROR  275     RUN AND GO NOT SET DURING DATA COMMAND
    573
        004476  071242                          EMT275
        004500  071366                          EHT1
        004502  071466                          EDT1
        004504  071514                          EFT1

    574
    575                                 ;ERROR  276     INCORRECT TAG BUS DURING DATA COMMAND
    576
        004506  071262                          EMT276
        004510  071366                          EHT1
        004512  071466                          EDT1
```

```
        004514  071514                          EFT1

    577
    578                             ;ERROR  277     OPI NOT SET DURING DATA COMMAND WHEN ON
    579                             ;               CYLINDER DIDNT DROP
    580
        004516  071300                          EMT277
        004520  071366                          EHT1
        004522  071466                          EDT1
        004524  071514                          EFT1

    581
    582                             ;ERROR  300     DATA COMMAND DID NOT ABORT WHEN SEEK ERROR SET
    583
        004526  071324                          EMT300
        004530  071366                          EHT1
        004532  071466                          EDT1
        004534  071514                          EFT1

    584
    585                             ;ERROR  301     SEARCH NOT ENABLED DURING DATA COMMAND
    586
        004536  071350                          EMT301
        004540  071366                          EHT1
        004542  071466                          EDT1
        004544  071514                          EFT1

    587
    588                             ;PUT ERROR TABLE HERE
    589
```

```
                                        ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS
                                        
004546  011600                  BADTMO: MOV     (SP),R0         ;SAVE PC WHERE THE TIME OUT OCCURED
004550  005740                          TST     -(R0)           ;ADJUST PC -2
004552  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004554  104401  004562                  TYPE    ,65$            ;;TYPE ASCIZ STRING
004560  000417                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
004620                          64$:
004620  010046                          MOV     R0,-(SP)        ;SETUP FOR TYPING OUT PC
004622  104402                          TYPOC
004624  000240                          NOP                     ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
                                                                ;TO STOP ON UNEXPECTED TIMEOUT.
                                        
                                        .SBTTL  START OF PROGRAM
                                        
004626  000240                  START:  NOP
004630  005227  000000                  INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
004634  001375                          BNE     .-4             ;OF WORD
004636  000005                          RESET                   ;RESET THE WORLD
                                        
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004640  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004644  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004646  022706  001154                  CMP     #SWR,R6         ;;DONE?
004652  001374                          BNE     .-6             ;;LOOP BACK IF NO
004654  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
004660  012737  057066  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
004666  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
004674  012737  057640  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004702  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004710  012737  061400  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004716  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
004724  012737  061506  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
004732  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
004740  013737  053672  053664          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
004746  005037  001206                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
004752  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
004756  112737  000001  001131          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
004764  012737  004764  001122          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
004772  012737  004772  001124          MOV     #.,$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.
005000  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005004  012737  005( )  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
005012  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005020  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005026  022777  177777  174120          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005034  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -
005036  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
005040  012716  005046          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
005044  000002                          RTI
005046  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005054  012737  000174  001156          MOV     #DISPREG,DISPLAY
```

```
        005062  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

        005066  005037  001230                  CLR     $PASS           ;;CLEAR PASS COUNT
        005072  132737  000200  001243          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
        005100  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
        005102  012737  001244  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
        005110                          67$:
    20                                  ;SETUP 'TIMEOUT'' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
    21  005110  012737  004546  000004          MOV     #BADTMO,ERRVEC  ;SETUP FOR UNEXPECTED TIMEOUT
    22  005116  012737  000300  000006          MOV     #PR6,ERRVEC+2   ;LEVEL 6
    23  005124  012746  000140                  MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
        005130  012746  005136                  MOV     #68$,-(SP)      ;;PUT NEW PC ON  STACK
        005134  000002                          RTI                     ;;POP NEW PC AND PS
        005136                          68$:
    24
    25                                  .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
        005136  005227  177777                  INC     #-1             ;;FIRST TIME?
        005142  001055                          BNE     69$             ;;BRANCH IF NO
        005144  022737  054026  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
        005152  001451                          BEQ     69$             ;;BRANCH IF YES
        005154  104401  005222                  TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
        005160  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
        005164  001012                          BNE     71$             ;;BRANCH IF YES
        005166  123727  001242  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
        005174  001406                          BEQ     71$             ;;BRANCH IF YES
        005176  023727  001154  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
        005204  001005                          BNE     72$             ;;BRANCH IF NO
        005206  104407                          GTSWR                   ;;GET SOFT-SWR SETTINGS
        005210  000403                          BR      72$
        005212  112737  000001  001150  71$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
        005220                          72$:
        005220  000426                          BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@CZRMPA0 - RM05/3/2 DISKLESS TEST, PART 1@<CRLF>
        005276                          69$:
    26
    27
    28                                  ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
    29                                  ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
    30  005276  005037  001326                  CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
    31  005302  122737  000016  000041          CMPB    #16,@#41        ;LOADED FROM AN RM05/3/2 ?
    32  005310  001160                          BNE     5$              ;BRANCH IF NOT
    33  005312  013737  000040  001326          MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
    34  005320  122737  000007  001326          CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
    35  005326  103002                          BHIS    1$              ;YES
    36  005330  105037  001326                  CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
    37  005334  005737  000042          1$:     TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
    38  005340  001425                          BEQ     3$              ;BR IF NEITHER
    39  005342  104401  005350                  TYPE    ,74$            ;;TYPE ASCIZ STRING
        005346  000412                          BR      73$             ;;GET OVER THE ASCIZ
                                        ;;74$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
        005374                          73$:
    40  005374  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
    41  005376  113716  001326                  MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
    42  005402  104403                          TYPOS                   ;TYPE THE ADDRESS
    43  005404     001                          .BYTE   1               ;ONLY 1 CHARACTER
```

```
     44 005405     000                            .BYTE   0               ;SUPRESS LEADING ZEROS
     45 005406  104401  001217                    TYPE    ,$CRLF          ;CR-LF
     46 005412  000517                            BR      5$              ;GET NUMBER OF DRIVES
     47
     48 005414  005227  177777          3$:       INC     #-1             ;FIRST TIME THRU HERE ?
     49 005420  001114                            BNE     5$              ;NO
     50 005422  104401  005430                    TYPE    ,76$            ;;TYPE ASCIZ STRING
        005426  000410                            BR      75$             ;;GET OVER THE ASCIZ
                                        ;;76$:    .ASCIZ  <CRLF>/TO TEST DRIVE /
        005450                          75$:
     51 005450  005046                            CLR     -(SP)           ;CLEAR WORD ON STACK
     52 005452  113716  001326                    MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
     53 005456  104403                            TYPOS                   ;TYPE DRIVE ADDRESS
     54 005460     001                            .BYTE   1               ;ONLY 1 CHARACTER
     55 005461     000                            .BYTE   0               ;SUPRESS LEADING ZEROS
     56 005462  104401  005470                    TYPE    ,78$            ;;TYPE ASCIZ STRING
        005466  000431                            BR      77$             ;;GET OVER THE ASCIZ
                                        ;;78$:    .ASCIZ  /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
        005552                          77$:
     57 005552  104401  005560                    TYPE    ,79$            ;;TYPE ASCIZ STRING
        005556  000435                            BR      5$              ;;GET OVER THE ASCIZ
                                        ;;79$:    .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
        005652                          5$:
     61
     62 005652  105737  001150                    TSTB    $AUTOB          ;RUNNING IN AUTO MODE ?
     63 005656  001506                            BEQ     STANDALONE      ;BR IF NO
     64 005660  012737  000377  001300            MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
     65
     66                                 ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
     67 005666                          XSIZ:
     68 005666  132737  000200  001243            BITB    #BIT7,$ENVM     ;SIZING ALLOWED ?
     69 005674  001066                            BNE     7$              ;NO
     70
     71 005676  005001                            CLR     R1              ;START FROM DRIVE 0
     72 005700  013700  001276                    MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
     73
     74 005704  136137  063052  001300  1$:       BITB    ATNTBL(R1),$DEVM        ;IS DEVICE PRESENT IN MAP ?
     75 005712  001453                            BEQ     6$              ;BR IF NO
     76 005714  012737  062675  006040            MOV     #LODEV,5$       ;GET ADDRESS OF LOAD DEVICE MESSAGE
     77 005722  005737  001326                    TST     XXDP            ;LOADED FROM RM05/3/2 ?
     78 005726  001403                            BEQ     2$              ;NO
     79 005730  123701  001326                    CMPB    XXDP,R1         ;IS THIS THE DRIVE ?
     80 005734  001426                            BEQ     3$              ;YES, TRY NEXT DRIVE
     81
     82 005736  012760  000040  000010  2$:       MOV     #CLR,RMCS2(R0)  ;CLEAR MASS BUS
     83 005744  010160  000010                    MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
     84 005750  005760  000012                    TST     RMDS(R0)        ;TRY TO ACCESS AN RM DRIVE REGISTER
     85 005754  012737  062715  006040            MOV     #NOTPRS,5$      ;GET ADDRESS OF NOT PRESENT MESSAGE
     86 005762  032760  010000  000010            BIT     #NED,RMCS2(R0)  ;IS DRIVE PRESENT ?
     87 005770  001010                            BNE     3$              ;BR IF NO
     88 005772  012737  062732  006040            MOV     #NOTAVL,5$      ;GET ADDRESS OF AVAILABLE MESSAGE
     89 006000  032760  004000  000000            BIT     #DVA,RMCS1(R0)  ;IS DRIVE AVAILABLE ?
     90 006006  001401                            BEQ     3$              ;BR IF NO
     91 006010  000414                            BR      6$
     92
     93 006012  1[illegible]  063052  001300  3$:       BICB    ATNTBL(R1),$DEVM        ;CLEAR DEVICE FROM BIT MAP
     94 006020  [illegible]04401  [illegible]1217          4$:       TYPE    ,$CRLF          ;CR-LF
```

```
     95 006024  104401  062667                TYPE    ,MSGDRV         ;TYPE 'DRIVE'
     96 006030  010146                        MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
        006032  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
        006034     002                        .BYTE   2               ;;TYPE 2 DIGIT(S)
        006035     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
     97 006036  104401                        TYPE                    ;TYPE ERROR MESSAGE
     98 006040  000000              5$:       .WORD   0               ;ADDRESS OF MESSAGE GOES HERE
     99
    100 006042  005201              6$:       INC     R1              ;INCREMENT THE DRIVE ADDRESS
    101 006044  020127  000007                CMP     R1,#7           ;ALL DRIVES ARE CHECKED ?
    102 006050  003715                        BLE     1$              ;BRANCH IF NOT
    103
    104 006052  104401  001217          7$:   TYPE    ,$CRLF          ;CR-LF
    105 006056  005004                        CLR     R4              ;THESE TWO LOOPS ARE ADDED TO
    106 006060  005304                        DEC     R4              ;WAIT FOR TTY TO FINISH TYPING.
    107 006062  001376                        BNE     .-2
    108 006064  005304                        DEC     R4
    109 006066  001376                        BNE     .-2
    110
    111 006070  000137  007000                JMP     CMNSTART        ;JUMP TO COMMON START
    112
```

```
                                .SBTTL  STANDALONE INPUT ROUTINES

006074                          STANDALONE:
006074  004737  060050                JSR     PC,$TKINT       ;INITIALIZE CONSOLE

006100  005227  177777                INC     #-1             ;FIRST TIME THRU HERE ?
006104  001426                        BEQ     3$              ;YES !!

                                ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
006106                          1$:
006106  104401  062234                TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS ?
006112  104411                        RDCHR                   ;GET RESPONSE
006114  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
006120  104401  001176                TYPE    ,$TMP1
006124  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE ?
006132  001004                        BNE     2$              ;NO!!
006134  104401  001217                TYPE    ,$CRLF          ;CR-LF
006140  000137  007000                JMP     CMNSTART        ;KEEP PREVIOUS PARAMETERS

006144  123727  001176  000116  2$:   CMPB    $TMP1,#'N       ;NO RESPONSE ?
006152  001427                        BEQ     5$              ;YES, GET NEW PARAMETERS
006154  104401  062631                TYPE    ,CNSL08         ;NO, TYPE ' ILLEGAL INPUT '
006160  000752                        BR      1$              ;TRY AGAIN

                                ;SEE IF OPERATOR WANTS HELP TEXT
006162                          3$:
006162  104401  062150                TYPE    ,MSHELP         ;WANT HELP ?
006166  104411                        RDCHR                   ;GET RESPONSE
006170  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006174  104401  001176                TYPE    ,$TMP1
006200  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ?
006206  001407                        BEQ     4$              ;YES
006210  123727  001176  000116        CMPB    $TMP1,#'N       ;WAS IT A NO RESPONSE ?
006216  001405                        BEQ     5$              ;YES
006220  104401  062631                TYPE    ,CNSL08         ;NO, TYPE ' ILLEGAL INPUT'
006224  000756                        BR      3$              ;TRY AGAIN
006226  104401  103712        4$:     TYPE    ,HELP           ;YES - TYPE HELP TEXT

                                ;SEE IF USER WANTS TO CHANGE UNIBUS ADDRESS
006232                          5$:
006232  104401  001217                TYPE    ,$CRLF          ;CR-LF
006236  104401  062201                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ?
006242  104411                        RDCHR                   ;GET RESPONSE
006244  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006250  104401  001176                TYPE    ,$TMP1
006254  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ?
006262  001407                        BEQ     6$              ;YES
006264  123727  001176  000116        CMPB    $TMP1,#'N       ;WAS IT A NO RESPONSE ?
006272  001525                        BEQ     12$             ;YES
006274  104401  062631                TYPE    ,CNSL08         ;NO, TYPE ' ILLEGAL INPUT '
006300  000756                        BR      5$+4            ;TRY AGAIN

                                ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY
006302                          6$:
006302  104401  062267                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
006306  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
006312  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
                                                TYPE    ,QUES           ;TYPE '' ? ''
                                                RDOCT                   ;GET NEW BUS ADDRESS
                                                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN ?
                                                BEQ     8$              ;YES-SKIP TO NEXT ENTRY
                                                CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE ?
                                                BLOS    7$              ;YES
                                                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
                                                BR      6$+4            ;TRY AGAIN
                                        7$:     MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS

                                        8$:     TYPE    ,CNSL03
                                                CLR     -(SP)
                                                MOVB    $VECT1,(SP)     ;GET CURRENT VECTOR ADDRESS
                                                TYPOS
                                                .BYTE   3               ;TYPE 3 DIGITS
                                                .BYTE   0               ;SUPPRESS LEADING ZEROS
                                                TYPE    ,QUES           ;TYPE '' ? ''
                                                RDOCT                   ;GET NEW VECTOR ADDRESS
                                                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
                                                BEQ     10$             ;YES-SKIP TO NEXT ENTRY
                                                CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000 ?
                                                BHI     9$              ;YES!!
                                                TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
                                                BR      8$              ;RETRY
                                        9$:     MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS

                                        10$:    TYPE    ,CNSL05
                                                CLR     -(SP)
                                                MOVB    $VECT1+1,(SP)   ;GET CURRENT BR LEVEL
                                                ASR     (SP)
                                                ASR     (SP)
                                                ASR     (SP)
                                                ASR     (SP)
                                                ASR     (SP)
                                                TYPOS
                                                .BYTE   1               ;ONLY 1 DIGIT
                                                .BYTE   0               ;SUPPRESS LEADING ZEROS
                                                TYPE    ,QUES
                                                RDOCT                   ;GET NEW PRIORITY
                                                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN ?
                                                BEQ     12$             ;YES-SKIP TO NEXT ENTRY
                                                CMP     $TMP1,#7        ;LEGAL PRIORITY ?
                                                BLT     11$             ;YES!!
                                                TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
                                                BR      10$             ;TRY AGAIN
                                        11$:    ASL     $TMP1           ;STORE NEW PRIORITY
                                                ASL     $TMP1
                                                ASL     $TMP1
                                                ASL     $TMP1
                                                ASL     $TMP1
                                                MOVB    $TMP1,$VECT1+1

                                        ;DIALOGUE TO INPUT DEVICE NUMBERS
                                        12$:
                                                INC     #-1             ;FIRST TIME THRU ?
                                                BNE     13$             ;BR IF NO
                                                TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
```

```
114 006560  104401  001217          13$:   TYPE    ,$CRLF          ;CR-LF
115 006564  005037  001300          14$:   CLR     $DEVM           ;CLEAR DEVICE MAP
116 006570  104401  062653                 TYPE    ,MSDRVS         ;TYPE 'DRIVE(S): '
117 006574  104411                         RDCHR
118 006576  012637  001176                 MOV     (SP)+,$TMP1     ;GET RESPONSE
119 006602  023727  001176  000101         CMP     $TMP1,#'A       ;IS INPUT ''A'' ?
120 006610  001007                         BNE     15$             ;NO
121 006612  104401  062134                 TYPE    ,ALL            ;YES, TYPE ''ALL'' AND GO
122 006616  012737  000377  001300         MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
123 006624  000137  005666                 JMP     XSIZ            ;AUTO SIZE.
124
125 006630  023727  001176  000015  15$:   CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
126 006636  001436                         BEQ     17$             ;YES
127 006640  104401  001176                 TYPE    ,$TMP1          ;ECHO RESPONSE
128 006644  023727  001176  000060         CMP     $TMP1,#'0       ;NUMBER < 0 ?
129 006652  002430                         BLT     17$             ;YES
130 006654  023727  001176  000067         CMP     $TMP1,#'7       ;NUMBER > 7 ?
131 006662  003427                         BLE     18$             ;NO
132 006664  000423                         BR      17$             ;ILLEGAL INPUT
133
134 006666  104411                 16$:    RDCHR
135 006670  012637  001176                 MOV     (SP)+,$TMP1     ;GET RESPONSE
136 006674  023727  001176  000015         CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
137 006702  001432                         BEQ     19$             ;YES
138 006704  104401  062145                 TYPE    ,COMMA          ;TYPE ', '
139 006710  104401  001176                 TYPE    ,$TMP1          ;ECHO RESPONSE
140 006714  023727  001176  000060         CMP     $TMP1,#'0       ;NUMBER < 0 ?
141 006722  002404                         BLT     17$             ;YES
142 006724  023727  001176  000067         CMP     $TMP1,#'7       ;NUMBER > 7 ?
143 006732  003403                         BLE     18$             ;NO
144 006734  104401  062631          17$:   TYPE    ,CNSL08         ;TYPE '' ?ILLEGAL INPUT''
145 006740  000711                         BR      14$             ;RETRY
146
147 006742  013701  001176          18$:   MOV     $TMP1,R1        ;R1 = DRIVE NUMBER
148 006746  042701  177770                 BIC     #^C7,R1
149 006752  156137  063052  001300         BISB    ATNTBL(R1),$DEVM        ;SET DEVICE IN MAP
150 006760  122737  000377  001300         CMPB    #377,$DEVM      ;DONE ?
151 006766  101337                         BHI     16$             ;NO
152 006770  104401  001217          19$:   TYPE    ,$CRLF          ;CR-LF
153 006774  000137  005666                 JMP     XSIZ            ;GO SIZE DEVICES
154
```

```
                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
                                        CMNSTART:
007000  013700  001300                          MOV     $DEVM,R0        ;R0 = DEVICE MAP
007004  012701  001464                          MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
007010  010137  001462                          MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
007014  012702  000001                          MOV     #1,R2           ;R2 = DEVICE POINTER
007020  005003                                  CLR     R3              ;R3 = DEVICE NUMBER
007022  030200                          1$:     BIT     R2,R0           ;IS THIS DEVICE IN MAP ?
007024  001406                                  BEQ     2$              ;NO !!
007026  010311                                  MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
007030  116361  063052  000001                  MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
007036  062701  000002                          ADD     #2,R1           ;ADVANCE ENTRY POINTER
007042  006302                          2$:     ASL     R2              ;ADVANCE DEVICE POINTER
007044  105702                                  TSTB    R2              ;DONE ALL DEVICES ?
007046  001402                                  BEQ     3$              ;YES
007050  005203                                  INC     R3              ;ADVANCE DEVICE NUMBER
007052  000763                                  BR      1$              ;ENTER NEXT DEVICE
007054  005011                          3$:     CLR     (R1)            ;TERMINATE TEST QUE

                                        ;SIZE FOR CLOCK
007056  004737  054760                          JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
007062  000413                                  BR      5$              ;YES - CLOCK IS PRESENT
007064                                  4$:
007064  104000                                  EMT
007066  104401  007074                          TYPE    ,65$            ;;TYPE ASCIZ STRING
007072  000405                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/PROG HLT/
007106                                  64$:
007106  000000                                  HALT                    ;PROGRAM HALT !!
007110  000765                                  BR      4$
007112                                  5$:
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
007112  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
007116  001012                                  BNE     66$             ;;BRANCH IF YES
007120  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
007126  001406                                  BEQ     66$             ;;BRANCH IF YES
007130  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
007136  001005                                  BNE     67$             ;;BRANCH IF NO
007140  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
007142  000403                                  BR      67$
007144  112737  000001  001150          66$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
007152                                  67$:

007152  000240                          READY:  NOP                     ;READY TO START TEST
007154  105737  001300                          TSTB    $DEVM           ;ANY DRIVES IN MAP ?
007160  001007                                  BNE     2$              ;BR IF YES
007162  005737  000042                          TST     @#42            ;ANY MONITOR PRESENT ?
007166  001002                                  BNE     1$              ;BR IF YES
007170  000137  004626                          JMP     START           ;JUMP TO START
007174  000137  053636          1$:     JMP     $EOP                    ;RETURN CONTROL TO MONITOR

007200  105037  001116          2$:     CLRB    $ISTNM                  ;RESET TEST NUMBER
007204  005037  001206                  CLR     $TIMES                  ;INITIALIZE NUMBER OF ITERATIONS
007210  004737  060050                  JSR     PC,$TKINT               ;INITIALIZE TTY
007214  012746  000300                  MOV     #PR6,-(SP)              ;;PUT NEW PS ON STACK
007220  012746  007226                  MOV     #64$,-(SP)              ;;PUT NEW PC ON  STACK
007224  000002                          RTI                             ;;POP NEW PC AND PS
```

```
                007226                          64$:
     41 007226  117737  172230  001234                  MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
     42
     43                                         ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND DETERMINE THE LAST TRACK
     44                                         ;OF THE DIFFERENT DRIVE TYPES
     45 007234  012737  002000  001330                  MOV     #TA4,LSTRK      ;ASSUME LAST TRACK FOR RM02/3 = 4.
     46 007242  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     47 007246  012760  000040  000010                  MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
     48 007254  117760  172202  000010                  MOVB    @TSTQUE,RMCS2(R0)       ;SELECT DEVICE UNDER TEST
     49 007262  016002  000026                          MOV     RMDT(R0),R2     ;GET RMDT AND
     50 007266  042702  177770                          BIC     #177770,R2      ;SAVE DRIVE TYPE BITS
     51 007272  022702  000007                          CMP     #7,R2           ;IS IT AN RM05 ?
     52 007276  001003                                  BNE     3$              ;NO, MUST BE AN RM02 OR RM03
     53 007300  012737  011000  001330                  MOV     #TA16!TA2,LSTRK ;YES--SET LAST TRACK = 18.
     54 007306                                  3$:
     55
```

```
                        .SBTTL  REGISTER AND STORAGE USAGE

                        ;REGISTER ASSIGNMENTS
                        ;
                        ;R0     =       UNIBUS ADDRESS OF RH CONTROLLER
                        ;R1     =       ADDRESS OF ENTRY IN TEST QUE CORRESPONDING TO THE
                        ;               UNIT UNDER TEST
                        ;R2,R3  =       WORKING REGISTERS FOR TEST IN PROGRESS, MUST BE
                        ;               SAVED BY SUBROUTINES
                        ;R4,R5  =       GENERAL WORKING REGISTERS, ARE NOT SAVED BY
                        ;               SUBROUTINES
                        ;R6     =       STACK POINTER
                        ;R7     =       LINKAGE REGISTER TO SUBROUTINES

                        ;STORAGE ASSIGNMENTS
                        ;
                        ;$TMP0-$TMP4    TEMPORARY STORAGE, NOT SAVED BY SUBROUTINES
                        ;$GDDAT,$BDDAT  EXPECTED AND RRECEIVED STATUS FOR ERROR TYPEOUT
                        ;$GDADR,$BDADR  ADDRESS OF EXPECTED AND RECEIVED STATUS IF APPLICABLE,
                        ;               ALSO THE ADDRESS OF A REGISTER ERROR
                        ;
                        ;$TSTN  =       TEST NUMBER
                        ;$UNIT  =       NUMBER OF DEVICE BEING TESTED
                        ;RGINBF =       THE REGISTER INPUT BUFFER HAS A STORAGE LOCATION FOR
                        ;               EACH REGISTER, AND IS USED WHEN READING STATUS AND
                        ;               CONTROL DATA
                        ;RGOTBF =       THE REGISTER OUTPUT BUFFER HAS A STORAGE LOCATION FOR
                        ;               EACH REGISTER, AND IS USED FOR ASSEMBLING DATA TO BE
                        ;               WRITTEN IN REGISTERS
```

```
     1                                  ;;*****************************************************************
                                        ;*TEST 1         TRANSFER TEST
                                        ;;*****************************************************************
                                        ;;
     007306                             TST1:
     007306  000004                             SCOPE                   ;SCOPE CALL
     007310  000240                             NOP
     007312  012706  001100                     MOV     #STACK,SP       ;LOAD THE STACK POINTER
     007316  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     007322  013701  001462                     MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     007326  012737  000001  001226             MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
     2
     3 007334  012702  000000                   MOV     #0,R2           ;R2 = REGISTER INDEX
     4
     5                                  ;CLEAR THE MASSBUS AND VERIFY THAT NONEXISTANT DEVICE ERROR IS RESET
     6 007340                           10$:
     7 007340  004737  055614                   JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
     8 007344  016037  000010  001142           MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
     9 007352  032737  010000  001142           BIT     #NED,$BDDAT
    10 007360  001417                           BEQ     20$
    11 007362  111137  001140                   MOVB    (R1),$GDDAT
    12 007366  042737  177770  001140           BIC     #^CUNTMSK,$GDDAT
    13 007374  052737  000100  001140           BIS     #IR,$GDDAT
    14 007402  010037  001136                   MOV     R0,$BDADR
    15 007406  062737  000010  001136           ADD     #RMCS2,$BDADR
    16 007414  104001                           EMT     1
    17 007416  000475                           BR      60$
    18
    19                                  ;READ THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE READ
    20                                  ;DOES NOT SET 'NED' ERROR
    21 007420                           20$:
    22 007420  010003                           MOV     R0,R3           ;R3 = REGISTER ADDRESS
    23 007422  060203                           ADD     R2,R3
    24 007424  011304                           MOV     (R3),R4         ;READ REGISTER
    25 007426  032760  010000  000010           BIT     #NED,RMCS2(R0)  ;IS 'NED' SET??
    26 007434  001470                           BEQ     70$             ;NO!!
    27
    28 007436  004737  055614                   JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
    29 007442  016037  000010  001142           MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
    30 007450  032737  010000  001142           BIT     #NED,$BDDAT
    31 007456  001417                           BEQ     30$
    32 007460  111137  001140                   MOVB    (R1),$GDDAT
    33 007464  042737  177770  001140           BIC     #^CUNTMSK,$GDDAT
    34 007472  052737  000100  001140           BIS     #IR,$GDDAT
    35 007500  010037  001136                   MOV     R0,$BDADR
    36 007504  062737  000010  001136           ADD     #RMCS2,$BDADR
    37 007512  104001                           EMT     1
    38 007514  000436                           BR      60$
    39
    40                                  ;WRITE THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE WRITE
    41                                  ;DOES NOT SET 'NED' ERROR
    42 007516                           30$:
    43 007516  012713  000000                   MOV     #0,(R3)         ;WRITE REGISTER
    44 007522  032760  010000  000010           BIT     #NED,RMCS2(R0)  ;IS 'NED' SET??
    45 007530  001432                           BEQ     70$             ;NO!!
    46
    47                                  ;COULD NOT READ OR WRITE THE REGISTER WITHOUT SETING 'NED' ERROR -
```

SEQ 0090

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

                                        ;GOT 'NONEXISTENT DEVICE' ERROR FOR EVERY REMOTE REGISTER ADDRESS
007602                                  50$:
007602  013737  001276  001136                MOV     $BASE,$BDADR    ;STORE BASE ADDRESS
007610  104002                              EMT     2
007612  000137  053600              60$:    JMP     $EOSP           ;GO SELECT NEXT DEVICE

007616                                  70$:

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

                                        ;;**********************************************************************
007616                                  TST2:
007616  000004                              SCOPE                   ;SCOPE CALL
007620  000240                              NOP
007622  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
007626  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
007632  013701  001462                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
007636  012737  000002  001226                MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

007644  004737  055614                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

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

                                        ;READ REMOTE REGISTERS TWICE
007700  012702  000001                      MOV     #1,R2
007704                                  10$:
007704  016037  000000  001332                MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
007712  016037  000006  001340                MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
007720  016037  000034  001366                MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
007726  016037  000032  001364                MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
007734  005302                              DEC     R2
007736  100362                              BPL     10$

                                        ;SEE IF ANY ONE BITS CAME BACK
007740  042737  177701  001332                BIC     #^CILF76,RMCS1I ;IS RMCS1 0??
```

```
                                BNE     20$             ;NO!!
                                TST     RMDAI           ;IS RMDA 0??
                                BNE     20$             ;NO!!
                                BIC     #XNUDC,RMDCI    ;IS RMDC 0??
                                BNE     20$             ;NO!!
                                BIC     #XNUOF,RMOFI    ;IS RMOF 0 ??
                                BNE     20$             ;NO!!

                        ;CANNOT READ/WRITE ANY ONE FROM REMOTE REGISTER
                                EMT     3
                        20$:

                        ;;*****************************************************************
                        ;*TEST 3        MASSBUS INITIALIZE TEST

                        ;;*****************************************************************
                        TST3:
                                SCOPE                   ;SCOPE CALL
                                NOP
                                MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                                MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

                                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

                        ;WRITE ONES IN SELECTED REGISTERS
                                MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
                                MOV     #-1,RMER1(R0)   ;LOAD RMER1
                                MOV     #-1,RMER2(R0)   ;LOAD RMER2

                        ;INITIALIZE MASSBUS WITH A CLEAR
                                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

                        ;READ THE REGISTERS THAT WERE WRITTEN
                                MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
                                MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
                                MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                        ;SEE IF ANY REGISTER BITS WERE CLEARED
                                BIS     #^CILF76,RMCS1I ;SET ANY BIT NOT WRITTEN
                                BIS     #XNUER2,RMER2I
                                CMP     #-1,RMCS1I      ;ANY ZEROS IN RMCS1??
                                BNE     10$             ;YES!!
                                CMP     #-1,RMER1I      ;ANY ZEROS IN RMERI??
                                BNE     10$             ;YES!!
                                CMP     #-1,RMER2I      ;ANY ZEROS IN RMER2??
                                BNE     10$

                        ;NONE OF THE BITS WERE CLEARED
                                EMT     4
                        10$:

                        ;;*****************************************************************
                        ;*TEST 4        CLEAR STUCK ACTIVE TEST

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

```
        010150                                  TST4:
        010150  000004                                 SCOPE                   ;SCOPE CALL
        010152  000240                                 NOP
        010154  012706  001100                         MOV     #STACK,SP       ;LOAD THE STACK POINTER
        010160  013700  001276                         MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        010164  013701  001462                         MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        010170  012737  000004  001226                 MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
   138
   139 010176  004737  055614                         JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
   140
   141                                          ;WRITE ONES IN TEST REGISTERS
   142 010202  012760  177777  000014                 MOV     #-1,RMER1(R0)   ;LOAD RMER1
   143 010210  012760  177777  000042                 MOV     #-1,RMER2(R0)   ;LOAD RMER2
   144 010216  012760  000001  000024                 MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
   145
   146                                          ;READ TEST REGISTERS AND SEE IF ANY BITS ARE ON
   147 010224  016037  000014  001346                 MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
   148 010232  016037  000042  001374                 MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
   149 010240  016037  000024  001356                 MOV     RMMR1(R0),RMMR1I        ;STORE RMMR1 IN INPUT BUFFER
   150 010246  042737  040000  001346                 BIC     #UNS,RMER1I     ;DONT ACCEPT UNSAFE
   151 010254  001011                                 BNE     10$             ;BRANCH IF ANY OTHER BITS ON
   152 010256  042737  040200  001374                 BIC     #SKI!DVC,RMER2I ;DONT ACCEPT SKI OR DVC
   153 010264  001005                                 BNE     10$             ;BRANCH IF ANY OTHER BITS ON
   154 010266  032737  000001  001356                 BIT     #DMD,RMMR1I     ;BRANCH IF DMC IS ON
   155 010274  001001                                 BNE     10$
   156 010276  104026                                 EMT     26
   157 010300                                  10$:
   158
   159                                          ;;*****************************************************************
                                                ;*TEST 5         TRISTATE TRANSFER TEST

                                                ;;*****************************************************************
        010300                                  TST5:
        010300  000004                                 SCOPE                   ;SCOPE CALL
        010302  000240                                 NOP
        010304  012706  001100                         MOV     #STACK,SP       ;LOAD THE STACK POINTER
        010310  013700  001276                         MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        010314  013701  001462                         MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        010320  012737  000005  001226                 MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
   160
   161 010326  005002                                 CLR     R2              ;CLEAR ERROR FLAGS
   162 010330  004737  055614                         JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
   163
   164                                          ;WRITE ONES IN SELECTED REGISTERS
   165 010334  012760  000076  000000                 MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
   166 010342  012760  177777  000006                 MOV     #-1,RMDA(R0)    ;LOAD RMDA
   167 010350  012760  177777  000014                 MOV     #-1,RMER1(R0)   ;LOAD RMER1
   168 010356  012760  177777  000032                 MOV     #-1,RMOF(R0)    ;LOAD RMOF
   169 010364  012760  177777  000042                 MOV     #-1,RMER2(R0)   ;LOAD RMER2
   170
   171                                          ;WRITE ZEROS IN SELECTED REGISTERS
   172 010372  012760  000000  000000                 MOV     #0,RMCS1(R0)    ;LOAD RMCS1
   173 010400  012760  000000  000006                 MOV     #0,RMDA(R0)     ;LOAD RMDA
   174 010406  012760  000000  000014                 MOV     #0,RMER1(R0)    ;LOAD RMER1
   175 010414  012760  000000  000032                 MOV     #0,RMOF(R0)     ;LOAD RMOF
   176 010422  012760  000000  000034                 MOV     #0,RMDC(R0)     ;LOAD RMDC
   177 010430  012760  000000  000042                 MOV     #0,RMER2(R0)    ;LOAD RMER2
```

```
                                        ;READ BACK ALL REGISTERS
010436  016037  000000  001332                MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
010444  016037  000006  001340                MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
010452  016037  000014  001346                MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
010460  016037  000032  001364                MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
010466  016037  000034  001366                MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
010474  016037  000042  001374                MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                        ;CHECK EACH REGISTER CONTENT FOR ZERO BITS WRITTEN & READ
010502  012702  177777                      MOV     #-1,R2          ;ACCUMULATE ZEROS IN R2
010506  052737  177701  001332              BIS     #^CILF76,RMCS1I ;SET ALL BITS NOT WRITTEN
010514  052737  161577  001364              BIS     #XNUOF,RMOFI
010522  052737  176000  001366              BIS     #XNUDC,RMDCI
010530  052737  001567  001374              BIS     #XNUER2,RMER2I
010536  005137  001332                      COM     RMCS1I          ;COMPLEMENT REGISTER CONTENTS
010542  005137  001340                      COM     RMDAI
010546  005137  001346                      COM     RMER1I
010552  005137  001364                      COM     RMOFI
010556  005137  001366                      COM     RMDCI
010562  005137  001374                      COM     RMER2I
010566  043702  001332                      BIC     RMCS1I,R2               ;ACCUMULATE ALL ZERO BITS
010572  043702  001340                      BIC     RMDAI,R2
010576  043702  001346                      BIC     RMER1I,R2
010602  043702  001364                      BIC     RMOFI,R2
010606  043702  001366                      BIC     RMDCI,R2
010612  043702  001374                      BIC     RMER2I,R2
010616  001407                              BEQ     10$             ;BRANCH IF EACH BIT IS ZERO

                                        ;ONE OR MORE BIT POSITIONS ARE NOT ZERO
010620  010237  001142                      MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
010624  005037  001140                      CLR     $GDDAT          ;LOAD EXPECTED RESULT
010630  104005                              EMT     5
010632  052702  000001                      BIS     #BIT0,R2        ;SET ERROR FLAG
010636                                  10$:
010636  004737  055614                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

                                        ;PRESET SELECTED REGISTERS TO ZEROS
                                        ;(ASSUME RMCS1, RMER1, RMER2 WERE CLEARED BY INIT)
010642  012760  000000  000006              MOV     #0,RMDA(R0)     ;LOAD RMDA
010650  012760  000000  000032              MOV     #0,RMOF(R0)     ;LOAD RMOF
010656  012760  000000  000034              MOV     #0,RMDC(R0)     ;LOAD RMDC

                                        ;WRITE ONES IN SELECTED REGISTERS
010664  012760  000076  000000              MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
010672  012760  177777  000006              MOV     #-1,RMDA(R0)    ;LOAD RMDA
010700  012760  016200  000032              MOV     #^CXNUOF,RMOF(R0)       ;LOAD RMOF
010706  012760  001777  000034              MOV     #^CXNUDC,RMDC(R0)       ;LOAD RMDC
010714  012760  177777  000014              MOV     #-1,RMER1(R0)   ;LOAD RMER1
010722  012760  176210  000042              MOV     #^CXNUER2,RMER2(R0)     ;LOAD RMER2

                                        ;READ ALL REGISTERS
010730  016037  000000  001332              MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
010736  016037  000006  001340              MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
010744  016037  000032  001364              MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
010752  016037  000034  001306              MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
010760  016037  000014  001346              MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
```

```
    234 010766  016037  000042  001374          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
    235
    236                                 ;CHECK EACH REGISTER CONTENT FOR ONE BITS WRITTEN & READ
    237 010774  042737  177701  001332          BIC     #^CILF76,RMCS1I         ;CLEAR ALL BITS NOT WRITTEN
    238 011002  042737  161577  001364          BIC     #XNJOF,RMOFI
    239 011010  042737  176000  001366          BIC     #XNUDC,RMDCI
    240 011016  042737  001567  001374          BIC     #XNUER2,RMER2I
    241 011024  005002                          CLR     R2                      ;ACCUMULATE ONES IN R2
    242 011026  053702  001332                  BIS     RMCS1I,R2               ;ACCUMULATE ALL ONE BITS
    243 011032  053702  001340                  BIS     RMDAI,R2
    244 011036  053702  001364                  BIS     RMOFI,R2
    245 011042  053702  001366                  BIS     RMDCI,R2
    246 011046  053702  001346                  BIS     RMER1I,R2
    247 011052  053702  001374                  BIS     RMER2I,R2
    248 011056  022702  177777                  CMP     #-1,R2                  ;SEE IF EACH BIT POSITION WAS ONE
    249 011062  001410                          BEQ     20$                     ;BRANCH IF NONE STUCK
    250
    251                                 ;ONE OR MORE BIT POSITIONS ARE NOT ONE
    252 011064  010237  001142                  MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
    253 011070  012737  177777  001140          MOV     #-1,$GDDAT              ;EXPECTED RESULT
    254 011076  104006                          EMT     6
    255 011100  052702  000002                  BIS     #BIT1,R2                ;SET ERROR FLAG
    256 011104                          20$:
    257 011104  005702                          TST     R2                      ;ANY ERRORS DETECTED ??
    258 011106  001126                          BNE     30$                     ;YES - DONT DO BIT TEST
    259 011110  012702  000001                  MOV     #1,R2                   ;R2=BIT POSITION
    260 011114                          25$:
        011114  004737  055614                  JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
    261
    262                                 ;WRITE THE BIT PATTERN IN SELECTED DEVICE REGISTERS
    263 011120  010260  000006                  MOV     R2,RMDA(R0)             ;LOAD RMDA
    264 011124  010260  000032                  MOV     R2,RMOF(R0)             ;LOAD RMOF
    265 011130  010260  000034                  MOV     R2,RMDC(R0)             ;LOAD RMDC
    266 011134  010260  000014                  MOV     R2,RMER1(R0)            ;LOAD RMER1
    267 011140  010260  000042                  MOV     R2,RMER2(R0)            ;LOAD RMER2
    268
    269                                 ;READ BACK THE REGISTERS
    270 011144  016037  000006  001340          MOV     RMDA(R0),RMDAI          ;STORE RMDA IN INPUT BUFFER
    271 011152  016037  000032  001364          MOV     RMOF(R0),RMOFI          ;STORE RMOF IN INPUT BUFFER
    272 011160  016037  000034  001366          MOV     RMDC(R0),RMDCI          ;STORE RMDC IN INPUT BUFFER
    273 011166  016037  000014  001346          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
    274 011174  016037  000042  001374          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
    275
    276                                 ;CHECK REGISTER CONTENTS FOR CORRECT PATTERN
    277 011202  005003                          CLR     R3                      ;R3=ACCUMULATED ONE BIT
    278 011204  012704  177777                  MOV     #-1,R4                  ;R4=ACCUMULATED ZERO BITS
    279 011210  013705  001340                  MOV     RMDAI,R5                ;GET ANY GOOD BITS FROM RMDA
    280 011214  050503                          BIS     R5,R3
    281 011216  005105                          COM     R5
    282 011220  040504                          BIC     R5,R4
    283 011222  013705  001364                  MOV     RMOFI,R5                ;GET GOOD BITS FROM RMOF
    284 011226  042705  161577                  BIC     #XNUOF,R5
    285 011232  050503                          BIS     R5,R3
    286 011234  005105                          COM     R5
    287 011236  042705  161577                  BIC     #XNUOF,R5
    288 011242  040504                          BIC     R5,R4
    289 011244  013705  001366                  MOV     RMDCI,R5                ;GET GOOD BITS FROM RMDC
```

```
    290 011250  042705  176000                  BIC     #XNUDC,R5
    291 011254  050503                          BIS     R5,R3
    292 011256  005105                          COM     R5
    293 011260  042705  176000                  BIC     #XNUDC,R5
    294 011264  040504                          BIC     R5,R4
    295 011266  013705  001346                  MOV     RMER1I,R5               ;GET GOOD BITS FROM RMER1
    296 011272  050503                          BIS     R5,R3
    297 011274  005105                          COM     R5
    298 011276  040504                          BIC     R5,R4
    299 011300  013705  001374                  MOV     RMER2I,R5               ;GET GOOD BITS FROM RMER2
    300 011304  042705  001567                  BIC     #XNUER2,R5
    301 011310  050503                          BIS     R5,R3
    302 011312  005105                          COM     R5
    303 011314  042705  001567                  BIC     #XNUER2,R5
    304 011320  040504                          BIC     R5,R4
    305 011322  010205                          MOV     R2,R5           ;RESET ALL ONES IN R3 EXCEPT
    306 011324  005105                          COM     R5              ,FOR THE TEST BIT
    307 011326  040503                          BIC     R5,R3
    308 011330  040204                          BIC     R2,R4           ;RESET TEST BIT IN R4
    309 011332  050403                          BIS     R4,R3           ;COMBINE ACCUMULATED 1'S + 0'S
    310 011334  020302                          CMP     R3,R2           ;IS PATTERN OK??
    311 011336  001406                          BEQ     26$             ;YES!!
    312 011340  010237  001140                  MOV     R2,$GDDAT               ;SAVE TEST PATTERN
    313 011344  010337  001142                  MOV     R3,$BDDAT               ;SAVE RESULT
    314 011350  104007                          EMT     7
    315 011352  000404                          BR      30$             ;SKIP TO NEXT
    316
    317                                 ;ADVANCE R2 TO THE NEXT PATTERN AND REPEAT TEST
    318 011354                          26$:
    319 011354  006302                          ASL     R2              ;SHIFT THE BIT
    320 011356  001402                          BEQ     30$             ;EXIT IF DONE
    321 011360  000137  011114                  JMP     25$
    322 011364                          30$:
    323
    329                                 ;;*****************************************************************
                                        ;*TEST 6         REGISTER SELECT TEST
                                        ;*
                                        ;*NOTE: REGISTER SELECT 16 IS TESTED BY THE ''ILR'' TEST
                                        ;*
                                        ;;*****************************************************************
        011364                          TST6:
        011364  000004                          SCOPE                   ;SCOPE CALL
        011366  000240                          NOP
        011370  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        011374  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        011400  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        011404  012737  000006  001226          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    330
    331                                 ;THE FOLLOWING TABLE GIVES MASSBUS REGISTER SELECT VALUES FOR
    332                                 ;EACH DEVICE REGISTER
    333
    334                                 ;       REGISTER        REG SEL
    335                                 ;       NAME            (16,8,4,2,1)
    336                                 ;
    337                                 ;       RMCS1           00000
    338                                 ;       RMDS            00001
    339                                 ;       RMER1           00010
```

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

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

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

                                ;TEST REG SEL 1 FOR S-A-0
011420  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
011424  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
011430  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
011434  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
011440  010360  000036                  MOV     R3,RMHR(R0)     ;LOAD RMHR
011444  016037  000014  001346          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
011452  016037  000034  001366          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
011460  020337  001346                  CMP     R3,RMER1I
011464  001007                          BNE     10$
011466  052737  176000  001366          BIS     #XNUDC,RMDCI
011474  020337  001366                  CMP     R3,RMDCI
011500  001001                          BNE     10$
011502  104010                          EMT     10

                                ;TEST REG SEL 1 FOR S-A-1
011504                          10$:
011504  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
011510  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
011514  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
011520  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
011524  010360  000016                  MOV     R3,RMAS(R0)     ;LOAD RMAS
011530  010360  000030                  MOV     R3,RMSN(R0)     ;LOAD RMSN
011534  010360  000040                  MOV     R3,RMMR2(R0)    ;LOAD RMMR2
011540  016037  000006  001340          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
011546  016037  000032  001364          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
011554  016037  000042  001374          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
011562  020337  001340                  CMP     R3,RMDAI
011566  001015                          BNE     20$
011570  052737  161577  001364          BIS     #XNUOF,RMOFI
011576  020337  001364                  CMP     R3,RMOFI
```

```
 397 011602  001007                          BNE     20$
 398 011604  052737  001567  001374          BIS     #XNUER2,RMER2I
 399 011612  020337  001374                  CMP     R3,RMER2I
 400 011616  001001                          BNE     20$
 401 011620  104011                          EMT     11
 402
 403                                 ;TEST REG SEL 2 FOR S-A-0
 404 011622                          20$:
 405 011622  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 406 011626  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
 407 011632  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
 408 011636  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
 409 011642  010360  000020                  MOV     R3,RMLA(R0)     ;LOAD RMLA
 410 011646  010360  000036                  MOV     R3,RMHR(R0)     ;LOAD RMHR
 411 011652  010360  000046                  MOV     R3,RMEC2(R0)    ;LOAD RMEC2
 412 011656  016037  000006  001340          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
 413 011664  016037  000032  001364          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
 414 011672  016037  000042  001374          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
 415 011700  020337  001340                  CMP     R3,RMDAI
 416 011704  001015                          BNE     30$
 417 011706  052737  161577  001364          BIS     #XNUOF,RMOFI
 418 011714  020337  001364                  CMP     R3,RMOFI
 419 011720  001007                          BNE     30$
 420 011722  052737  001567  001374          BIS     #XNUER2,RMER2I
 421 011730  020337  001374                  CMP     R3,RMER2I
 422 011734  001001                          BNE     30$
 423 011736  104012                          EMT     12
 424
 425                                 ;TEST REG SEL 2 FOR S-A-1
 426 011740                          30$:
 427 011740  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 428 011744  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
 429 011750  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
 430 011754  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
 431 011762  010360  000030                  MOV     R3,RMSN(R0)     ;LOAD RMSN
 432 011766  016037  000014  001346          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
 433 011774  016037  000034  001366          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
 434 012002  052737  177701  001346          BIS     #^CILF76,RMER1I
 435 012010  020337  001346                  CMP     R3,RMER1I
 436 012014  001007                          BNE     40$
 437 012016  052737  176000  001366          BIS     #XNUDC,RMDCI
 438 012024  020337  001366                  CMP     R3,RMDCI
 439 012030  001001                          BNE     40$
 440 012032  104013                          EMT     13
 441
 442                                 ;TEST REG SEL 4 FOR S-A-0
 443 012034                          40$:
 444 012034  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 445 012040  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
 446 012044  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
 447 012050  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
 448 012054  010360  000026                  MOV     R3,RMDT(R0)     ;LOAD RMDT
 449 012060  010360  000042                  MOV     R3,RMER2(R0)    ;LOAD RMER2
 450 012064  010360  000044                  MOV     R3,RMEC1(R0)    ;LOAD RMEC1
 451 012070  016037  000014  001346          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
 452 012076  016037  000032  001364          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
 453 012104  016037  000034  001366          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
```

```
                                        CMP     R3,RMER1I
                                        BNE     50$
                                        BIS     #XNUOF,RMOFI
                                        CMP     R3,RMOFI
                                        BNE     50$
                                        BIS     #XNUDC,RMDCI
                                        CMP     R3,RMDCI
                                        BNE     50$
                                        EMT     14

                                ;TEST REG SEL 4 FOR S-A-1
                                50$:
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        MOV     R2,RMDA(R0)     ;LOAD RMDA
                                        MOV     R2,RMER2(R0)    ;LOAD RMER2
                                        MOV     R3,RMDS(R0)     ;LOAD RMDS
                                        MOV     R3,RMOF(R0)     ;LOAD RMOF
                                        MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
                                        MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
                                        CMP     R3,RMDAI
                                        BNE     60$
                                        BIS     #XNUER2,RMER2I
                                        CMP     R3,RMER2I
                                        BNE     60$
                                        EMT     15

                                ;TEST REG SEL 8 FOR S-A-0
                                60$:
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        MOV     R2,RMER1(R0)    ;LOAD RMER1
                                        MOV     R2,RMDA(R0)     ;LOAD RMDA
                                        MOV     R3,RMDC(R0)     ;LOAD RMDC
                                        MOV     R3,RMER2(R0)    ;LOAD RMER2
                                        MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
                                        MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
                                        CMP     R3,RMER1I
                                        BNE     70$
                                        CMP     R3,RMDAI
                                        BNE     70$
                                        EMT     16

                                ;TEST REG SEL 8 FOR S-A-1
                                70$:
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        MOV     R2,RMOF(R0)     ;LOAD RMOF
                                        MOV     R2,RMDC(R0)     ;LOAD RMDC
                                        MOV     R2,RMER2(R0)    ;LOAD RMER2
                                        MOV     R3,RMDS(R0)     ;LOAD RMDS
                                        MOV     R3,RMER1(R0)    ;LOAD RMER1
                                        MOV     R3,RMDA(R0)     ;LOAD RMDA
                                        MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
                                        MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
                                        MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
                                        BIS     #XNUOF,RMOFI
                                        BNE     80$
                                        CMP     #XNUDC,RMDCI
                                        CMP     R3,RMDCI
```

```
                                                                    I 8
    511 012414  001007                          BNE     80$
    512 012416  052737  001567  001374          BIS     #XNUER2,RMER2I
    513 012424  020337  001374                  CMP     R3,RMER2I
    514 012430  001001                          BNE     80$
    515 012432  104017                          EMT     17
    516 012434                          80$:
    517
    521                                 ;;*****************************************************************************
                                        ;*TEST 7         DRIVE TYPE TEST
                                        ;;*****************************************************************************
        012434                          TST7:
        012434  000004                  SCOPE                   ;SCOPE CALL
        012436  000240                  NOP
        012440  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        012444  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        012450  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        012454  012737  000007  001226  MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    522
    523 012462  016002  000026          MOV     RMDT(R0),R2     ;STORE RMDT AT R2
    524 012466  022702  020024          CMP     #SNGPRT,R2      ;SINGLE PORT RM03 ?
    525 012472  001437                  BEQ     10$             ;YES !!
    526 012474  022702  024024          CMP     #DULPRT,R2      ;DUAL PORT RM03 ?
    527 012500  001434                  BEQ     10$             ;YES !!
    528
    529 012502  022702  020025          CMP     #SNGPRT!BIT0,R2 ;SINGLE PORT RM02 ?
    530 012506  001431                  BEQ     10$             ;YES !!
    531 012510  022702  024025          CMP     #DULPRT!BIT0,R2 ;DUAL PORT RM02 ?
    532 012514  001426                  BEQ     10$             ;YES !!
    533
    534 012516  022702  020027          CMP     #SNGPRT!BIT1!BIT0,R2    ;SINGLE PORT RM05 ?
    535 012522  001423                  BEQ     10$             ;YES !!
    536 012524  022702  024027          CMP     #DULPRT!BIT1!BIT0,R2    ;DUAL PORT RM05 ?
    537 012530  001420                  BEQ     10$
    538
    539 012532  010237  001142          MOV     R2,$BDDAT       ;GET RECIEVED DRIVE TYPE
    540 012536  012737  020024  001174  MOV     #SNGPRT,$TMP0   ;GET EXPECTED DRIVE TYPE
    541 012544  012737  024024  001176  MOV     #DULPRT,$TMP1   ;   ''      ''
    542 012552  010037  001136          MOV     R0,$BDADR       ;LOAD BAD ADDRESS
    543 012556  062737  000026  001136  ADD     #RMDT,$BDADR
    544 012564  104057                  EMT     57
    545 012566  000137  053600          JMP     $EOSP           ;GO TO NEXT DEVICE
    546 012572                          10$:
    547
    548                                 ;;*****************************************************************************
                                        ;*TEST 10        DEVICE AVAILABLE TEST
                                        ;;*****************************************************************************
        012572                          TST10:
        012572  000004                  SCOPE                   ;SCOPE CALL
        012574  000240                  NOP
        012576  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        012602  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        012606  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        012612  012737  000010  001226  MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
    549
    550 012620  004737  055614          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
```

```
551 012624 016037 000000 001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
552 012632 042737 173777 001142          BIC     #^CDVA,$BDDAT   ;CLEAR ALL BUT DVA
553 012640 001006                        BNE     10$             ;BRANCH IF DVA SET
554 012642 012737 004000 001140          MOV     #DVA,$GDDAT     ;SETUP EXPECTED
555 012650 010037 001136                 MOV     R0,$BDADR       ;SETUP REG ADDRESS
556 012654 104060                        EMT     60
557 012656                       10$:
558
559                              ;;*****************************************************************
                                 ;*TEST 11        HOLDING REGISTER TRANSFER TEST

                                 ;;*****************************************************************
    012656                       TST11:
    012656 000004                        SCOPE                   ;SCOPE CALL
    012660 000240                        NOP
    012662 012706 001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
    012666 013700 001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
    012672 013701 001462                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
    012676 012737 000011 001226          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
560
561 012704 004737 055614                 JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
562 012710 005003                        CLR     R3              ;CLEAR ERROR FLAGS
563 012712 010037 001136                 MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
564 012716 062737 000036 001136          ADD     #RMHR,$BDADR
565
566                              ;WRITE ONES THEN ZEROS IN RMHR AND CHECK FOR S-A-1 BITS.
567                              ;NOTE THAT IT IS NECESSARY TO WRITE SOME OTHER REGISTER IN
568                              ;ORDER TO WRITE THE HOLDING REGISTER, AND RMDA IS USED FOR THIS
569                              ;PURPOSE.
570 012724 012760 177777 000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
571 012732 012760 000000 000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
572 012740 016037 000036 001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
573 012746 005137 001142                 COM     $BDDAT          ;ANY ERROR??
574 012752 001405                        BEQ     10$             ;NO!!
575 012754 005037 001140                 CLR     $GDDAT          ;LOAD EXPECTED
576 012760 104061                        EMT     61
577 012762 052703 000001                 BIS     #BIT0,R3        ;SET ERROR FLAGS
578
579                              ;WRITE ZEROS THEN ONES IN RMHR AND CHECK FOR S-A-0 BITS.
580 012766                       10$:
581 012766 012760 000000 000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
582 012774 012760 177777 000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
583 013002 016037 000036 001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
584 013010 005137 001142                 COM     $BDDAT          ;RMHR IS COMPLEMENTED WHEN READ
585 013014 012737 177777 001140          MOV     #-1,$GDDAT      ;SETUP EXPECTED
586 013022 023737 001140 001142          CMP     $GDDAT,$BDDAT   ;ANY ERROR??
587 013030 001403                        BEQ     20$             ;NO!!
588 013032 104062                        EMT     62
589 013034 052703 000002                 BIS     #BIT1,R3        ;SET ERROR FLAG
590
591                              ;IF NO PREVIOUS ERRORS, WRITE AND READ SHIFTING ONE BIT PATTERN.
592 013040                       20$:
593 013040 005703                        TST     R3              ;ANY FLAGS SET??
594 013042 001025                        BNE     50$             ;YES!!
595 013044 012702 000001                 MOV     #1,R2           ;R2=DATA PATTERN
596 013050                       30$:
597 013050 012760 000000 000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
```

```
 598 013056  010260  000006                MOV     R2,RMDA(R0)       ;LOAD RMDA
 599 013062  016037  000036  001142        MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
 600 013070  005137  001142                COM     $BDDAT            ;RMHR IS COMPLEMENTED
 601 013074  023702  001142                CMP     $BDDAT,R2         ;ANY ERROR??
 602 013100  001404                        BEQ     40$               ;NO!!
 603 013102  010237  001140                MOV     R2,$GDDAT         ;SETUP EXPECTED
 604 013106  104063                        EMT     63
 605 013110  000402                        BR      50$               ;DO NOT COLLECT ALL ERRORS
 606
 607 013112  006302              40$:      ASL     R2                ;SHIFT TO NEXT PATTERN
 608 013114  001355                        BNE     30$               ;CONTINUE IF NOT DONE
 609
 610 013116                      50$:
 611
 612                             ;;*****************************************************************
                                 ;*TEST 12        CONTROL STATUS #1 TRANSFER TEST

                                 ;;*****************************************************************
     013116                      TST12:
     013116  000004                        SCOPE                     ;SCOPE CALL
     013120  000240                        NOP
     013122  012706  001100                MOV     #STACK,SP         ;LOAD THE STACK POINTER
     013126  013700  001276                MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
     013132  013701  001462                MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
     013136  012737  000012  001226        MOV     #12,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
 613
 614 013144  005003                        CLR     R3                ;R3 = ERROR INDICATOR
 615 013146  004737  055614                JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
 616
 617                             ;WRITE ONES IN RMCS1, BITS 01-05, THEN CLEAR. READ AND
 618                             ;CHECK FOR S-A-1 BITS.
 619 013152  012760  000076  000000        MOV     #ILF76,RMCS1(R0)          ;LOAD RMCS1
 620 013160  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 621 013164  016037  000000  001142        MOV     RMCS1(R0),$BDDAT          ;STORE RMCS1 AT $BDDAT
 622 013172  042737  177701  001142        BIC     #^CILF76,$BDDAT
 623 013200  001410                        BEQ     5$
 624 013202  005037  001140                CLR     $GDDAT
 625 013206  010037  001136                MOV     R0,$BDADR
 626 013212  062737  000000  001136        ADD     #RMCS1,$BDADR
 627 013220  104043                        EMT     43
 628
 629                             ;WRITE ONES IN RMCS1, BITS 01-05, THEN WRITE ZEROS. READ AND CHECK FOR
 630                             ;S-A-1 BITS.
 631 013222                      5$:
 632 013222  012760  000076  000000        MOV     #ILF76,RMCS1(R0)          ;LOAD RMCS1
 633 013230  012760  000000  000000        MOV     #0,RMCS1(R0)    ;LOAD RMCS1
 634 013236  016037  000000  001142        MOV     RMCS1(R0),$BDDAT          ;STORE RMCS1 AT $BDDAT
 635 013244  042737  177701  001142        BIC     #^CILF76,$BDDAT
 636 013252  001412                        BEQ     10$
 637 013254  005037  001140                CLR     $GDDAT
 638 013260  010037  001136                MOV     R0,$BDADR
 639 013264  062737  000000  001136        ADD     #RMCS1,$BDADR
 640 013272  104023                        EMT     23
 641 013274  052703  000001                BIS     #BIT0,R3          ;SET ERROR FLAG
 642
 643                             ;WRITE ZEROS IN RMCS1, THEN ONES, READ AND CHECK S-A-0 BITS.
 644 013300                      10$:
```

```
                                        MOV     #0,RMCS1(R0)    ;LOAD RMCS1
                                        MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
                                        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
                                        BIC     #^CILF76,$BDDAT
                                        MOV     #ILF76,$GDDAT
                                        CMP     $GDDAT,$BDDAT
                                        BEQ     20$
                                        MOV     R0,$BDADR
                                        ADD     #RMCS1,$BDADR
                                        EMT     24
                                        BIS     #BIT1,R3        ;SET ERROR FLAG

                                ;WRITE A SHIFTING ONE BIT PATTERN IN RMCS1, READ AND CHECK FOR STUCK BITS.
                                20$:
                                        TST     R3              ;OMIT IF ANY ERRORS
                                        BNE     50$
                                        MOV     #2,R2           ;R2 = TEST PATTERN
                                30$:
                                        MOV     R2,R3           ;R3 = EXPECTED RESULT, BITS 1-5
                                        BIC     #^CILF76,R3
                                        MOV     #0,RMCS1(R0)    ;LOAD RMCS1
                                        MOV     R2,RMCS1(R0)    ;LOAD RMCS1
                                        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
                                        BIC     #^CILF76,$BDDAT
                                        CMP     R3,$BDDAT
                                        BEQ     40$
                                        MOV     R3,$GDDAT
                                        MOV     R0,$BDADR
                                        ADD     #RMCS1,$BDADR
                                        EMT     25
                                40$:    ASL     R2              ;SHIFT TO NEXT BIT
                                        BNE     30$             ;CONTINUE IF R2 NOT ZERO
                                50$:

                                ;;*******************************************************************
                                ;*TEST 13       ERROR REGISTER #1 TRANSFER TEST

                                ;;*******************************************************************
                                TST13:
                                        SCOPE                   ;SCOPE CALL
                                        NOP
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                                        MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        CLR     R3              ;CLEAR ERROR FLAG

                                ;WRITE ONES IN RMER1, CLEAR AND CHECK FOR S-A-1 BITS
                                        MOV     #-1,RMER1(R0)   ;LOAD RMER1

                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
                                        MOV     RMER1I,$BDDAT
                                        BIC     #^C<PAR!RMR!ILF!ILR>,$BDDAT
                                        BEQ     10$
                                        CLR     $GDDAT
```

```
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104027                          EMT     27
                                                10$:
                        013737  001346  001142          MOV     RMER1I,$BDDAT
                        042737  074017  001142          BIC     #^C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,$BDDAT
                        001410                          BEQ     20$
                        005037  001140                  CLR     $GDDAT
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104030                          EMT     30
                                                20$:
                        013737  001346  001142          MOV     RMER1I,$BDDAT
                        042737  147777  001142          BIC     #^C<OPI!DTE>,$BDDAT
                        001410                          BEQ     30$
                        005037  001140                  CLR     $GDDAT
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104031                          EMT     31

                                                ;WRITE ONES THEN ZEROS IN RMER1, READ AND CHECK FOR S-A-1 BITS
                                                30$:
                        012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
                        012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
                        016037  000014  001346          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
                        013737  001346  001142          MOV     RMER1I,$BDDAT
                        042737  177770  001142          BIC     #^C<RMR!ILF!ILR>,$BDDAT
                        001412                          BEQ     40$
                        005037  001140                  CLR     $GDDAT
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104032                          EMT     32
                        052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
                        013737  001346  001142  40$:     MOV     RMER1I,$BDDAT
                        042737  074017  001142          BIC     #^C<DCK!IAE!AOE!HCRC!HCE!ECH!WCF!FER>,$BDDAT
                        001412                          BEQ     50$
                        005037  001140                  CLR     $GDDAT
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104033                          EMT     33
                        052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
                                                50$:
                        013737  001346  001142          MOV     RMER1I,$BDDAT
                        042737  147777  001142          BIC     #^C<OPI!DTE>,$BDDAT
                        001412                          BEQ     60$
                        005037  001140                  CLR     $GDDAT
                        010037  001136                  MOV     R0,$BDADR
                        062737  000014  001136          ADD     #RMER1,$BDADR
                        104034                          EMT     34
                        052703  030000                  BIS     #BIT,R3

                                                ;WRITE ZEROS THEN ONES IN RMER1,READ AND CHECK FOR S-A-0 BITS
                                                60$:
                        012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
                        012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
                        016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
                        012737  177777  001140          MOV     #-1,$GDDAT
```

```
                                CMP     $GDDAT,$BDDAT
                                BEQ     70$
                                MOV     R0,$BDADR
                                ADD     #RMER1,$BDADR
                                EMT     35
                                BIS     #BIT1,R3

                        ;WRITE A SHIFTING 1 BIT IN RMER1 AND CHECK FOR STUCK BITS
                        ;NOTE:  DONT TEST UNSAFE OR PARITY
                        70$:
                                TST     R3              ;SKIP THIS PART IF ANY ERRORS
                                BNE     120$
                                MOV     #1,R2           ;R2 = TEST PATTERN
                        80$:
                                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                MOV     R2,RMER1(R0)    ;LOAD RMER1
                                MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
                                BIT     #PAR,R2         ;DONT TEST PAR = 0
                                BNE     90$
                                BIC     #PAR,$BDDAT
                        90$:    BIT     #UNS,R2         ;DONT TEST UNS = 0
                                BNE     100$
                                BIC     #UNS,$BDDAT
                        100$:   CMP     R2,$BDDAT
                                BEQ     110$
                                MOV     R2,$GDDAT
                                MOV     R0,$BDADR
                                ADD     #RMER1,$BDADR
                                EMT     36

                        110$:   ASL     R2              ;SHIFT TO NEXT BIT
                                BNE     80$             ;CONTINUE IF R2 NOT ZERO
                        120$:

                        ;;*****************************************************************
                        ;*TEST 14        CLEAR OFFSET STUCK ACTIVE TEST

                        ;;*****************************************************************
                        TST14:
                            .   SCOPE                   ;SCOPE CALL
                                NOP
                                MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                                MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                MOV     #-1,RMOF(R0)    ;LOAD RMOF
                                MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
                                BIC     #^COFD,$BDDAT
                                BNE     10$             ;BRANCH IF OFD IS A ONE
                                MOV     #OFD,$GDDAT     ;SETUP ERROR MESSAGE
                                MOV     R0,$BDADR
                                ADD     #RMOF,$BDADR
                                EMT     172
                        10$:                            ;END OF TEST
```

```
                        ;;*****************************************************************
                        ;*TEST 15       OFFSET REGISTER TRANSFER TEST
                        ;;*****************************************************************
014346                  TST15:
014346  000004                SCOPE                   ;SCOPE CALL
014350  000240                NOP
014352  012706  001100        MOV     #STACK,SP       ;LOAD THE STACK POINTER
014356  013700  001276        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
014362  013701  001462        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
014366  012737  000015  001226 MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
014374  005003                CLR     R3              ;RESET ERROR FLAGS
014376  010037  001136        MOV     R0,$BDADR       ;SETUP BAD ADDRESS
014402  062737  000032  001136 ADD     #RMOF,$BDADR
014410  005037  001140        CLR     $GDDAT          ;SETUP EXPECTED DATA

                        ;WRITE ONES THEN ZEROS IN RMOF AND CHECK FOR S-A-1 BITS.
014414  012760  177777  000032 MOV     #-1,RMOF(R0)    ;LOAD RMOF
014422  012760  000000  000032 MOV     #0,RMOF(R0)     ;LOAD RMOF
014430  016037  000032  001364 MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER

                        ;CHECK UNUSED BITS OF RMOF FOR ZERO
014436  013737  001364  001142 MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
014444  042737  016200  001142 BIC     #^CXNUOF,$BDDAT
014452  001403                BEQ     10$             ;BRANCH IF NO ERROR
014454  104053                EMT     53
014456  052703  000001        BIS     #BIT0,R3        ;SET ERROR FLAG

                        ;CHECK USED BITS OF RMOF FOR ZERO
014462                  10$:
014462  013737  001364  001142 MOV     RMOFI,$BDDAT    ;GET USED BITS
014470  042737  161577  001142 BIC     #XNUOF,$BDDAT
014476  001403                BEQ     20$             ;BRANCH IF NO ERROR
014500  104054                EMT     54
014502  052703  000001        BIS     #BIT0,R3        ;SET ERROR FLAG

                        ;WRITE ZEROS THEN ONES ON RMOF AND CHECK FOR S-A-0 BITS.
014506                  20$:
014506  012760  000000  000032 MOV     #0,RMOF(R0)     ;LOAD RMOF
014514  012760  177777  000032 MOV     #-1,RMOF(R0)    ;LOAD RMOF
014522  016037  000032  001364 MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER

                        ;CHECK UNUSED BITS OF RMOF FOR ZERO.
014530  013737  001364  001142 MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
014536  042737  016200  001142 BIC     #^CXNUOF,$BDDAT
014544  001403                BEQ     30$             ;BRANCH IF NO ERROR
014546  104053                EMT     53
014550  052703  000001        BIS     #BIT0,R3

                        ;CHECK USED BITS OF RMOF FOR ONE.
014554                  30$:
014554  013737  001364  001142 MOV     RMOFI,$BDDAT    ;GET USED BITS
014562  042737  161577  001142 BIC     #XNUOF,$BDDAT
014570  012737  016200  001140 MOV     #^CXNUOF,$GDDAT ;SETUP EXPECTED STATUS
014576  023737  001140  001142 CMP     $GDDAT,$BDDAT
014604  001403                BEQ     40$             ;BRANCH IF NO ERROR
```

```
                                        EMT     55
                                        BIS     #BIT1,R3

                                ;IF NO PREVIOUS ERRORS, TEST RMOF WITH SHIFTING ONE BIT PATTERN.
                                40$:
                                        TST     R3              ;ANY ERROR??
                                        BNE     70$             ;YES!!
                                        MOV     #1,R2           ;STARTING DATA PATTERN
                                50$:
                                        MOV     #0,RMOF(R0)     ;LOAD RMOF
                                        MOV     R2,RMOF(R0)     ;LOAD RMOF
                                        MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
                                        MOV     R2,R3           ;SETUP EXPECTED RESULT
                                        BIC     #XNUOF,R3       ;CLEAR UNUSED BITS
                                        CMP     R3,$BDDAT       ;COMPARE EXPECTED & RECEIVED
                                        BEQ     60$             ;BRANCH IF NO ERROR
                                        MOV     R3,$GDDAT       ;LOAD EXPECTED
                                        EMT     56
                                60$:    ASL     R2              ;SHIFT TO NEXT BIT
                                        BNE     50$             ;CONTINUE IF NOT DONE

                                70$:

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

                                ;;*******************************************************************
                                TST16:
                                        SCOPE                   ;SCOPE CALL
                                        NOP
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                                        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                        CLR     R3              ;RESET ERROR FLAGS
                                        MOV     R0,$BDADR       ;SETUP BAD ADDRESS
                                        ADD     #RMER2,$BDADR
                                        CLR     $GDDAT          ;SETUP EXPECTED DATA
                                        MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT MODE TO ALLOW
                                                                ;"SSE" TO BE SET IN RMER2

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

                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
                                        MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                ;TEST UNUSED BITS FOR ZERO-FAILURE ON IF
                                        MOV     RMER2I,$BDDAT
                                        BIC     #^CXNUER2,$BDDAT
                                        BEQ     10$             ;BRANCH IF NO ERROR
                                        EMT     44
                                        BIS     #EITO,R3        ;SET ERROR FLAG

                                ;TEST "OPE", "IVC", "LSC" FOR ZERO-FAILURE ON CS, IF
```

```
                                        10$:
015020                                          MOV     RMER2I,$BDDAT
015020  013737  001374  001142                  BIC     #^C<OPE!IVC!LSC>,$BDDAT
015026  042737  143777  001142                  BEQ     20$             ;BRANCH IF NO ERROR
015034  001403                                  EMT     45
015036  104045                                  BIS     #BIT0,R3        ;SET ERROR FLAG
015040  052703  000001
                                        ;TEST 'LBC', 'DPE' FOR ZERO-FAILURE ON DS, IF
                                        20$:
015044                                          MOV     RMER2I,$BDDAT
015044  013737  001374  001142                  BIC     #^C<LBC!DPE>,$BDDAT
015052  042737  175767  001142                  BEQ     30$             ;BRANCH IF NO ERROR
015060  001403                                  EMT     46
015062  104046                                  BIS     #BIT0,R3        ;SET ERROR FLAG
015064  052703  000001
                                        ;WRITE ONES IN RMER2 THEN WRITE ZEROS AND CHECK FOR S-A-1 BITS.
                                        30$:
015070                                          MOV     #-1,RMER2(R0)   ;LOAD RMER2
015070  012760  177777  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
015076  012760  000000  000042                  MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
015104  016037  000042  001374
                                        ;TEST 'DPE', 'IVC', 'LSC' FOR ZERO-FAILURE ON CS, IF
015112  013737  001374  001142                  MOV     RMER2I,$BDDAT
015120  042737  143777  001142                  BIC     #^C<OPE!IVC!LSC>,$BDDAT
015126  001403                                  BEQ     40$             ;BRANCH IF NO ERROR
015130  104047                                  EMT     47
015132  052703  000001                          BIS     #BIT0,R3        ;SET ERROR FLAG
                                        ;TEST 'LBC', 'DPE' FOR ZERO-FAILURE ON DS, IF
                                        40$:
015136                                          MOV     RMER2I,$BDDAT
015136  013737  001374  001142                  BIC     #^C<LBC!DPE>,$BDDAT
015144  042737  175767  001142                  BEQ     50$             ;BRANCH IF NO ERROR
015152  001403                                  EMT     50
015154  104050                                  BIS     #BIT0,R3        ;SET ERROR FLAG
015156  052703  000001
                                        ;WRITE ZEROS IN RMER2 THEN WRITE ONES AND CHECK FOR S-A-0 BITS.
                                        50$:
015162                                          MOV     #0,RMER2(R0)    ;LOAD RMER2
015162  012760  000000  000042                  MOV     #-1,RMER2(R0)   ;LOAD RMER2
015170  012760  177777  000042                  MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
015176  016037  000042  001374
                                        ;TEST UNUSED BITS FOR ZERO-FAILURE ON IF
015204  013737  001374  001142                  MOV     RMER2I,$BDDAT
015212  042737  176210  001142                  BIC     #^CXNUER2,$BDDAT
015220  001403                                  BEQ     60$             ;BRANCH IF NO ERROR
015222  104044                                  EMT     44
015224  052703  000001                          BIS     #BIT0,R3        ;SET ERROR FLAG
                                        ;TEST USED BITS FOR ONE-FAILURE ON IF
                                        60$:
015230                                          MOV     RMER2I,$BDDAT
015230  013737  001374  001142                  BIC     #XNUER2,$BDDAT
015236  042737  001567  001142                  MOV     #^CXNUER2,$GDDAT
015244  012737  176210  001140                  CMP     $GDDAT,$BDDAT
015252  023737  001140  001142                  BEQ     70$             ;BRANCH IF NO ERROR
015260  001403                                  EMT     51
015262  104051                                  BIS     #BIT1,R3        ;SET ERROR FLAG
015264  052703  000002
```

```
                                        ;IF NO PREVIOUS ERROR, TEST BIT INTERFERENCE WITH SHIFTING ONE BIT.
015270                                  70$:
015270  005703                          TST     R3              ;ANY ERRORS?
015272  001044                          BNE     120$            ;YES!!
015274  012702  000001                  MOV     #1,R2           ;R2=DATA PATTERN
015300                                  80$:
015300  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
015304  013760  001440  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
015312  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
015316  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
015324  032702  040000                  BIT     #SKI,R2         ;IS SKI BEING SET?
015330  001003                          BNE     90$             ;YES!!
015332  042737  040000  001142          BIC     #SKI,$BDDAT     ;DONT TEST SKI FOR ZERO
015340  032702  000200                  90$:    BIT     #DVC,R2         ;IS DVC BEING SET??
015344  001003                          BNE     100$            ;YES!!
015346  042737  000200  001142          BIC     #DVC,$BDDAT     ;DONT TEST DVC FOR ZERO
015354  010237  001140                  100$:   MOV     R2,$GDDAT
015360  042737  001567  001140          BIC     #XNUER2,$GDDAT  ;UNUSED BITS SHOULD BE ZERO
015366  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;ANY ERRORS??
015374  001401                          BEQ     110$            ;NO!!
015376  104052                          EMT     52

015400  006302                  110$:   ASL     R2              ;SHIFT TO NEXT DATA BIT
015402  001336                          BNE     80$             ;CONTINUE IF NOT DONE

015404                                  120$:

                                        ;;**********************************************************************
                                        ;*TEST 17       SERIAL NUMBER TEST

                                        ;;**********************************************************************
    015404                              TST17:
    015404  000004                      SCOPE                   ;SCOPE CALL
    015406  000240                      NOP
    015410  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
    015414  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
    015420  013701  001462              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
    015424  012737  000017  001226      MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

015432  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
015436  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR TYPEOUT
015442  062737  000030  001136          ADD     #RMSN,$BDADR
015450  012702  000031                  MOV     #25.,R2         ;READ RMSN 25 TIMES

                                        ;READ RMSN AND USE THE RESULT AS EXPECTED VALUE
015454  016037  000030  001140          MOV     RMSN(R0),$GDDAT ;STORE RMSN AT $GDDAT

                                        ;READ RMSN AND COMPARE WITH INITIAL VALUE
015462                                  10$:
015462  016037  000030  001142          MOV     RMSN(R0),$BDDAT ;STORE RMSN AT $BDDAT
015470  023737  001140  001142          CMP     $GDDAT,$BDDAT
015476  001401                          BEQ     20$             ;BRANCH IF SERIAL NUMBER CONSISTENT
015500  104136                          EMT     136

                                        ;DECREMENT COUNT AND CONTINUE IF NOT DONE
015502                                  20$:
```

```
  994 015502  005302                    DEC     R2
  995 015504  100366                    BPL     10$
  996 015506                        30$:                            ;END OF TEST
  997
  998                                   ;;*****************************************************************
                                        ;*TEST 20       CONTROL BUS PARITY DETECTION TEST
                                        ;;*****************************************************************
      015506                            TST20:
      015506  000004                    SCOPE                       ;SCOPE CALL
      015510  000240                    NOP
      015512  012706  001100            MOV     #STACK,SP           ;LOAD THE STACK POINTER
      015516  013700  001276            MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
      015522  013701  001462            MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
      015526  012737  000020  001226    MOV     #20,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
  999
 1000                                   ;SETUP FOR FIRST TEST LOOP (NO ERROR)
 1001 015534  005037  001140            CLR     $GDDAT              ;'PAR' SHOULD BE ZERO
 1002 015540  111137  001416            MOVB    (R1),RMCS20         ;SETUP RMCS2 VALUE
 1003 015544  042737  177770  001416    BIC     #^CUNTMSK,RMCS20
 1004 015552  012737  000001  001444    MOV     #1,RMHR0            ;INITIALIZE DATA PATTERN
 1005 015560  000402                    BR      6$                  ;SKIP INCREMENT FIRST TIME
 1006 015562  006337  001444        5$: ASL     RMHR0               ;SHIFT TO NEXT PATTERN
 1007
 1008                                   ;CLEAR AND VERIFY THAT 'PAR' IS RESET
 1009 015566                        6$:
 1010 015566  004737  055614            JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
 1011 015572  016037  000014  001142    MOV     RMER1(R0),$BDDAT            ;STORE RMER1 AT $BDDAT
 1012 015600  016037  000042  001374    MOV     RMER2(R0),RMER2I            ;STORE RMER2 IN INPUT BUFFER
 1013 015606  042737  177767  001142    BIC     #^CPAR,$BDDAT       ;DID 'PAR' RESET?
 1014 015614  001415                    BEQ     20$                 ;YES!!
 1015 015616  010037  001136            MOV     R0,$BDADR           ;SETUP REGISTER ADDRESS
 1016 015622  062737  000014  001136    ADD     #RMER1,$BDADR
 1017 015630  032737  000010  001374    BIT     #DPE,RMER2I         ;IS 'DPE' SET??
 1018 015636  001002                    BNE     10$                 ;YES!!
 1019 015640  104067                    EMT     67
 1020 015642  000453                    BR      50$
 1021 015644                        10$:
      015644  104070                    EMT     70
 1022 015646  000451                    BR      50$
 1023
 1024                                   ;WRITE TEST PATTERN AND VERIFY 'PAR' STATUS
 1025 015650                        20$:
 1026 015650  013760  001416  000010    MOV     RMCS20,RMCS2(R0)            ;LOAD RMCS2
 1027 015656  013760  001444  000036    MOV     RMHR0,RMHR(R0)      ;LOAD RMHR
 1028 015664  016037  000014  001142    MOV     RMER1(R0),$BDDAT            ;STORE RMER1 AT $BDDAT
 1029 015672  042737  177767  001142    BIC     #^CPAR,$BDDAT
 1030 015700  023737  001140  001142    CMP     $GDDAT,$BDDAT       ;IS 'PAR' CORRECT??
 1031 015706  001410                    BEQ     40$                 ;YES!!
 1032 015710  032737  000020  001416    BIT     #PAT,RMCS20         ;SHOULD 'PAR' BE SET?
 1033 015716  001002                    BNE     30$                 ;YES!!
 1034 015720  104071                    EMT     71
 1035 015722  000423                    BR      50$
 1036 015724                        30$:
      015724  104072                    EMT     72
 1037 015726  000421                    BR      50$                 ;SKIP TO NEXT
 1038                                   ;GO TO NEXT PATTERN
```

```
1039 015730  005737  001444          40$:    TST     RMHRO           ;IS DATA PATTERN COMPLETE??
1040 015734  001312                          BNE     5$              ;NO!!
1041 015736  032737  000020  001416          BIT     #PAT,RMCS20     ;IS TEST COMPLETE??
1042 015744  001012                          BNE     50$             ;YES!!
1043                                ;SETUP FOR SECOND TEST LOOP (ERROR)
1044 015746  052737  000020  001416          BIS     #PAT,RMCS20     ;TURN ON BAD PARITY
1045 015754  012737  000010  001140          MOV     #PAR,$GDDAT     ;EXPECT ERROR
1046 015762  012737  000001  001444          MOV     #1,RMHRO        ;INITIALIZE DATA PATTERN
1047 015770  000676                          BR      6$              ;START LOOP
1048
1049 015772                         50$:                            ;END OF TEST
1050
1051                                ;;*************************************************************************
                                    ;*TEST 21        CONTROL BUS PARITY GENERATION TEST

                                    ;;*************************************************************************
     015772                         TST21:
     015772  000004                         SCOPE                   ;SCOPE CALL
     015774  000240                         NOP
     015776  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
     016002  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     016006  013701  001462                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     016012  012737  000021  001226         MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1052
1053                                ;SETUP FOR TEST (NO 'RROR)
1054 016020  012737  000001  001444          MOV     #1,RMHRO        ;INITIALIZE DATA PATTERN
1055 016026  005037  001140                  CLR     $GDDAT          ;MCPE SHOULD BE ZERO
1056 016032  000402                          BR      20$             ;DONT SHIFT FIRST TIME
1057
1058                                ;SHIFT DATA PATTERN
1059 016034                         10$:
1060 016034  006337  001444                  ASL     RMHRO
1061 016040                         20$:
1062 016040  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1063 016044  013760  001444  000036          MOV     RMHRO,RMHR(R0)  ;LOAD RMHR
1064
1065                                ;TRANSFER DATA TO RH, VERIFY NO 'MCPE'' ERROR
1066 016052  016037  000036  001370          MOV     RMHR(R0),RMHRI  ;STORE RMHR IN INPUT BUFFER
1067 016060  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
1068 016066  042737  157777  001142          BIC     #^CMCPE,$BDDAT  ;WAS BAD PARITY DETECTED??
1069 016074  001402                          BEQ     30$             ;NO!!
1070 016076  104073                          EMT     73
1071 016100  000403                          BR      40$
1072
1073                                ;GO TO NEXT PATTERN
1074 016102                         30$:
1075 016102  005737  001444                  TST     RMHRO           ;DONE ALL PATTERNS??
1076 016106  001352                          BNE     10$             ;NO!!
1077 016110                         40$:                            ;END OF TEST
1078
1079                                ;;*************************************************************************
                                    ;*TEST 22        RMDA,RMDC FAULT TEST

                                    ;;*************************************************************************
     016110                         TST22:
     016110  000004                         SCOPE                   ;SCOPE CALL
     016112  000240                         NOP
```

```
        016114  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        016120  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        016124  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        016130  012737  000022  001226          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  1080
  1081  016136  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  1082
  1083                                  ;WRITE ZEROS, THEN ONES IN RMDA,RMDC-READ AND TEST FOR S-A-0
  1084  016142  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
  1085  016150  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
  1086  016156  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
  1087  016164  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
  1088  016172  016037  000006  001340          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
  1089  016200  016037  000034  001366          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
  1090  016206  022737  177777  001340          CMP     #-1,RMDAI       ;IS ANY REGISTER ALL ONES??
  1091  016214  001410                          BEQ     10$             ;YES!!
  1092  016216  052737  176000  001366          BIS     #XNUDC,RMDCI    ;SET UNUSED BITS
  1093  016224  022737  177777  001366          CMP     #-1,RMDCI
  1094  016232  001401                          BEQ     10$             ;YES!!
  1095  016234  104042                          EMT     42
  1096
  1097  016236                          10$:
  1098
  1099                                  ;;*******************************************************************
                                        ;*TEST 23        DISK ADDRESS TRANSFER TEST
                                        ;;*******************************************************************
        016236                          TST23:
        016236  000004                          SCOPE                   ;SCOPE CALL
        016240  000240                          NOP
        016242  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        016246  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        016252  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        016256  012737  000023  001226          MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  1100
  1101  016264  005003                          CLR     R3              ;R3 = ERROR INDICATOR
  1102  016266  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  1103
  1104                                  ;WRITE ONES IN RMDA, THEN WRITE ZEROS, READ BACK AND CHECK FOR S-A-1 BITS
  1105  016272  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
  1106  016300  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
  1107  016306  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
  1108  016314  005737  001142                  TST     $BDDAT
  1109  016320  001412                          BEQ     10$
  1110  016322  005037  001140                  CLR     $GDDAT
  1111  016326  010037  001136                  MOV     R0,$BDADR
  1112  016332  062737  000006  001136          ADD     #RMDA,$BDADR
  1113  016340  104020                          EMT     20
  1114  016342  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
  1115                                  ;WRITE ZEROS IN RMDA, THEN WRITE ONES, READ BACK AND CHECK FOR S-A-0 BITS
  1116  016346                          10$:
        016346  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
  1117  016354  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
  1118  016362  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
  1119  016370  023727  001142  177777          CMP     $BDDAT,#-1
  1120  016376  001413                          BEQ     20$
  1121  016400  012737  177777  001140          MOV     #-1,$GDDAT
```

```
1122 016406 010037 001136          MOV     R0,$BDADR
1123 016412 062737 000006 001136   ADD     #RMDA,$BDADR
1124 016420 104021                 EMT     21
1125 016422 052703 000002          BIS     #BIT1,R3        ;SET ERROR FLAG
1126                         ;WRITE A SHIFTING 1 BIT PATTERN IN RMDA, READ, AND CHECK FOR STUCK BITS
1127 016426 005703           20$:  TST     R3              ;OMIT BIT TEST IF ANY ERROR
1128 016430 001027                 BNE     50$
1129 016432 012702 000001          MOV     #1,R2           ;R2 = TEST PATTERN
1130 016436                  30$:
     016436 012760 000000 000006   MOV     #0,RMDA(R0)     ;LOAD RMDA
1131 016444 010260 000006          MOV     R2,RMDA(R0)     ;LOAD RMDA
1132 016450 016037 000006 001142   MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1133 016456 023702 001142          CMP     $BDDAT,R2
1134 016462 001410                 BEQ     40$
1135 016464 010237 001140          MOV     R2,$GDDAT
1136 016470 010037 001136          MOV     R0,$BDADR
1137 016474 062737 000006 001136   ADD     #RMDA,$BDADR
1138 016502 104022                 EMT     22
1139 016504 006302           40$:  ASL     R2              ;SHIFT TO NEXT BIT
1140 016506 001353                 BNE     30$             ;CONTINUE IF R2 NOT ZERO
1141 016510                  50$:
1142
1143                         ;;*******************************************************************
                             ;*TEST 24        DESIRED CYLINDER TRANSFER TEST
                             ;;*******************************************************************
     016510                  TST24:
     016510 000004                  SCOPE                   ;SCOPE CALL
     016512 000240                  NOP
     016514 012706 001100           MOV     #STACK,SP       ;LOAD THE STACK POINTER
     016520 013700 001276           MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     016524 013701 001462           MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     016530 012737 000024 001226    MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1144
1145 016536 005003                 CLR     R3              ;RESET ERROR FLAGS
1146 016540 004737 055614          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1147 016544 005037 001140          CLR     $GDDAT          ;LOAD EXPECTED
1148 016550 010037 001136          MOV     R0,$BDADR       ;LOAD REG ADDRESS
1149 016554 062737 000034 001136   ADD     #RMDC,$BDADR
1150
1151                         ;WRITE ONES IN RMDC AND VERIFY THAT UNUSED BITS ARE ZERO
1152 016562 012760 177777 000034   MOV     #-1,RMDC(R0)    ;LOAD RMDC
1153 016570 016037 000034 001142   MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
1154 016576 042737 001777 001142   BIC     #^CXNUDC,$BDDAT         ;CLEAR ALL USED BITS
1155 016604 001403                 BEQ     5$                      ;BRANCH IF NO ERROR
1156 016606 104134                 EMT     134
1157 016610 052703 000001          BIS     #BIT0,R3        ;SET ERROR FLAG
1158
1159                         ;WRITE ONES IN RMDC, THEN WRITE ZEROS, READ AND CHECK FOR S-A-1 BITS
1160 016614                  5$:
1161 016614 012760 177777 000034   MOV     #-1,RMDC(R0)    ;LOAD RMDC
1162 016622 012760 000000 000034   MOV     #0,RMDC(R0)     ;LOAD RMDC
1163 016630 016037 000034 001142   MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
1164 016636 042737 176000 001142   BIC     #XNUDC,$BDDAT   ;CLEAR UNUSED BITS
1165 016644 001403                 BEQ     10$             ;BRANCH IF NO ERROR
1166 016646 104037                 EMT     37
1167 016650 052703 000001          BIS     #BIT0,R3        ;SET ERROR FLAG
```

```
1168
1169                                        ;WRITE ZEROS, THEN ONES IN RMDC, READ AND CHECK FOR S-A-0 BITS
1170 016654                                 10$:
1171 016654  012760  000000  000034              MOV     #0,RMDC(R0)       ;LOAD RMDC
1172 016662  012760  177777  000034              MOV     #-1,RMDC(R0)      ;LOAD RMDC
1173 016670  016037  000034  001142              MOV     RMDC(R0),$BDDAT   ;STORE RMDC AT $BDDAT
1174 016676  052737  176000  001142              BIS     #XNUDC,$BDDAT     ;SET UNUSED BITS
1175 016704  012737  177777  001140              MOV     #-1,$GDDAT        ;LOAD EXPECTED RESULT
1176 016712  023737  001140  001142              CMP     $GDDAT,$BDDAT     ;IS RMDC ALL ONES ??
1177 016720  001403                             BEQ     20$               ;YES !!
1178 016722  104040                             EMT     40
1179 016724  052703  000002                     BIS     #BIT1,R3          ;SET ERROR FLAG
1180
1181                                        ;OMIT BIT TEST IF ANY ERRORS
1182 016730                                 20$:
1183 016730  005703                             TST     R3
1184 016732  001026                             BNE     60$
1185 016734  012702  000001                     MOV     #1,R2             ;R2 = TEST PATTERN
1186
1187                                        ;TEST RMDC WITH SHIFTING ONE BIT
1188 016740                                 30$:
1189 016740  012760  000000  000034              MOV     #0,RMDC(R0)       ;LOAD RMDC
1190 016746  010260  000034                     MOV     R2,RMDC(R0)       ;LOAD RMDC
1191 016752  010203                             MOV     R2,R3             ;R3 = EXPECTED RESULT
1192 016754  042703  176000                     BIC     #XNUDC,R3         ;CLEAR ANY UNUSED BITS
1193 016760  016037  000034  001142              MOV     RMDC(R0),$BDDAT   ;STORE RMDC AT $BDDAT
1194 016766  023703  001142                     CMP     $BDDAT,R3
1195 016772  001404                             BEQ     50$
1196 016774  010337  001140                     MOV     R3,$GDDAT
1197 017000  104041                             EMT     41
1198 017002  000402                             BR      60$               ;SKIP TO NEXT
1199
1200 017004  006302                        50$: ASL     R2                ;SHIFT TO NEXT BIT
1201 017006  001354                             BNE     30$               ;CONTINUE IF R2 NOT ZERO
1202
1203 017010                                 60$:
1204
1205                                        ;;*****************************************************************
                                            ;*TEST 25       ILLEGAL REGISTER TEST

                                            ;;*****************************************************************
     017010                                 TST25:
     017010  000004                             SCOPE                     ;SCOPE CALL
     017012  000240                             NOP
     017014  012706  001100                     MOV     #STACK,SP         ;LOAD THE STACK POINTER
     017020  013700  001276                     MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
     017024  013701  001462                     MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
     017030  012737  000025  001226              MOV     #25,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX
1206
1207 017036  005037  001140                     CLR     $GDDAT            ;''ILR'' SHOULD BE ZERO
1208 017042  012702  000000                     MOV     #0,R2             ;R2=REGISTER INDEX
1209
1210                                        ;CLEAR AND VERIFY THAT ''ILR'' STATUS IS ZERO
1211 017046                                 10$:
1212 017046  004737  055614                     JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
1213 017052  016037  000014  001142              MOV     RMER1(R0),$BDDAT          ;STORE RMER1 AT $BDDAT
1214 017060  042737  177775  001142              BIC     #^CILR,$BDDAT
```

```
1215 017066  001411                        BEQ     20$             ;BRANCH IF ILR IS RESET
1216 017070  005037  001140                CLR     $GDDAT          ;SETUP GOOD DATA, REG ADR
1217 017074  010037  001136                MOV     R0,$BDADR
1218 017100  062737  000014  001136        ADD     #RMER1,$BDADR
1219 017106  104064                        EMT     64
1220 017110  000550                        BR      140$
1221
1222                                ;WRITE THE REGISTER (INDEX=R2) AND TEST ILR STATUS
1223 017112                         20$:
1224 017112  010003                        MOV     R0,R3           ;R3=REG ADDRESS
1225 017114  060203                        ADD     R2,R3
1226 017116  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
1227 017122  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
1228 017126  012737  017170  000004        MOV     #40$,ERRVEC     ;SETUP FOR BUS TIMEOUT
1229 017134  012737  000300  000006        MOV     #PR6,ERRVEC+2
1230 017142  005004                        CLR     R4              ;R4=REGISTER VALUE
1231 017144  022702  000010                CMP     #RMCS2,R2
1232 017150  001001                        BNE     30$
1233 017152  111104                        MOVB    (R1),R4         ;SELECT DRIVE IF RMCS2
1234 017154  010413                 30$:   MOV     R4,(R3)         ;WRITE TEST REGISTER
1235 017156  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1236 017162  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1237 017166  000416                        BR      60$
1238 017170  012716  017176         40$:   MOV     #45$,(SP)       ;DUMMY RTI ADDRESS
1239 017174  000002                        RTI                     ;RESTORE PRIORITY
1240 017176                         45$:
     017176  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1241 017202  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1242 017206  020227  000046                CMP     R2,#RMEC2       ;WERE ALL REGISTERS READ??
1243 017212  101003                        BHI     50$             ;YES!!
1244 017214  010337  001136                MOV     R3,$BDADR
1245 017220  104074                        EMT     74
1246 017222  000503                 50$:   BR      140$
1247
1248 017224                         60$:
     017224  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
1249 017232  042737  177775  001142        BIC     #^CILR,$BDDAT
1250 017240  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;IS ''ILR'' OK??
1251 017246  001411                        BEQ     80$             ;YES!!
1252 017250  010337  001174                MOV     R3,$TMP0        ;SAVE ADDRESS
1253 017254  032737  000002  001140        BIT     #ILR,$GDDAT     ;SHOULD ''ILR'' BE SET??
1254 017262  001002                        BNE     70$             ;YES!!
1255 017264  104065                        EMT     65
1256 017266  000401                        BR      80$
1257 017270                         70$:
     017270  104066                        EMT     66
1258
1259                                ;ADVANCE TO THE NEXT REGISTER ADDRESS
1260 017272                         80$:
1261 017272  062702  000002                ADD     #2,R2           ;INCREMENT INDEX
1262 017276  022702  000050                CMP     #50,R2          ;TIME TO TRY RH70 ?
1263 017302  101261                        BHI     10$             ;BRANCH IF NOT
1264 017304  103437                        BLO     110$            ;BRANCH IF ALREADY CHECKED
1265                                ;       CMP     #RMBAE,R2       ;IS THIS RMBAE??
1266                                ;       BNE     100$            ;NO!!
1267 017306  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
1268 017312  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
```

```
 1269 017316  012737  017414  000004          MOV     #130$,ERRVEC    ;SETUP FOR TIMEOUT
 1270 017324  012737  000300  000006          MOV     #PR6,ERRVEC+2
 1271                                 ;        PUTCS1  #A17!A16
 1272                                 ;        GETBAE
 1273                                 ;        BIC     #^C<BIT1!BIT0>,RMBAEI
 1274                                 ;        BNE     90$             ;BRANCH IF RH70 CONTROLLER
 1275                                 ;        BIS     #ILR,$GDDAT     ;''ILR'' SHOULD BE SET
 1276                                 ;90$:    POP     ERRVEC+2
 1277                                 ;        POP     ERRVEC
 1278                                 ;100$:   CMP     #RMCS3+2,R2     ;DONE ALL LEGAL REGISTERS
 1279                                 ;        BNE     110$
 1280                                 ;        BIS     #ILR,$GDDAT     ;''ILR'' SHOULD BE SET
 1281 017332  052737  000002  001140          BIS     #ILR,$GDDAT     ;SET ILR
 1282 017340  012702  000054                  MOV     #54,R2          ;START AT INDEX 54 IF RH70 WITH 22 REG
 1283 017344  012760  001400  000000          MOV     #A17!A16,RMCS1(R0)      ;SET EXTEND BITS
 1284 017352  016037  000050  001402          MOV     50(R0),RMBAEI   ;READ THE EXTENDED BITS
 1285 017360  042737  177774  001402          BIC     #177774,RMBAEI  ;CHOP OFF
 1286 017366  001002                          BNE     90$             ;BRANCH IF RH70 WITH 22 REG
 1287 017370  012702  000050                  MOV     #50,R2          ;OTHERWISE NOT A RH70 OR RH70 WITH 32 REG
 1288 017374  012637  000006          90$:    MOV     (SP)+,ERRVEC+2  ;
 1289 017400  012637  000004                  MOV     (SP)+,ERRVEC    ;
 1290 017404  022702  000074          110$:   CMP     #74,R2          ;DONE ALL TESTS
 1291 017410  101410                          BLOS    140$            ;YES!!
 1292 017412  000615                  120$:   BR      10$
 1293
 1294 017414  012716  017422          130$:   MOV     #135$,(SP)      ;DUMMY RTI ADDRESS
 1295 017420  000002                          RTI                     ;RESTORE PRIORITY
 1296 017422                          135$:
      017422  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 1297 017426  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 1298 017432                          140$:                           ;END OF TEST
 1299
 1300                                 ;;*****************************************************************
                                      ;*TEST 26        RESET GO BY INIT TEST
                                      ;;*****************************************************************
      017432                          TST26:
      017432  000004                          SCOPE                   ;SCOPE CALL
      017434  000240                          NOP
      017436  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      017442  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      017446  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      017452  012737  000026  001226          MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1301
 1302 017460  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 1303 017464  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR MSG
 1304
 1305                                 ;SET GO, INITIALIZE AND VERIFY THAT GO IS RESET
 1306 017470  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
 1307
 1308 017476  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 1309 017502  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 1310 017510  042737  177776  001142          BIC     #^CGO,$BDDAT
 1311 017516  001403                          BEQ     10$             ;BRANCH IF GO IS RESET
 1312 017520  005037  001140                  CLR     $GDDAT
 1313 017524  104137                          EMT     137
 1314
```

```
1315 017526                                  10$:                            ;END OF TEST
1316
1317                                         ;;*****************************************************************
                                             ;*TEST 27       DIAGNOSTIC MODE TEST
                                             ;;*****************************************************************
     017526                                  TST27:
     017526  000004                                 SCOPE                   ;SCOPE CALL
     017530  000240                                 NOP
     017532  012706  001100                         MOV     #STACK,SP       ;LOAD THE STACK POINTER
     017536  013700  001276                         MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     017542  013701  001462                         MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     017546  012737  000027  001226                 MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1318
1319 017554  010037  001136                         MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1320 017560  062737  000024  001136                 ADD     #RMMR1,$BDADR
1321 017566  005003                                 CLR     R3              ;INITIALIZE ERROR FLAGS
1322
1323                                         ;INITIALIZE AND VERIFY THAT 'DMD' IS RESET
1324 017570  004737  055614                         JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1325 017574  016037  000024  001142                 MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1326 017602  042737  177776  001142                 BIC     #^CDMD,$BDDAT
1327 017610  001403                                 BEQ     10$             ;BRANCH IF 'DMD' IS ZERO
1328 017612  005037  001140                         CLR     $GDDAT
1329 017616  104075                                 EMT     75
1330
1331                                         ;SET AND RESET 'DMD' USING REGISTER TRANSFER-VERIFY 'DMD' NOT S-A-1
1332 017620                                  10$:
1333 017620  012760  000001  000024                 MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1334 017626  012760  000000  000024                 MOV     #0,RMMR1(R0)    ;LOAD RMMR1
1335 017634  016037  000024  001142                 MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1336 017642  042737  177776  001142                 BIC     #^CDMD,$BDDAT
1337 017650  001405                                 BEQ     20$             ;BRANCH IF DMD NOT S-A-1
1338 017652  005037  001140                         CLR     $GDDAT
1339 017656  104076                                 EMT     76
1340 017660  052703  000001                         BIS     #BIT0,R3        ;SET ERROR FLAG
1341
1342                                         ;RESET AND SET 'DMD' USING REGISTER TRANSFER-VERIFY 'DMD' NOT S-A-0
1343 017664                                  20$:
1344 017664  012760  000000  000024                 MOV     #0,RMMR1(R0)    ;LOAD RMMR1
1345 017672  012760  000001  000024                 MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1346 017700  016037  000024  001142                 MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1347 017706  042737  177776  001142                 BIC     #^CDMD,$BDDAT
1348 017714  001006                                 BNE     30$             ;BRANCH IF DMD NOT S-A-0
1349 017716  012737  000001  001140                 MOV     #DMD,$GDDAT
1350 017724  104077                                 EMT     77
1351 017726  052703  000002                         BIS     #BIT1,R3        ;SET ERROR FLAG
1352
1353                                         ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE WITH SHIFTING
1354                                         ;ONE BIT PATTERN
1355 017732                                  30$:
1356 017732  005703                                 TST     R3              ;ANY ERRORS DETECTED??
1357 017734  001027                                 BNE     60$             ;YES!!
1358 017736  012702  000002                         MOV     #BIT1,R2        ;INITAILIZE DATA PATTERN
1359 017742                                  40$:
     017742  012760  000000  000024                 MOV     #0,RMMR1(R0)    ;LOAD RMMR1
1360 017750  010260  000024                         MOV     R2,RMMR1(R0)    ;LOAD RMMR1
```

```
 1361 017754  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 1362 017762  042737  177776  001142          BIC     #^CDMD,$BDDAT
 1363 017770  001407                          BEQ     50$             ;BRANCH IF DMD NOT SET
 1364 017772  010237  001174                  MOV     R2,$TMP0        ;SAVE DATA
 1365 017776  010237  001174                  MOV     R2,$TMP0        ;SAVE DATA
 1366 020002  005037  001140                  CLR     $GDDAT          ;DMD SHOULD BE ZERO
 1367 020006  104100                          EMT     100
 1368
 1369                                 ;SHIFT TO NEXT DATA BIT AND CONTINUE TES  IF NOT DONE
 1370 020010                          50$:
 1371 020010  006302                          ASL     R2
 1372 020012  001353                          BNE     40$
 1373 020014                          60$:                            ;END OF TEST
 1374
 1375                                 ;;*****************************************************************
                                      ;*TEST 30        MOL TEST
                                      ;;*****************************************************************
      020014                          TST30:
      020014  000004                          SCOPE                   ;SCOPE CALL
      020016  000240                          NOP
      020020  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      020024  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      020030  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      020034  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1376
 1377 020042  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 1378 020046  010037  001136                  MOV     R0,$BDADR       ;R0=REGISTER ADDRESS
 1379 020052  062737  000012  001136          ADD     #RMDS,$BDADR
 1380 020060  005003                          CLR     R3              ;R3=ERROR FLAG
 1381
 1382                                 ;SET DIAGNOSTIC MODE AND VERIFY THAT 'MOL'' IS ZERO
 1383 020062  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 1384 020070  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 1385 020076  042737  167777  001142          BIC     #^CMOL,$BDDAT
 1386 020104  001405                          BEQ     10$
 1387 020106  005037  001140                  CLR     $GDDAT
 1388 020112  104101                          EMT     101
 1389 020114  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
 1390
 1391                                 ;SET MAINTENANCE UNIT READY AND VERIFY THAT 'MOL'' IS ONE
 1392 020120                          10$:
 1393 020120  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 1394 020126  012760  001001  000024          MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
 1395 020134  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 1396 020142  042737  167777  001142          BIC     #^CMOL,$BDDAT
 1397 020150  001006                          BNE     20$
 1398 020152  012737  010000  001140          MOV     #MOL,$GDDAT
 1399 020160  104102                          EMT     102
 1400 020162  052703  000002                  BIS     #BIT1,R3
 1401
 1402                                 ;IF NO PREVIOUS ERROR, VERIFY VOLUME VALID IS RESET AND
 1403                                 ;TEST FOR BIT INTERFERENCE
 1404 020166                          20$:
 1405 020166  005703                          TST     R3              ;ANY ERROR DETECTED??
 1406 020170  001057                          BNE     70$             ;YES!!
 1407 020172  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
```

```
1408 020200  042737  177677  001142          BIC     #^CVV,$BDDAT
1409 020206  001403                          BEQ     25$             ;BRANCH IF VV RESET
1410 020210  005037  001140                  CLR     $GDDAT
1411 020214  104135                          EMT     135
1412 020216                          25$:
1413 020216  012702  000001                  MOV     #1,R2           ;INITIALIZE DATA PATTERN
1414 020222                          30$:
     020222  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1415 020230  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1416 020234  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1417 020242  042737  167777  001142          BIC     #^CMOL,$BDDAT
1418 020250  005003                          CLR     R3              ;SETUP EXPECTED 'MOL'
1419 020252  032702  001000                  BIT     #MUR,R2
1420 020256  001402                          BEQ     40$
1421 020260  052703  010000                  BIS     #MOL,R3         ;'MOL' SHOULD BE ONE
1422 020264  020337  001142          40$:    CMP     R3,$BDDAT       ;IS MOL OK??
1423 020270  001405                          BEQ     50$             ;YES!!
1424 020272  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1425 020276  010337  001140                  MOV     R3,$GDDAT
1426 020302  104103                          EMT     103
1427
1428                                 ;SHIFT TO NEXT PATTERN
1429 020304                          50$:
1430 020304  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
1431 020310  001002                          BNE     60$
1432 020312  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
1433 020316  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
1434 020320  001403                          BEQ     70$             ;BRANCH IF DONE
1435 020322  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1436 020326  000735                          BR      30$             ;CONTINUE
1437
1438 020330                          70$:                            ;END OF TEST
1439
1440                                 ;;*****************************************************************
                                     ;*TEST 31        WRITE LOCK TEST
                                     ;;*****************************************************************
     020330                          TST31:
     020330  000004                          SCOPE                   ;SCOPE CALL
     020332  000240                          NOP
     020334  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     020340  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     020344  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     020350  012737  000031  001226          MOV     #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1441
1442 020356  005003                          CLR     R3              ;R3=ERROR FLAG
1443 020360  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1444 020364  062737  000012  001136          ADD     #RMDS,$BDADR
1445 020372  005037  001140                  CLR     $GDDAT          ;WRL SHOULD BE ZERO
1446 020376  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1447
1448                                 ;SET DIAGNOSTIC MODE AND VERIFY 'WRL' IS ZERO
1449 020402  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1450 020410  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1451 020416  042737  173777  001142          BIC     #^CWRL,$BDDAT
1452 020424  001403                          BEQ     10$             ;BRANCH IF WRL IS ZERO
1453 020426  104104                          EMT     104
```

```
1454 020430  052703  000001          BIS     #BIT0,R3        ;SET ERROR FLAG
1455
1456                                ;SET MAINTENANCE WRITE PROTECT AND VERIFY 'WRL' IS ONE
1457 020434                         10$:
1458 020434  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1459 020442  012760  000011  000024        MOV     #DMD!MWP,RMMR1(R0)      ;LOAD RMMR1
1460 020450  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1461 020456  042737  173777  001142        BIC     #^CWRL,$BDDAT
1462 020464  001006                        BNE     20$             ;BRANCH IF WRL IS NOE
1463 020466  012737  004000  001140        MOV     #WRL,$GDDAT     ;WRL SHOULD BE SET
1464 020474  104105                        EMT     105
1465 020476  052703  000002          BIS     #BIT1,R3        ;SET ERROR FLAG
1466
1467                                ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE ON 'MWP'
1468 020502                         20$:
1469 020502  005703                        TST     R3              ;ANY OTHER ERROR??
1470 020504  001045                        BNE     70$             ;YES!!
1471 020506  012702  000001          MOV     #1,R2           ;INITIALIZE DATA PATTERN
1472
1473                                ;TRANSFER DATA TO RMMR1, READ RMDS AND VERIFY WRL
1474 020512                         30$:
1475 020512  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1476 020520  010260  000024                MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1477 020524  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1478 020532  042737  173777  001142        BIC     #^CWRL,$BDDAT   ;CLEARUP RECEIVED 'WRL'
1479 020540  005003                        CLR     R3              ;GENERATE EXPECTED 'WRL'
1480 020542  032702  000010                BIT     #MWP,R2
1481 020546  001402                        BEQ     40$
1482 020550  052703  004000                BIS     #WRL,R3         ;WRL SHOULD BE SET
1483 020554  020337  001142        40$:    CMP     R3,$BDDAT       ;IS WRL OK??
1484 020560  001405                        BEQ     50$             ;YES!!
1485 020562  010337  001140                MOV     R3,$GDDAT       ;SAVE EXPECTED
1486 020566  010237  001174                MOV     R2,$TMP0        ;SAVE DATA PATTERN
1487 020572  104106                        EMT     106
1488
1489                                ;ADVANCE TO NEXT DATA BIT
1490 020574                         50$:
1491 020574  042702  000001                BIC     #DMD,R2         ;DONT SHIFT DMD
1492 020600  001002                        BNE     60$
1493 020602  012702  000001                MOV     #DMD,R2         ;DONT TRUNCATE TEST
1494 020606  006302                60$:    ASL     R2              ;SHIFT DATA BIT
1495 020610  001403                        BEQ     70$             ;EXIT IF DONE
1496 020612  052702  000001                BIS     #DMD,R2         ;KEEP DIAGNOSTIC MODE ON
1497 020616  000735                        BR      30$             ;CONTINUE TEST
1498
1499 020620                         70$:                            ;END OF TEST
1500
1501                                ;;*****************************************************************
                                    ;*TEST 32        DRIVE FAULT TEST

                                    ;;*****************************************************************
      020620                        TST32:
      020620  000004                        SCOPE                   ;SCOPE CALL
      020622  000240                        NOP
      020624  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
      020630  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      020634  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
```

```
020640  012737  000032  001226          MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1502
1503 020646  004737  055614              JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1504 020652  005003                      CLR     R3              ;INITIALIZE ERROR FLAGS
1505 020654  010037  001136              MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1506 020660  062737  000042  001136      ADD     #RMER2,$BDADR
1507 020666  005037  001140              CLR     $GDDAT          ;'DVC' AND 'UNS' SHOULD BE ZERO
1508
1509                                 ;SET AND RESET MAINTENANCE DRIVE FAULT, VERIFY THAT 'DVC' IS NOT
1510                                 ;STUCK-AT-ONE.
1511 020672  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1512 020700  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1513 020706  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
1514 020714  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1515 020722  042737  177577  001142          BIC     #^CDVC,$BDDAT   ;IS 'DVC' RESET??
1516 020730  001406                          BEQ     10$             ;YES!!
1517 020732  104107                          EMT     107
1518 020734  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
1519 020740  012737  040000  001140          MOV     #UNS,$GDDAT
1520
1521                                 ;VERIFY THAT 'UNS' IS SAME AS 'DVC'
1522 020746                          10$:
1523 020746  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
1524 020754  042737  137777  001142          BIC     #^CUNS,$BDDAT
1525 020762  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS 'UNS' OK??
1526 020770  001414                          BEQ     30$             ;YES!!
1527 020772  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1528 020776  062737  000014  001136          ADD     #RMER1,$BDADR
1529 021004  032737  040000  001140          BIT     #UNS,$GDDAT     ;SHOULD 'UNS' BE ON??
1530 021012  001002                          BNE     20$             ;YES!!
1531 021014  104110                          EMT     110
1532 021016  000401                          BR      30$
1533 021020                          20$:
1534 021020  104111                          EMT     111
1535
1536                                 ;RESET AND SET 'MDF', VERIFY THAT 'DVC' IS NOT S-A-0.
1537 021022                          30$:
1538 021022  012737  000200  001140          MOV     #DVC,$GDDAT     ;DVC SHOULD BE ON
1539 021030  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1540 021036  012760  000101  000024          MOV     #DMD!MDF,RMMR1(R0)      ;LOAD RMMR1
1541 021044  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1542 021052  042737  177577  001142          BIC     #^CDVC,$BDDAT
1543 021060  001012                          BNE     40$             ;BRANCH IF DVC IS SET
1544 021062  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1545 021066  062737  000042  001136          ADD     #RMER2 $BDADR
1546 021074  104112                          EMT     112
1547 021076  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
1548 021102  005037  001140                  CLR     $GDDAT          ;UNS SHOULD BE OFF
1549
1550                                 ;VERIFY THAT 'UNS' IS SAME AS 'DVC'
1551 021106                          40$:
1552 021106  005737  001140                  TST     $GDDAT          ;CHANGE DVC TO UNS
1553 021112  001403                          BEQ     50$
1554 021114  012737  040000  001140          MOV     #UNS,$GDDAT
1555 021122                          50$:
1556 021122  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 557 021130  042737  137777  001142          BIC     #^CUNS,$BDDAT
```

```
1558 021136  023737  001140  001142          CMP     $GDDAT,$BDDAT
1559 021144  001414                          BEQ     70$             ;BRANCH IF UNS IS OK
1560 021146  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1561 021152  062737  000014  001136          ADD     #RMER1,$BDADR
1562 021160  032737  040000  001140          BIT     #UNS,$GDDAT     ;SHOULD UNS BE ON??
1563 021166  001002                          BNE     60$             ;YES!!
1564 021170  104113                          EMT     113
1565 021172  000401                          BR      70$
1566 021174                          60$:
1567 021174  104114                          EMT     114
1568
1569                                 ;IF THERE WERE NO PREVIOUS ERRORS, TEST FOR BIT INTERFERENCE ON
1570                                 ;'MDF'
1571 021176                          70$:
1572 021176  005703                          TST     R3
1573 021200  001056                          BNE     120$            ;BRANCH IF ANY OTHER ERRORS
1574 021202  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1575 021206                          80$:
     021206  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1576 021214  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1577 021222  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1578 021226  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1579 021234  042737  177577  001142          BIC     #^CDVC,$BDDAT   ;GET RESULTS
1580 021242  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED
1581 021246  032702  000100                  BIT     #MDF,R2         ;WAS MDF SET??
1582 021252  001403                          BEQ     90$             ;NO!!
1583 021254  052737  000200  001140          BIS     #DVC,$GDDAT     ;YES-DVC SHOULD BE ON
1584 021262  023737  001140  001142  90$:    CMP     $GDDAT,$BDDAT
1585 021270  001410                          BEQ     100$            ;BRANCH IF DVC IS OK
1586 021272  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1587 021276  062737  000042  001136          ADD     #RMER2,$BDADR
1588 021304  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1589 021310  104115                          EMT     115
1590
1591                                 ;SHIFT TO NEXT BIT POSITION
1592 021312                          100$:
1593 021312  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
1594 021316  001002                          BNE     110$
1595 021320  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
1596 021324  006302                  110$:   ASL     R2              ;SHIFT
1597 021326  001403                          BEQ     120$            ;EXIT IF DONE
1598 021330  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1599 021334  000724                          BR      80$             ;CONTINUE
1600
1601 021336                          120$:                           ;END OF TEST
1602
1603                                 ;;*****************************************************************
                                     ;*TEST 33        SEEK ERROR TEST

                                     ;;*****************************************************************
     021336                          TST33:
     021336  000004                          SCOPE                   ;SCOPE CALL
     021340  000240                          NOP
     021342  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     021346  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     021352  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     021356  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
1604
1605 021364  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1606 021370  005003                          CLR     R3              ;CLEAR ERROR FLAGS
1607 021372  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1608 021376  062737  000042  001136          ADD     #RMER2,$BDADR
1609
1610                                 ;SET DIAGNOSTIC MODE AND VERIFY THAT "SKI" CAN BE RESET
1611 021404  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1612 021412  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1613 021420  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1614 021426  042737  137777  001142          BIC     #^CSKI,$BDDAT
1615 021434  001405                          BEQ     10$             ;BRANCH IF SKI IS RESET
1616 021436  005037  001140                  CLR     $GDDAT          ;SKI SHOULD BE ZERO
1617 021442  104116                          EMT     116
1618 021444  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
1619
1620                                 ;SET MAINTENANCE SEEK ERROR AND VERIFY THAT "SKI" CAN BE SET
1621 021450                          10$:
1622 021450  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1623 021456  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1624 021464  012760  000201  000024          MOV     #DMD!MSER,RMMR1(R0)     ;LOAD RMMR1
1625 021472  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1626 021500  042737  137777  001142          BIC     #^CSKI,$BDDAT
1627 021506  001005                          BNE     20$             ;BRANCH IF SKI IS SET
1628 021510  012737  040000  001140          MOV     #SKI,$GDDAT     ;CANT SET SKI
1629 021516  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
1630
1631                                 ;IF NO PREVIOUS ERROR, CHECK FOR BIT INTERFERENCE SETTING MAINTENANCE
1632                                 ;SEEK ERROR.
1633 021522                          20$:
1634 021522  005703                          TST     R3
1635 021524  001051                          BNE     70$             ;BRANCH IF ANY OTHER ERRORS
1636 021526  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1637 021532                          30$:
1638 021532  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1639 021540  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1640 021546  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1641 021552  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1642 021560  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;GET SKI STATUS
1643 021566  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED RESULT
1644 021572  032702  000200                  BIT     #MSER,R2
1645 021576  001403                          BEQ     40$
1646 021600  052737  040000  001140          BIS     #SKI,$GDDAT     ;SKI SHOULD BE ON
1647 021606  023737  001140  001142  40$:    CMP     $GDDAT,$BDDAT
1648 021614  001403                          BEQ     50$             ;BRANCH IF SKI IS OK
1649 021616  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1650 021622  104120                          EMT     120
1651
1652                                 ;ADVANCE TEST PATTERN IN R2
1653 021624                          50$:
1654 021624  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD BIT
1655 021630  001002                          BNE     60$
1656 021632  012702  000001                  MOV     #DMD,R2         ;DON, TRUNCATE TEST
1657 021636  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
1658 021640  001403                          BEQ     70$             ;EXIT IF DONE
1659 021642  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1660 021646  000731                          BR      30$             ;CONTINUE TEST
```

```
                                        70$:                            ;END OF TEST
                                        ;;*****************************************************************
                                        ;*TEST 34        PIP TEST
                                        ;;*****************************************************************
021650                                  TST34:
021650  000004                                SCOPE                       ;SCOPE CALL
021652  000240                                NOP
021654  012706  001100                        MOV     #STACK,SP           ;LOAD THE STACK POINTER
021660  013700  001276                        MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
021664  013701  001462                        MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
021670  012737  000034  001226                MOV     #34,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
021676  004737  055614                        JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
021702  005003                              CLR     R3                  ;RESET ERROR FLAGS
021704  010037  001136                      MOV     R0,$BDADR           ;SETUP REGISTER ADDRESS
021710  062737  000012  001136              ADD     #RMDS,$BDADR
                                        ;SET MAINTENANCE ON CYLINDER 'MOC'' AND VERIFY THAT 'PIP'' CAN BE RESET.
021716  012760  000001  000024              MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
021724  012760  000401  000024              MOV     #DMD!MOC,RMMR1(R0)      ;LOAD RMMR1
021732  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
021740  042737  157777  001142              BIC     #^CPIP,$BDDAT
021746  001405                              BEQ     10$                 ;BRANCH IF PIP IS RESET
021750  005037  001140                      CLR     $GDDAT
021754  104121                              EMT     121
021756  052703  000001                      BIS     #BIT0,R3            ;SET ERROR FLAG
                                        ;RESET MAINTENANCE ON CYLINDER AND VERIFY THAT 'PIP'' CAN BE SET
021762                                  10$:
021762  012760  000001  000024              MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
021770  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
021776  042737  157777  001142              BIC     #^CPIP,$BDDAT
022004  001006                              BNE     20$                 ;BRANCH IF PIP IS SET
022006  012737  020000  001140              MOV     #PIP,$GDDAT
022014  104122                              EMT     122
022016  052703  000002                      BIS     #BIT1,R3            ;SET ERROR FLAG
                                        ;IF NO PREVIOUS ERROR, TEST FOR ADJACENT BIT SETTING 'MOC''
022022                                  20$:
022022  005703                              TST     R3
022024  001046                              BNE     70$                 ;BRANCH IF ANY PREVIOUS ERROR
022026  012702  000001                      MOV     #1,R2               ;INITIALIZE TEST PATTERN
                                        ;WRITE THE TEST PATTERN, CHECK MOC USING PIP
022032                                  30$:
022032  012760  000001  000024              MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
022040  010260  000024                      MOV     R2,RMMR1(R0)    ;LOAD RMMR1
022044  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
022052  042737  157777  001142              BIC     #^CPIP,$BDDAT   ;GET PIP STATUS
022060  005037  001140                      CLR     $GDDAT          ;SETUP EXPECTED RESULT
022064  032702  000400                      BIT     #MOC,R2
022070  001003                              BNE     40$
022072  052737  020000  001140              BIS     #PIP,$GDDAT     ;PIP SHOULD BE SET
022100  023737  001140  001142  40$:       CMP     $GDDAT,$BDDAT
```

```
1708 022106  001403                        BEQ     50$             ;BRANCH IF PIP OK
1709 022110  010237  001174                MOV     R2,$TMP0        ;SAVE TEST PATTERN
1710 022114  104123                        EMT     123
1711
1712                                ;ADVANCE THE TEST PATTERN
1713 022116                         50$:
1714 022116  042702  000001                BIC     #DMD,R2         ;DONT SHIFT DMD
1715 022122  001002                        BNE     60$
1716 022124  012702  000001                MOV     #DMD,R2         ;DONT TRUNCATE TEST
1717 022130  006302                 60$:   ASL     R2              ;SHIFT BIT
1718 022132  001403                        BEQ     70$             ;EXIT IF DONE
1719 022134  052702  000001                BIS     #DMD,R2         ;KEEP DMD ON
1720 022140  000734                        BR      30$             ;CONTINUE TEST
1721
1722 022142                         70$:                            ;END OF TEST
1723
1724                                ;;*******************************************************************
                                    ;*TEST 35        EBL TEST
                                    ;;*******************************************************************
     022142                         TST35:
     022142  000004                        SCOPE                   ;SCOPE CALL
     022144  000240                        NOP
     022146  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     022152  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     022156  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     022162  012737  000035  001226        MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1725
1726 022170  005003                        CLR     R3              ;RESET ERROR FLAGS
1727 022172  010037  001136                MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1728 022176  062737  000024  001136        ADD     #RMMR1,$BDADR
1729 022204  005037  001140                CLR     $GDDAT          ;SETUP EXPECTED RESULT
1730
1731                                ;CLEAR AND VERIFY THAT END OF BLOCK IS RESET
1732 022210  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1733 022214  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1734 022222  042737  157777  001142        BIC     #^CEBL,$BDDAT
1735 022230  001403                        BEQ     10$             ;BRANCH IF EBL IS RESET
1736 022232  104124                        EMT     124
1737 022234  052703  000001                BIS     #BIT0,R3        ;SET ERROR FLAG
1738
1739                                ;SET AND RESET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-1.
1740 022240                         10$:
1741 022240  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1742 022246  012760  020001  000024        MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
1743 022254  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1744 022262  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1745 022270  042737  157777  001142        BIC     #^CEBL,$BDDAT
1746 022276  001403                        BEQ     20$             ;BRANCH IF EBL IS RESET
1747 022300  104125                        EMT     125
1748 022302  052703  000001                BIS     #BIT0,R3        ;SET ERROR FLAG
1749
1750                                ;RESET AND SET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-0
1751 022306                         20$:
1752 022306  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1753 022314  012760  020001  000024        MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
1754 022322  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
```

```
1755 022330  042737  157777  001142          BIC     #^CEBL,$BDDAT
1756 022336  001006                          BNE     30$             ;BRANCH IF EBL IS SET
1757 022340  012737  020000  001140          MOV     #EBL,$GDDAT
1758 022346  104126                          EMT     126
1759 022350  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
1760
1761                                 ;IF NO PREVIOUS ERRORS, TEST FOR ADJACENT BIT INTERFERENCE ON 'DEBL'.
1762 022354                          30$:
1763 022354  005703                          TST     R3
1764 022356  001042                          BNE     70$             ;BRANCH IF ANY ERROR
1765 022360  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1766
1767                                 ;WRITE, READ AND VERIFY THE TEST PATTERN IN R2
1768 022364                          40$:
1769 022364  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1770 022372  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1771 022376  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1772 022404  042737  157777  001142          BIC     #^CEBL,$BDDAT
1773 022412  010203                          MOV     R2,R3           ;GENERATE EXPECTED RESULT
1774 022414  042703  157777                  BIC     #^CEBL,R3
1775 022420  020337  001142                  CMP     R3,$BDDAT
1776 022424  001405                          BEQ     50$             ;BRANCH IF EBL IS OK
1777 022426  010337  001140                  MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
1778 022432  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1779 022436  104127                          EMT     127
1780
1781                                 ;SHIFT TO NEXT BIT POSITION
1782 022440                          50$:
1783 022440  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
1784 022444  001002                          BNE     60$
1785 022446  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE DMD
1786 022452  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
1787 022454  001403                          BEQ     70$             ;EXIT IF DONE
1788 022456  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1789 022462  000740                          BR      40$             ;CONTINUE TEST
1790
1791 022464                          70$:                            ;END OF TEST
1792
1805
1806                                 ;;*****************************************************************
                                     ;*TEST 36       LAST SECTOR, LAST TRACK TEST
                                     ;*
                                     ;*TRANSFER TEST PATTERN TO RMDA THEN VERIFY LAST SECTOR 'LS' AND LAST
                                     ;*SECTOR/TRACK 'LST' FOR EACH TRANSFER. THE TABLE BELOW LISTS THE VALUE
                                     ;*OF RMDA FOR WHICH LS AND LST ARE SET.
                                     ;*
                                     ;*                      18 BIT MODE     16 BIT MODE
                                     ;*              LS =    XXX035          XXX037
                                     ;*
                                     ;*RM02/03 -     LST=    002035          002037
                                     ;*RM05    -     LST=    011035          011037
                                     ;;*****************************************************************
      022464                         TST36:
      022464  000004                         SCOPE                   ;SCOPE CALL
      022466  000240                         NOP
      022470  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
      022474  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
```

```
        022500  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        022504  012737  000036  001226          MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  1807
  1808  022512  005002                          CLR     R2              ;INITIALIZE TEST PATTERN
  1809  022514  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
  1810  022520  062737  000024  001136          ADD     #RMMR1,$BDADR
  1811  022526  005037  001440                  CLR     RMOFO           ;START IN 18 BIT MODE
  1812  022532  013737  001330  023022          MOV     LSTRK,80$       ;SETUP LAST TRACK AND
  1813  022540  112737  000035  023022          MOVB    #035,80$        ;LAST SECTOR (29.)
  1814  022546                          10$:
  1815  022546  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  1816  022552  013760  001440  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
  1817  022560  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
  1818  022564  016037  000024  001356          MOV     RMMR1(R0),RMMR1I        ;STORE RMMR1 IN INPUT BUFFER
  1819  022572  013737  001356  001142          MOV     RMMR1I,$BDDAT   ;VERIFY 'LS''
  1820  022600  042737  177773  001142          BIC     #^CLS,$BDDAT
  1821  022606  005037  001140                  CLR     $GDDAT          ;GENERATE EXPECTED 'LS''
  1822  022612  032737  010000  001440          BIT     #FMT16,RMOFO    ;16 BIT MODE ?
  1823  022620  001004                          BNE     20$             ;YES!!
  1824  022622  123702  023022                  CMPB    80$,R2          ;18 BIT MODE LAST SECTOR ?
  1825  022626  001007                          BNE     35$             ;NO !!
  1826  022630  000403                          BR      30$
  1827  022632  123702  023022          20$:    CMPB    80$,R2          ;16 BIT MODE LAST SECTOR ?
  1828  022636  001003                          BNE     35$             ;NO !!
  1829
  1830  022640  052737  000004  001140  30$:    BIS     #LS,$GDDAT      ;LS SHOULD BE ON-16 BIT MODE
  1831  022646  023737  001140  001142  35$:    CMP     $GDDAT,$BDDAT
  1832  022654  001407                          BEQ     40$             ;BRANCH IF LS IS CORRECT
  1833  022656  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
  1834  022662  013737  001440  001176          MOV     RMOFO,$TMP1     ;SAVE OFFSET REGISTER FOR ERROR
  1835  022670  104130                          EMT     130
  1836  022672  000454                          BR      90$             ;SKIP TO NEXT
  1837
  1838  022674  013737  001356  001142  40$:    MOV     RMMR1I,$BDDAT   ;VERIFY 'LST''
  1839  022702  042737  177775  001142          BIC     #^CLST,$BDDAT
  1840  022710  005037  001140                  CLR     $GDDAT          ;GENERATE EXPECTED 'LST''
  1841  022714  032737  010000  001440          BIT     #FMT16,RMOFO    ;16 BIT MODE??
  1842  022722  001004                          BNE     50$             ;YES!!
  1843  022724  023702  023022                  CMP     80$,R2          ;18 BIT MODE LAST TRACK/SECTOR ?
  1844  022730  001007                          BNE     65$             ;NO !!
  1845  022732  000403                          BR      60$
  1846  022734  023702  023022          50$:    CMP     80$,R2          ;16 BIT MODE LAST TRACK/SECTOR ?
  1847  022740  001003                          BNE     65$             ;NO !!
  1848
  1849  022742  052737  000002  001140  60$:    BIS     #LST,$GDDAT     ;LST SHOULD BE SET
  1850  022750  023737  001140  001142  65$:    CMP     $GDDAT,$BDDAT
  1851  022756  001404                          BEQ     70$
  1852  022760  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
  1853  022764  104131                          EMT     131
  1854  022766  000416                          BR      90$             ;SKIP TO NEXT
  1855
  1856                                  ;ADVANCE TO NEXT TEST PATTERN, CHANGE TO 16 BIT MODE IF ALL
  1857                                  ;18 BIT TESTS DONE.
  1858  022770                          70$:
  1859  022770  005202                          INC     R2              ;INCREMENT PATTERN
  1860  022772  001265                          BNE     10$             ;CONTINUE IF NOT DONE
  1861  022774  032737  010000  001440          BIT     #FMT16,RMOFO    ;DONE 16 BIT TEST ?
```

```
1862 023002  001010                        BNE     90$             ;YES!!
1863 023004  012737  010000  001440        MOV     #FMT16,RMOFO    ;DO 16 BIT FORMAT TEST
1864 023012  112737  000037  023022        MOVB    #037,80$        ;SET LAST SECTOR FOR 16 BIT MODE (31.)
1865 023020  000652                        BR      10$
1866
1867 023022  000000                80$:    .WORD   0               ;HOLDS LAST TRACK/SECTOR ADDRESS
1868
1869 023024                        90$:
1870
1874                               ;;*****************************************************************
                                   ;*TEST 37       RMDA COUNT TEST

                                   ;;*****************************************************************
     023024                        TST37:
     023024  000004                        SCOPE                   ;SCOPE CALL
     023026  000240                        NOP
     023030  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     023034  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     023040  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     023044  012737  000037  001226        MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1875
1876 023052  010037  001136                MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1877 023056  062737  000006  001136        ADD     #RMDA,$BDADR
1878 023064  005037  001440                CLR     RMOFO           ;START WITH 18 BIT FORMAT
1879 023070  012737  000001  001140        MOV     #1,$GDDAT       ;SETUP FIRST COUNT
1880 023076  012737  000035  023476        MOV     #29.,110$       ;LAST SECTOR
1881
1882                               ;INCREMENT SECTOR COUNT USING DIAGNOSTIC END OF BLOCK STARTING AT
1883                               ;SECTOR 0 AND CONTINUING UNTIL TRACK ADDRESS INCREMENTS
1884                               ;       .CLEAR THE MASSBUS
1885                               ;       .SET FORMAT
1886                               ;       .LOAD SECTOR AND TRACK ADDRESS
1887                               ;       .ENABLE DEBUG CLOCK
1888                               ;       .SET GO BIT
1889 023104                        10$:
1890 023104  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1891 023110  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1892 023116  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
1893 023124  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
1894 023132  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
1895 023140  013760  001440  000032        MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1896 023146  013737  001440  001174        MOV     RMOFO,$TMPO     ;SAVE OFFSET FOR ERROR TYPE
1897 023154  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1898 023162  012760  000001  000000        MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
1899
1900                               ;SET AND RESET EBL TO INCREMENT RMDA THEN VERIFY RMDA.
1901 023170                        25$:
1902 023170  012760  060001  000024        MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
1903 023176  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1904 023204  016037  000006  001142        MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1905 023212  023737  001142  001140        CMP     $BDDAT,$GDDAT
1906 023220  001402                        BEQ     30$             ;BRANCH IF RMDA OK
1907 023222  104132                        EMT     132
1908 023224  000416                        BR      50$             ;OUT OF SYNC-SKIP TO NEXT
1909
1910                               ;ADVANCE EXPECTED SECTOR COUNT AND CONTINUE IF ONE CYCLE NOT
1911                               ;COMPLETE
```

```
1912 023226                                  30$:
1913 023226  005237  001140                        INC     $GDDAT          ;INCREMENT EXPECTED SECTOR
1914 023232  123737  001142  023476                CMPB    $BDDAT,110$     ;WAS THE LAST SECTOR JUST COUNTED??
1915 023240  001004                                BNE     40$             ;NO!!
1916 023242  105037  001140                        CLRB    $GDDAT          ;YES-NEXT SECTOR SHOULD BE ZERO
1917 023246  105237  001141                        INCB    $GDDAT+1        ;INCREMENT TRACK ADDRESS
1918 023252  105737  001142          40$:          TSTB    $BDDAT          ;HAS A FULL CYCLE BEEN COUNTED??
1919 023256  001401                                BEQ     50$             ;YES-DO NEXT
1920 023260  000743                                BR      25$             ;CONTINUE SECTOR TEST
1921
1922                                         ;INCREMENT TRACK COUNT USING DIAGNOSTIC END OF BLOCK.  START AT TRACK 0,
1923                                         ;LAST SECTOR AND COUNT ONE COMPLETE TRACK CYCLE.
1924 023262                                  50$:
1925 023262  013737  023476  001414                MOV     110$,RMDAO      ;START SECTOR ADDRESS = 0
1926 023270  012737  000400  001140                MOV     #TA1,$GDDAT     ;FIRST VALUE AFTER INCREMENT
1927
1928                                         ;       .CLEAR THE MASSBUS
1929                                         ;       .SET FORMAT
1930                                         ;       .LOAD LAST SECTOR ADDRESS AND TEST TRACK ADDRESS
1931                                         ;       .ENABLE DEBUG CLOCK
1932                                         ;       .SET GO BIT
1933 023276                                  60$:
1934 023276  004737  055614                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1935 023302  013760  001440  000032                MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1936 023310  013760  001414  000006                MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
1937 023316  012760  000001  000024                MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1938 023324  012760  040001  000024                MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1939 023332  012760  000001  000000                MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
1940
1941                                         ;CLOCK RMDA USING DIAGNOSTIC END OF BLOCK
1942 023340  012760  060001  000024                MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
1943 023346  012760  040001  000024                MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1944
1945                                         ;VERIFY RMDA ACCORDING TO $GDDAT
1946 023354  016037  000006  001142                MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1947 023362  023737  001140  001142                CMP     $GDDAT,$BDDAT
1948 023370  001402                                BEQ     70$
1949 023372  104133                                EMT     133
1950 023374  000441                                BR      120$            ;OUT OF SYNC-SKIP TO NEXT
1951
1952                                         ;SETUP FOR NEXT INCREMENT OF RMDA TRACK ADDRESS
1953 023376                                  70$:
1954 023376  1 5237  001141                        INCB    $GDDAT+1        ;ADVANCE EXPECTED TRACK
1955 023402  123737  001143  001331                CMPB    $BDDAT+1,LSTRK+1        ;WAS THE LAST TRACK JUST COUNTED??
1956 023410  001002                                BNE     80$             ;NO!!
1957 023412  005037  001140                        CLR     $GDDAT          ;YES-NEXT TRACK, SECTOR SHOULD BE ZERO
1958 023416  013737  001142  001414  80$:          MOV     $BDDAT,RMDAO    ;HAS A FULL CYCLE BEEN COUNTED??
1959 023424  001404                                BEQ     90$             ;YES!!
1960 023426  113737  023476  001414                MOVB    110$,RMDAO      ;INCREMENT FROM LAST SECTOR
1961 023434  000720                                BR      60$
1962 023436                                  90$:
1963 023436  032737  010000  001440                BIT     #FMT16,RMOFO    ;DONE BOTH FORMATS??
1964 023444  001015                                BNE     120$            ;YES!!
1965 023446  012737  010000  001440                MOV     #FMT16,RMOFO    ;SET FORMAT BIT FOR 16
1966 023454  012737  000037  023476                MOV     #31.,110$       ;SET LAST SECTOR FOR 16 BIT MODE
1967 023462  000400                                BR      100$
1968
```

```
 1969 023464  012737  000001  001140  100$:   MOV     #1,$GDDAT       ;SET FIRST COUNT VALUE
 1970 023472  000137  023104          10$:    JMP     10$             ;REPEAT TEST
 1971
 1972 023476  000000                  110$:   .WORD   0               ;STORAGE FOR LAST SECTOR VALUE
 1973
 1974 023500                           120$:
 1975
 1976                                  ;;*****************************************************************
                                       ;*TEST 40       RMDC COUNT TEST
                                       ;;*****************************************************************
      023500                           TST40:
      023500  000004                          SCOPE                   ;SCOPE CALL
      023502  000240                          NOP
      023504  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      023510  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      023514  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      023520  012737  000040  001226          MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1977
 1978                                  ;CLEAR RMDC, SET FORMAT AND SETUP PROGRAM PARAMETERS
 1979 023526  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
 1980 023532  062737  000034  001136          ADD     #RMDC,$BDADR
 1981 023540  005037  001440                  CLR     RMOFO           ;START WITH 18 BIT FORMAT
 1982 023544  013737  001330  001414          MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
 1983 023552  112737  000035  001414          MOVB    #29.,RMDAO      ;LAST SECTOR
 1984 023560                           10$:
 1985 023560  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 1986 023564  012737  000001  001140          MOV     #1,$GDDAT       ;LOAD FIRST INCREMENTAL VALUE
 1987 023572  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 1988
 1989                                  ;       .CLEAR THE MASSBUS
 1990                                  ;       .SET OFFSET
 1991                                  ;       .LOAD LAST SECTOR AND TRACK ADDRESS
 1992                                  ;       .ENABLE DEBUG CLOCK
 1993                                  ;       .SET GO BIT
 1994 023600                           20$:
 1995 023600  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 1996 023604  013760  001440  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
 1997 023612  013737  001440  001174          MOV     RMOFO,$TMP0     ;SAVE FOR ERROR MSG
 1998 023620  013760  001414  000006          MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
 1999 023626  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 2000 023634  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 2001 023642  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
 2002
 2003                                  ;CLOCK THE CYLINDER ADDRESS USING DEBL
 2004 023650  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
 2005 023656  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 2006 023664  016037  000034  001142          MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
 2007 023672  023737  001140  001142          CMP     $GDDAT,$BDDAT
 2008 023700  001402                          BEQ     30$             ;BRANCH IF RMDC=RMDC+1
 2009 023702  104140                          EMT     140
 2010 023704  000427                          BR      60$             ;OUT OF SYNC-SKIP TO END
 2011
 2012                                  ;ADVANCE EXPECTED RESULT FOR NEXT INCREMENT
 2013 023706                           30$:
 2014 023706  005237  001140                  INC     $GDDAT          ;ADVANCE NEXT RESULT
 2015 023712  022737  002000  001140          CMP     #1024.,$GDDAT   ;SHOULD NEXT VALUE BE ZERO??
```

```
2016 023720 001002                      BNE     40$             ;NO!!
2017 023722 005037 001140               CLR     $GDDAT          ;YES-RMDC SHOULD OVERFLOW
2018 023726 005737 001142       40$:    TST     $BDDAT          ;IS ONE CYCLE COMPLETE??
2019 023732 001401                      BEQ     50$             ;YES!!
2020 023734 000721                      BR      20$             ;CONTINUE
2021 023736                     50$:
2022 023736 032737 010000 001440        BIT     #FMT16,RMOFO    ;DONE 16 BIT FORMAT MODE ?
2023 023744 001007                      BNE     60$             ;YES !!
2024 023746 012737 010000 001440        MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT AND
2025 023754 112737 000037 001414        MOVB    #31.,RMDAO      ;LOAD LAST SECTOR FOR 16 BIT MODE
2026 023762 000676                      BR      10$             ;REPEAT TEST
2027 023764                     60$:
2028
2029                            ;;*****************************************************************
                                ;*TEST 41        LBT TEST

                                ;;*****************************************************************
     023764                     TST41:
     023764 000004                      SCOPE                   ;SCOPE CALL
     023766 000240                      NOP
     023770 012706 001100               MOV     #STACK,SP       ;LOAD THE STACK POINTER
     023774 013700 001276               MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     024000 013701 001462               MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     024004 012737 000041 001226        MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2030
2031 024012 010037 001136               MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2032 024016 062737 000012 001136        ADD     #RMDS,$BDADR
2033
2034 024024 005037 001440               CLR     RMOFO           ;START WITH 18 BIT MODE
2035 024030 013737 001330 001414        MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
2036 024036 112737 000035 001414        MOVB    #29.,RMDAO      ;LAST SECTOR
2037
2038                                    ;       CLEAR THE MASSBUS
2039                                    ;       SET FORMAT
2040                                    ;       LOAD LAST TRACK AND SECTOR
2041                                    ;       LOAD LAST CYLINDER
2042                                    ;
2043                                    ;       VERIFY THAT "LBT" IS RESET
2044 024044                     10$:
2045 024044 004737 055614               JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2046 024050 012760 001466 000034        MOV     #822.,RMDC(R0)  ;LOAD RMDC
2047 024056 013760 001414 000006        MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
2048 024064 013760 001440 000032        MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
2049 024072 013737 001440 001174        MOV     RMOFO,$TMP0     ;SAVE OFFSET REG FOR ERROR MSG
2050 024100 016037 000012 001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2051 024106 042737 175777 001142        BIC     #^CLBT,$BDDAT
2052 024114 001403                      BEQ     20$             ;BRANCH IF LBT IS RESET
2053 024116 005037 001140               CLR     $GDDAT          ;LBT SHOULD BE ZERO
2054 024122 104141                      EMT     141
2055
2056                                    ;       ENABLE DEBUG CLOCK
2057                                    ;       SET GO
2058                                    ;       FORCE EBL
2059                                    ;
2060                                    ;       VERIFY THAT LBT IS SET
2061 024124                     20$:
2062 024124 012760 000001 000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
```

```
 2063 024132  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 2064 024140  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 2065 024146  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
 2066 024154  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
 2067 024162  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
 2068 024170  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 2069 024176  042737  175777  001142          BIC     #^CLBT,$BDDAT
 2070 024204  001005                          BNE     30$             ;BRANCH IF LBT IS SET
 2071 024206  012737  002000  001140          MOV     #LBT,$GDDAT
 2072 024214  104142                          EMT     142
 2073 024216  000413                          BR      40$
 2074 024220                          30$:
 2075 024220  032737  010000  001440          BIT     #FMT16,RMOFO    ;DONE 16 BIT FORMAT ?
 2076 024226  001007                          BNE     40$             ;YES !!
 2077 024230  012737  010000  001440          MOV     #FMT16,RMOFO    ;SET 16 BIT MODE AND
 2078 024236  112737  000037  001414          MOVB    #31.,RMDAO      ;LAST SECTOR
 2079 024244  000677                          BR      10$             ;TEST AGAIN
 2080 024246                          40$:
 2081
 2082                                 ;;*****************************************************************
                                      ;*TEST 42        COMPOSITE ERROR TEST
                                      ;;*****************************************************************
      024246                          TST42:
      024246  000004                          SCOPE                   ;SCOPE CALL
      024250  000240                          NOP
      024252  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      024256  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      024262  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      024266  012737  000042  001226          MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2083
 2084 024274  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 2085 024300  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
 2086 024304  062737  000012  001136          ADD     #RMDS,$BDADR
 2087
 2088                                 ;USING DIAGNOSTIC MODE, CLEAR ALL ERRORS AND VERIFY THAT COMPOSITE
 2089                                 ;ERROR IS RESET.
 2090 024312  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 2091 024320  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 2092 024326  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 2093 024334  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 2094 024342  042737  137777  001142          BIC     #^CERR,$BDDAT
 2095 024350  001403                          BEQ     10$             ;BRANCH IF ERR IS RESET
 2096 024352  005037  001140                  CLR     $GDDAT
 2097 024356  104143                          EMT     143
 2098 024360  012737  040000  001140  10$:     MOV     #ERR,$GDDAT
 2099
 2100                                 ;SET BOTH ERROR REGISTERS AND VERIFY THAT COMPOSITE ERROR IS SET
 2101 024366  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
 2102 024374  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
 2103 024402  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 2104 024410  042737  137777  001142          BIC     #^CERR,$BDDAT
 2105 024416  001001                          BNE     20$             ;BRANCH IF ERR IS SET
 2106 024420  104144                          EMT     144
 2107
 2108                                 ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER1
 2109 024422                          20$:
```

```
2110 024422  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
2111
2112                                        ;WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
2113 024426                                 30$:
2114 024426  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2115 024432  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2116 024440  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
2117 024446  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
2118 024454  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
2119 024460  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2120 024466  042737  137777  001142          BIC     #^CERR,$BDDAT
2121 024474  001005                          BNE     40$             ;BRANCH IF COMPOSITE ERROR SET
2122 024476  010237  001174                  MOV     R2,$TMP0        ;SAVE RMER1 TEST PATTERN
2123 024502  005037  001176                  CLR     $TMP1           ;SAVE RMER2 TEST PATTERN
2124 024506  104145                          EMT     145
2125
2126                                        ;ADVANCE THE TEST PATTERN FOR RMER1
2127 024510                                 40$:
2128 024510  006302                          ASL     R2
2129 024512  001345                          BNE     30$             ;CONTINUE IF TEST NOT DONE
2130
2131                                        ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER2
2132 024514                                 50$:
2133 024514  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
2134 024520  012737  010000  001440          MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT
2135
2136                                        ;WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
2137 024526                                 60$:
2138 024526  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2139 024532  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2140 024540  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
2141 024546  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
2142 024554  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
2143 024560  013760  001440  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
2144 024566  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2145 024574  042737  137777  001142          BIC     #^CERR,$BDDAT
2146 024602  012737  040000  001140          MOV     #ERR,$GDDAT     ;SETUP EXPECTED VALUE FOR COMP ERROR
2147 024610  032702  001567                  BIT     #XNUER2,R2
2148 024614  001402                          BEQ     65$             ;BRANCH IF TEST BIT IS A USED BIT
2149 024616  005037  001140                  CLR     $GDDAT          ;TEST BIT IS NOT USED - ERR SHOULD BE 0
2150 024622  023737  001140  001142  65$:    CMP     $GDDAT,$BDDAT
2151 024630  001405                          BEQ     70$             ;BRANCH IF COMP ERROR IS OK
2152 024632  005037  001174                  CLR     $TMP0           ;SAVE RMER1 TEST PATTERN
2153 024636  010237  001176                  MOV     R2,$TMP1        ;SAVE RMER2 TEST PATTERN
2154 024642  104145                          EMT     145
2155
2156                                        ;ADVANCE THE TEST PATTERN FOR RMER2
2157 024644                                 70$:
2158 024644  006302                          ASL     R2
2159 024646  001327                          BNE     60$             ;CONTINUE IF TEST NOT DONE
2160 024650                                 80$:
2161
2162                                        ;;*****************************************************************
                                            ;*TEST 43        WRITE GO TEST

                                            ;;*****************************************************************
     024650                                 TST43:
```

```
        024650  000004                          SCOPE                   ;SCOPE CALL
        024652  000240                          NOP
        024654  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        024660  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        024664  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        024670  012737  000043  001226          MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   2163
   2164 024676  010037  001136                  MOV     R0,$BDADR       ;COPY RMCS1 ADDRESS
   2165 024702  005002                          CLR     R2              ;INITIALIZE FUNCTION CODE
   2166
   2167                                 ;CLEAR THE MASSBUS, SET DIAGNOSTIC MODE AND ENABLE DEBUG CLOCK
   2168 024704                          10$:
   2169 024704  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
   2170 024710  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
   2171 024716  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
   2172 024724  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
   2173 024732  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
   2174
   2175                                 ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
   2176 024740  010203                          MOV     R2,R3           ;SETUP FUNCTION CODE
   2177 024742  052703  000001                  BIS     #GO,R3
   2178 024746  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
   2179 024752  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
   2180 024760  032737  000001  001142          BIT     #GO,$BDDAT
   2181 024766  001007                          BNE     20$             ;BRANCH IF GO IS SET
   2182
   2183                                 ;REPORT THE ERROR-CANT SET GO WITH THIS FUNCTION CODE
   2184 024770  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
   2185 024776  010337  001140                  MOV     R3,$GDDAT       ;SAVE FUNCTION CODE
   2186 025002  104146                          EMT     146
   2187 025004  000405                          BR      30$
   2188
   2189                                 ;ADVANCE R2 TO THE NEXT FUNCTION CODE
   2190 025006                          20$:
   2191 025006  062702  000002                  ADD     #2,R2
   2192 025012  022702  000076                  CMP     #ILF76,R2
   2193 025016  103332                          BHIS    10$
   2194
   2195 025020                          30$:                            ;END OF TEST
   2196
   2197                                 ;;*****************************************************************
                                        ;*TEST 44        BRANCH MULTIPLEXOR TEST

                                        ;;*****************************************************************
        025020                          TST44:
        025020  000004                          SCOPE                   ;SCOPE CALL
        025022  000240                          NOP
        025024  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        025030  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        025034  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        025040  012737  000044  001226          MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   2198
   2199 025046  010037  001136                  MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
   2200 025052  062737  000040  001136          ADD     #RMMR2,$BDADR
   2201 025060  012702  025304                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
   2202
   2203                                 ;CLEAR THE MASSBUS AND SET DEBUG CLOCK ENABLE
```

```
2204 025064                                    10$:
2205 025064  004737  055614                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2206 025070  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2207 025076  012760  041001  000024                  MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2208 025104  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
2209 025112  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
2210
2211                                           ;THE TEST BIT SHOULD BE ONE BECAUSE THE ADDRESS IS ALL ONES WHEN
2212                                           ;THE COMMAND SEQUENCER IS INITIALIZED.
2213 025120  016037  000040  001142                  MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
2214 025126  032737  010000  001142                  BIT     #TST,$BDDAT
2215 025134  001010                                  BNE     15$             ;BRANCH IF TEST BIT IS ON
2216 025136  042737  167777  001142                  BIC     #^CTST,$BDDAT   ;SETUP FOR ERROR TYPE
2217 025144  012737  010000  001140                  MOV     #TST,$GDDAT
2218 025152  104147                                  EMT     147
2219 025154  000452                                  BR      40$             ;SKIP REST OF TEST
2220
2221                                           ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO THE DEVICE,
2222                                           ;THEN STEP THE COMMAND SEQUENCER ACCORDING TO THE TABLE.
2223 025156                                    15$:
2224 025156  111203                                  MOVB    (R2),R3
2225 025160  052703  000001                          BIS     #GO,R3
2226 025164  042703  177700                          BIC     #^CFNCMSK,R3    ;R3=FUNCTION CODE, GO BIT
2227 025170  010360  000000                          MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2228 025174  010337  001174                          MOV     R3,$TMP0        ;SAVE R3 FOR ERROR MSG
2229
2230 025200  116203  000001                          MOVB    1(R2),R3        ;GET CLOCK COUNT IN R3
2231 025204  042703  177400                          BIC     #^C377,R3
2232 025210                                    20$:
2233 025210  012760  141001  000024                  MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
2234 025216  012760  041001  000024                  MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2235 025224  005303                                  DEC     R3              ;DECREMENT CLOCK COUNT
2236 025226  001370                                  BNE     20$             ;ISSUE CLOCKS TILL ZERO
2237
2238                                           ;GET THE TEST BIT AND COMPARE IT WITH THE TABLE ENTRY
2239 025230  016037  000040  001142                  MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
2240 025236  042737  167777  001142                  BIC     #^CTST,$BDDAT
2241 025244  016237  000002  001140                  MOV     2(R2),$GDDAT
2242 025252  023737  001140  001142                  CMP     $GDDAT,$BDDAT
2243 025260  001402                                  BEQ     30$             ;BRANCH IF TEST BIT OK
2244 025262  104150                                  EMT     150
2245 025264  000406                                  BR      40$             ;SKIP REST OF TEST
2246
2247                                           ;MOVE THE TABLE POINTER AND CONTINUE IF NEXT ENTRY POSITIVE
2248 025266                                    30$:
2249 025266  062702  000004                          ADD     #4,R2
2250 025272  105762  000001                          TSTB    1(R2)
2251 025276  100401                                  BMI     40$             ;BRANCH IF DONE TEST
2252 025300  000671                                  BR      10$             ;REPEAT TEST
2253 025302  000436                            40$:  BR      200$            ;JUMP OVER TABLE
2254
2255                                           ;TABLE OF FUNCTION CODES, CLOCK COUNTS, AND TEST BITS
2256 025304                                    100$:
2257 025304     000                                   .BYTE   NOP             ;MUX ADDRESS=DATA COMMAND
2258 025305     001                                   .BYTE   1
2259 025306  000000                                   .WORD   0               ;TEST BIT=0
2260
```

```
2261 025310     000                     .BYTE   NOP             ;MUX ADDRESS=UNIT READY
2262 025311     002                     .BYTE   2
2263 025312  000000                     .WORD   0               ;TEST BIT=0
2264
2265 025314     010                     .BYTE   DRVCLR          ;MUX ADDRESS=F4
2266 025315     001                     .BYTE   1
2267 025316  010000                     .WORD   TST             ;TEST BIT=1
2268
2269 025320     050                     .BYTE   WCD             ;MUX ADDRESS=F4
2270 025321     001                     .BYTE   1
2271 025322  000000                     .WORD   0               ;TEST BIT=0
2272
2273 025324     012                     .BYTE   RLEASE          ;MUX ADDRESS=F4
2274 025325     001                     .BYTE   1
2275 025326  010000                     .WORD   TST             ;TEST BIT=1
2276
2277 025330     052                     .BYTE   WCH             ;MUX ADDRESS=F4
2278 025331     001                     .BYTE   1
2279 025332  000000                     .WORD   0               ;TEST BIT=0
2280
2281 025334     020                     .BYTE   RIP             ;MUX ADDRESS=F4
2282 025335     001                     .BYTE   1
2283 025336  010000                     .WORD   TST             ;TEST BIT=1
2284
2285 025340     060                     .BYTE   WD              ;MUX ADDRESS=F4
2286 025341     001                     .BYTE   1
2287 025342  000000                     .WORD   0               ;TEST BIT=0
2288
2289 025344     022                     .BYTE   PAKACK          ;MUX ADDRESS=F4
2290 025345     001                     .BYTE   1
2291 025346  010000                     .WORD   TST             ;TEST BIT=1
2292
2293 025350     062                     .BYTE   WH              ;MUX ADDRESS=F4
2294 025351     001                     .BYTE   1
2295 025352  000000                     .WORD   0               ;TEST BIT=0
2296
2297 025354     030                     .BYTE   SEARCH          ;MUX ADDRESS=F4
2298 025355     001                     .BYTE   1
2299 025356  010000                     .WORD   TST             ;TEST BIT=1
2300
2301 025360     070                     .BYTE   RD              ;MUX ADDRESS=F4
2302 025361     001                     .BYTE   1
2303 025362  000000                     .WORD   0               ;TEST BIT=0
2304
2305 025364     032                     .BYTE   ILF32           ;MUX ADDRESS=F4
2306 025365     001                     .BYTE   1
2307 025366  010000                     .WORD   TST             ;TEST BIT=1
2308
2309 025370     072                     .BYTE   RH              ;MUX ADDRESS=F4
2310 025371     001                     .BYTE   1
2311 025372  000000                     .WORD   0               ;TEST BIT=0
2312
2313 025374     000                     .BYTE                   ;END OF TABLE
2314 025375     377                     .BYTE   -1
2315 025376  000000                     .WORD
2316 025400                     200$:                           ;END OF TEST
2317
```

```
2318                                    ;;*****************************************************************
                                        ;*TEST 45       SET/RESET GO TEST
                                        ;;*****************************************************************
     025400                             TST45:
     025400  000004                             SCOPE                   ;SCOPE CALL
     025402  000240                             NOP
     025404  012706  001100                     MOV     #STACK,SP       ;LOAD THE STACK POINTER
     025410  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     025414  013701  001462                     MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     025420  012737  000045  001226             MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2319
2320 025426  012702  026042                     MOV     #200$,R2        ;INITIALIZE FUNCTION CODE POINTER
2321
2322                                    ;CLEAR, THEN SET DIAGNOSTIC MODE, CLEAR COMPOSITE ERROR, SET MEDIUM
2323                                    ;ON LINE AND ENABLE DEBUG CLOCK
2324 025432                             10$:
2325 025432  004737  055614                     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2326 025436  012760  000001  000024             MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2327 025444  012760  041001  000024             MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2328 025452  012760  000000  000014             MOV     #0,RMER1(R0)    ;LOAD RMER1
2329 025460  012760  000000  000042             MOV     #0,RMER2(R0)    ;LOAD RMER2
2330
2331                                    ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1 AND VERIFY GO IS SET
2332 025466  111203                             MOVB    (R2),R3         ;GET FUNCTION CODE
2333 025470  042703  177701                     BIC     #^C<CSIZE!CS1F76>,R3    ;CLEAR UNUSED BITS
2334 025474  052703  000001                     BIS     #GO,R3          ;SET GO
2335 025500  010360  000000                     MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2336 025504  016037  000000  001142             MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2337 025512  032737  000001  001142             BIT     #GO,$BDDAT
2338 025520  001011                             BNE     20$             ;BRANCH IF GO IS SET
2339 025522  042737  177700  001142             BIC     #^C<CFNCMSK>,$BDDAT
2340 025530  010337  001140                     MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
2341 025534  010037  001136                     MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2342 025540  104151                             EMT     151
2343 025542  000536                             BR      100$
2344
2345                                    ;GET READY STATUS AND VERIFY THAT IT IS THE COMPLEMENT OF GO
2346 025544                             20$:
2347 025544  005037  001140                     CLR     $GDDAT          ;EXPECT DRY TO BE OFF
2348 025550  032737  000001  001142             BIT     #GO,$BDDAT      ;WAS GO SET??
2349 025556  001003                             BNE     30$             ;YES!!
2350 025560  012737  000200  001140             MOV     #DRY,$GDDAT     ;GO WAS NOT SET, DRY SHOULD BE
2351 025566                             30$:
     025566  016037  000012  001142             MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2352 025574  042737  177577  001142             BIC     #^CDRY,$BDDAT
2353 025602  023737  001140  001142             CMP     $GDDAT,$BDDAT
2354 025610  001406                             BEQ     40$             ;BRANCH IF DRY IS OK
2355 025612  010037  001136                     MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2356 025616  062737  000012  001136             ADD     #RMDS,$BDADR
2357 025624  104152                             EMT     152
2358
2359                                    ;STEP THE DEBUG CLOCK AND VERIFY THAT GO REMAINS SET
2360 025626                             40$:
2361 025626  116204  000001                     MOVB    1(R2),R4        ;GET NUMBER OF CLOCK CYCLES
2362 025632  042704  177400                     BIC     #^C377,R4
2363 025636                             50$:
```

```
2364 025636 012760 141001 000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2365 025644 012760 041001 000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2366 025652 016037 000000 001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2367 025660 042737 177700 001142          BIC     #^CFNCMSK,$BDDAT  ;CLEAR UNUSED BITS
2368 025666 010037 001136                 MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2369 025672 010337 001140                 MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
2370
2371                                ;DECREMENT CLOCK COUNT AND EXIT LOOP IF ZERO
2372 025676 005304                        DEC     R4
2373 025700 001406                        BEQ     60$
2374 025702 032737 000001 001142          BIT     #GO,$BDDAT      ;IS GO STILL SET??
2375 025710 001352                        BNE     50$             ;YES!!
2376 025712 104153                        EMT     153
2377 025714 000451                        BR      100$            ;OUT OF SYNC=SKIP TO NEXT
2378
2379                                ;GO SHOULD NOW BE RESET AND DRY SHOULD BE SET
2380 025716                        60$:
2381 025716 032737 000001 001142          BIT     #GO,$BDDAT      ;IS GO RESET??
2382 025724 001405                        BEQ     70$             ;YES!!
2383 025726 042737 000001 001140          BIC     #GO,$GDDAT      ;SETUP EXPECTED RESULT
2384 025734 104154                        EMT     154
2385 025736 000440                        BR      100$
2386 025740                        70$:
2387 025740 012737 000200 001140          MOV     #DRY,$GDDAT     ;EXPECT DRIVE READY TO BE SET
2388 025746 032737 000001 001142          BIT     #GO,$BDDAT      ;DID GO RESET??
2389 025754 001402                        BEQ     80$             ;YES!!
2390 025756 005037 001140                 CLR     $GDDAT          ;GO IS SET-
2391 025762                        80$:
     025762 016037 000012 001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2392 025770 042737 177577 001142          BIC     #^CDRY,$BDDAT
2393 025776 010037 001136                 MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2394 026002 062737 000012 001136          ADD     #RMDS,$BDADR
2395 026010 023737 001140 001142          CMP     $GDDAT,$BDDAT   ;IS DRIVE READY OK??
2396 026016 001401                        BEQ     90$             ;YES!!
2397 026020 104152                        EMT     152
2398
2399                               ;ADVANCE TO THE NEXT FUNCTION CODE TO BE TESTED-EXIT IF DONE
2400 026022                        90$:
2401 026022 062702 000002                 ADD     #2,R2           ;MOVE TABLE POINTER
2402 026026 105762 000001                 TSTB    1(R2)           ;END OF TABLE??
2403 026032 100402                        BMI     100$            ;YES!!
2404 026034 000137 025432                 JMP     10$             ;TEST THIS FUNCTION CODE
2405 026040 000423                 100$:  BR      300$            ;GO TO NEXT TEST
2406
2407                               ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
2408 026042                        200$:
2409 026042      002                      .BYTE   ILF02           ;ILLEGAL FUNCTION CODE #2
2410 026043      001                      .BYTE   1
2411
2412 026044      004                      .BYTE   SEEK            ;SEEK COMMAND
2413 026045      001                      .BYTE   1
2414
2415 026046      006                      .BYTE   RECAL           ;RECALIBRATE COMMAND
2416 026047      001                      .BYTE   1
2417
2418 026050      014                      .BYTE   OFFSET          ;OFFSET COMMAND
2419 026051      001                      .BYTE   1
```

```
2420
2421 026052  016                    .BYTE   RTC             ;RETURN TO CENTER LINE COMMAND
2422 026053  001                    .BYTE   1
2423
2424 026054  024                    .BYTE   ILF24           ;ILLEGAL FUNCTION CODE #24
2425 026055  001                    .BYTE   1
2426
2427 026056  026                    .BYTE   ILF26           ;ILLEGAL FUNCTION CODE #26
2428 026057  001                    .BYTE   1
2429
2430 026060  034                    .BYTE   ILF34           ;ILLEGAL FUNCTION CODE #34
2431 026061  001                    .BYTE   1
2432
2433 026062  036                    .BYTE   ILF36           ;ILLEGAL FUNCTION CODE #36
2434 026063  001                    .BYTE   1
2435
2436 026064  042                    .BYTE   ILF42           ;ILLEGAL FUNCTION CODE #42
2437 026065  001                    .BYTE   1
2438
2439 026066  044                    .BYTE   ILF44           ;ILLEGAL FUNCTION CODE #44
2440 026067  001                    .BYTE   1
2441
2442 026070  046                    .BYTE   ILF46           ;ILLEGAL FUNCTION CODE #46
2443 026071  001                    .BYTE   1
2444
2445 026072  054                    .BYTE   ILF54           ;ILLEGAL FUNCTION CODE #54
2446 026073  001                    .BYTE   1
2447
2448 026074  056                    .BYTE   ILF56           ;ILLEGAL FUNCTION CODE #56
2449 026075  001                    .BYTE   1
2450
2451 026076  064                    .BYTE   ILF64           ;ILLEGAL FUNCTION CODE #64
2452 026077  001                    .BYTE   1
2453
2454 026100  066                    .BYTE   ILF66           ;ILLEGAL FUNCTION CODE #66
2455 026101  001                    .BYTE   1
2456
2457 026102  074                    .BYTE   ILF74           ;ILLEGAL FUNCTION CODE #74
2458 026103  001                    .BYTE   1
2459
2460 026104  076                    .BYTE   ILF76           ;ILLEGAL FUNCTION CODE #76
2461 026105  001                    .BYTE   1
2462
2463 026106  000                    .BYTE                   ;END OF TABLE
2464 026107  377                    .BYTE   -1
2465
2466 026110                  300$:                          ;END OF TEST
2467
2468
                            ;;*****************************************************************
                            ;*TEST 46       END 1 RESET GO TEST
                            ;;*****************************************************************
     026110                 TST46:
     026110  000004                 SCOPE                  ;SCOPE CALL
     026112  000240                 NOP
     026114  012706  001100         MOV     #STACK,SP      ;LOAD THE STACK POINTER
     026120  013700  001276         MOV     $BASE,R0       ;R0 = UNIBUS ADDRESS
```

```
        026124  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        026130  012737  000046  001226          MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  2469
  2470  026136  012702  026374                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
  2471  026142  010037  001136                  MOV     R0,$BDADR       ;COPY RMCS1 ADDRESS
  2472
  2473                                  ;CLEAR MASSBUS, THEN SET MEDIUM ON LINE AND ENABLE DEBUG CLOCK
  2474  026146                          10$:
  2475  026146  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  2476  026152  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
  2477  026160  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  2478  026166  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
  2479  026174  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
  2480
  2481                                  ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
  2482  026202  111203                          MOVB    (R2),R3         ;GET FUNCTION CODE FROM
  2483  026204  042703  177701                  BIC     #^C1F76,R3      ;TABLE AND SET GO
  2484  026210  052703  000001                  BIS     #GO,R3
  2485  026214  010337  001140                  MOV     R3,$GDDAT       ;SAVE FUNCTION CODE FOR MSG
  2486  026220  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
  2487  026224  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
  2488  026232  032737  000001  001142          BIT     #GO,$BDDAT
  2489  026240  001005                          BNE     20$             ;BRANCH IF GO IS SET
  2490  026242  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
  2491  026250  104151                          EMT     151
  2492  026252  000447                          BR      60$             ;OUT OF SYNC-SKIP
  2493
  2494                                  ;GET THE NUMBER OF CLOCK CYCLES FROM THE TABLE, SAVE EXPECTED STATUS
  2495  026254                          20$:
  2496  026254  116204  000001                  MOVB    1(R2),R4        ;R4=CLOCK COUNT
  2497  026260  042704  177400                  BIC     #^C377,R4
  2498
  2499                                  ;STEP THE DEBUG CLOCK AND VERIFY GO STATUS ON UNTIL CLOCK COUNT EXPIRES.
  2500  026264                          30$:
  2501  026264  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
  2502  026272  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  2503  026300  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
  2504  026306  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
  2505  026314  005304                          DEC     R4
  2506  026316  001406                          BEQ     40$             ;BRANCH IF GO SHOULD BE OFF
  2507  026320  032737  000001  001142          BIT     #GO,$BDDAT
  2508  026326  001356                          BNE     30$             ;CONTINUE IF GO IS ON
  2509  026330  104153                          EMT     153
  2510  026332  000417                          BR      60$             ;OUT OF SYNC-SKIP
  2511
  2512                                  ;VERIFY THAT GO RESET AT END1
  2513  026334                          40$:
  2514  026334  032737  000001  001142          BIT     #GO,$BDDAT      ;DID GO RESET??
  2515  026342  001405                          BEQ     50$             ;YES!!
  2516  026344  042737  000001  001140          BIC     #GO,$GDDAT
  2517  026352  104154                          EMT     154
  2518  026354  000406                          BR      60$
  2519
  2520                                  ;GET THE NEXT FUNCTION CODE FROM THE TABLE
  2521  026356                          50$:
  2522  026356  062702  000002                  ADD     #2,R2
  2523  026362  105762  000001                  TSTB    1(R2)
```

```
 2524 026366  100401                    BMI     60$             ;BRANCH IF END OF TABLE
 2525 026370  000666                    BR      10$             ;TEST THIS FUNCTION CODE
 2526 026372  000404            60$:    BR      200$            ;JUMP OVER TABLE
 2527
 2528                           ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
 2529 026374                    100$:
 2530 026374      012                   .BYTE   RLEASE          ;RELEASE COMMAND
 2531 026375      002                   .BYTE   2
 2532
 2533 026376      030                   .BYTE   SEARCH          ;SEARCH COMMAND
 2534 026377      002                   .BYTE   2
 2535
 2536 026400      032                   .BYTE   ILF32           ;ILLEGAL FUNCTON #32
 2537 026401      002                   .BYTE   2
 2538
 2539 026402      000                   .BYTE                   ;END OF TABLE
 2540 026403      377                   .BYTE   -1
 2541 026404                    200$:                           ;END OF TEST
 2542
 2543                           ;;*****************************************************************
                                ;*TEST 47        SET PULSE TEST
                                ;;*****************************************************************
      026404                    TST47:
      026404  000004                    SCOPE                   ;SCOPE CALL
      026406  000240                    NOP
      026410  012706  001100            MOV     #STACK,SP       ;LOAD THE STACK POINTER
      026414  013700  001276            MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      026420  013701  001462            MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      026424  012737  000047  001226    MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2544
 2545 026432  010037  001136            MOV     R0,$BDADR       ;COPY REG ADDRESS FOR MSG
 2546 026436  062737  000024  001136    ADD     #RMMR1,$BDADR
 2547 026444  012702  026702            MOV     #1C0$,R2        ;INITIALIZE TABLE POINTER
 2548
 2549                           ;CLEAR THE MASS BUS, ENABLE DEBUG CLOCK, AND RESET ERROR REGISTERS
 2550 026450                    10$:
 2551 026450  004737  055614            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 2552 026454  012760  000001  000024    MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 2553 026462  012760  041001  000024    MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
 2554 026470  012760  000000  000014    MOV     #0,RMER1(R0)    ;LOAD RMER1
 2555 026476  012760  000000  000042    MOV     #0,RMER2(R0)    ;LOAD RMER2
 2556
 2557                           ;VERIFY THAT CONTINUE, "CONT" IS RESET AFTER CLEAR
 2558 026504  016037  000024  001142    MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 2559 026512  042737  177677  001142    BIC     #^CCONT,$BDDAT
 2560 026520  001404                    BEQ     20$             ;BRANCH IF CONT WAS CLEARED
 2561 026522  005037  001140            CLR     $GDDAT          ;FOR ERROR MSG
 2562 026526  104155                    EMT     155
 2563 026530  000463                    BR      70$
 2564
 2565                           ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO RMCS1
 2566 026532                    20$:
 2567 026532  111203                    MOVB    (R2),R3
 2568 026534  052703  000001            BIS     #GO,R3
 2569 026540  042703  177700            BIC     #^CFNCMSK,R3    ;R3=FUNCTION CODE AND GO
 2570 026544  010360  000000            MOV     R3,RMCS1(R0)    ;LOAD RMCS1
```

```
2571 026550  010337  001174                    MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
2572
2573                                    ;GET THE CLOCK COUNT FROM THE TABLE
2574 026554  116203  000001                    MOVB    1(R2),R3
2575 026560  042703  177400                    BIC     #^C377,R3
2576
2577                                    ;GET THE BIT STREAM FOR CONTINUE FROM THE TABLE
2578 026564  016204  000002                    MOV     2(R2),R4
2579
2580                                    ;STEP THE COMMAND SEQUENCER AND VERIFY CONTINUE STATUS
2581 026570                             30$:
2582 026570  012760  141001  000024            MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
2583 026576  012760  041001  000024            MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2584 026604  016037  000024  001142            MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
2585 026612  042737  177677  001142            BIC     #^CCONT,$BDDAT
2586 026620  005037  001140                    CLR     $GDDAT          ;GENERATE EXPECTED CONTINUE
2587 026624  032704  000001                    BIT     #BIT0,R4
2588 026630  001403                            BEQ     40$
2589 026632  012737  000100  001140            MOV     #CONT,$GDDAT
2590 026640  023737  001140  001142  40$:      CMP     $GDDAT,$BDDAT
2591 026646  001402                            BEQ     50$             ;BRANCH IF CONTINUE IS OK
2592 026650  104156                            EMT     156
2593 026652  000412                            BR      70$             ;SKIP
2594
2595                                    ;DECREMENT CLOCK COUNT AND SHIFT BIT STREAM
2596 026654                             50$:
2597 026654  005303                            DEC     R3
2598 026656  001402                            BEQ     60$             ;BRANCH IF CLOCK COUNT EXPIRED
2599 026660  006204                            ASR     R4              ;SHIFT TO NEXT CONTINUE BIT
2600 026662  000742                            BR      30$             ;TEST NEXT CLOCK CYCLE
2601
2602                                    ;ADVANCE TABLE POINTER-EXIT IF DONE
2603 026664                             60$:
2604 026664  062702  000004                    ADD     #4,R2
2605 026670  105762  000001                    TSTB    1(R2)
2606 026674  100401                            BMI     70$             ;EXIT IF CLOCK COUNT NEGATIVE
2607 026676  000664                            BR      10$             ;CONTINUE TEST
2608 026700  000442                    70$:    BR      200$            ;JUMP OVER TABLE
2609
2610                                    ;TABLE OF FUNCTION CODES, CLOCK COUNTS AND CONTINUE BITS FOR TEST
2611 026702                             100$:
2612 026702     000                            .BYTE   NOP             ;NOP COMMAND
2613 026703     004                            .BYTE   4               ;4 CLOCKS
2614 026704  000000                            .WORD   ^B0000          ;CONTINUE=0000
2615
2616 026706     002                            .BYTE   ILF02           ;ILLEGAL FUNCTION 2
2617 026707     002                            .BYTE   2
2618 026710  000000                            .WORD   ^B00
2619
2620 026712     004                            .BYTE   SEEK            ;SEEK COMMAND
2621 026713     002                            .BYTE   2
2622 026714  000000                            .WORD   ^B00
2623
2624 026716     006                            .BYTE   RECAL           ;RECALIBRATE COMMAND
2625 026717     002                            .BYTE   2
2626 026720  000000                            .WORD   ^B00
2627
```

```
2628 026722     010                     .BYTE   DRVCLR          ;DRIVE CLEAR COMMAND
2629 026723     002                     .BYTE   2
2630 026724  000001                     .WORD   ^B01
2631
2632 026726     012                     .BYTE   RLEASE          ;RELEASE COMMAND
2633 026727     003                     .BYTE   3
2634 026730  000000                     .WORD   ^B000
2635
2636 026732     014                     .BYTE   OFFSET          ;OFFSET COMMAND
2637 026733     002                     .BYTE   2
2638 026734  000000                     .WORD   ^B00
2639
2640 026736     016                     .BYTE   RTC             ;RETURN TO CENTER COMMAND
2641 026737     002                     .BYTE   2
2642 026740  000000                     .WORD   ^B00
2643
2644 026742     020                     .BYTE   RIP             ;READ IN PRESET COMMAND
2645 026743     004                     .BYTE   4
2646 026744  000016                     .WORD   ^B1110
2647
2648 026746     022                     .BYTE   PAKACK          ;PACK ACKNOWLEDGE
2649 026747     004                     .BYTE   4
2650 026750  000016                     .WORD   ^B1110
2651
2652 026752     024                     .BYTE   ILF24           ;ILLEGAL FUNCTION 24
2653 026753     002                     .BYTE   2
2654 026754  000000                     .WORD   ^B00
2655
2656 026756     026                     .BYTE   ILF26           ;ILLEGAL FUNCTION 26
2657 026757     002                     .BYTE   2
2658 026760  000000                     .WORD   ^B00
2659
2660 026762     030                     .BYTE   SEARCH          ;SEARCH COMMAND
2661 026763     003                     .BYTE   3
2662 026764  000000                     .WORD   ^B000
2663
2664 026766     032                     .BYTE   ILF32           ;ILLEGAL FUNCTION 32
2665 026767     003                     .BYTE   3
2666 026770  000000                     .WORD   ^B000
2667
2668 026772     034                     .BYTE   ILF34           ;ILLEGAL FUNCTION 34
2669 026773     002                     .BYTE   2
2670 026774  000000                     .WORD   ^B00
2671
2672 026776     036                     .BYTE   ILF36           ;ILLEGAL FUNCTION 36
2673 026777     002                     .BYTE   2
2674 027000  000000                     .WORD   ^B00
2675
2676 027002     000                     .BYTE                   ;END OF TABLE
2677 027003     377                     .BYTE   -1
2678 027004  000000                     .WORD
2679
2680 027006                     200$:                           ;END OF TEST
2681
2682                             ;;*****************************************************************
                                ;*TEST 50        SET/RESET IVC TEST
```

```
                                ;;*****************************************************************
027006                          TST50:
027006  000004                          SCOPE                   ;SCOPE CALL
027010  000240                          NOP
027012  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
027016  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
027022  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
027026  012737  000050  001226          MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2683
2684 027034  010037  001136            MOV     R0,$BDADR       ;SETUP REG ADDRESS
2685 027040  062737  000042  001136    ADD     #RMER2,$BDADR
2686 027046  005002                    CLR     R2              ;R2=FUNCTION CODE
2687
2688                            ;INITIALIZE AND VERIFY THAT IVC STATUS IS ZERO.
2689 027050                     10$:
2690 027050  004737  055614            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2691 027054  012760  000001  000024    MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2692 027062  012760  041001  000024    MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2693 027070  012760  000000  000014    MOV     #0,RMER1(R0)    ;LOAD RMER1
2694 027076  012760  000000  000042    MOV     #0,RMER2(R0)    ;LOAD RMER2
2695 027104  016037  000042  001142    MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2696 027112  042737  167777  001142    BIC     #^CIVC,$BDDAT
2697 027120  001404                    BEQ     20$             ;BRANCH IF IVC IS ZERO
2698 027122  005037  001140            CLR     $GDDAT
2699 027126  104157                    EMT     157
2700 027130  000444                    BR      40$             ;SKIP REST OF TEST
2701
2702
2703                            ;LOAD THE FUNCTION CODE WITH GO BIT, STEP THE COMMAND SEQUENCER OFF
                                ;ADDRESS 0 AND VERIFY IVC STATUS.
2704 027132                     20$:
2705 027132  010203                    MOV     R2,R3           ;SETUP FUNCTION CODE
2706 027134  052703  000001            BIS     #GO,R3
2707 027140  010360  000000            MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2708 027144  012760  141001  000024    MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2709 027152  012760  041001  000024    MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2710 027160  016037  000042  001142    MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2711 027166  042737  167777  001142    BIC     #^CIVC,$BDDAT   ;SET ACTUAL STATUS
2712 027174  016237  062752  001140    MOV     FNCDTB(R2),$GDDAT  ;SETUP EXPECTED STATUS FROM
2713 027202  042737  167777  001140    BIC     #^CIVC,$GDDAT   ;FUNCTION CODE TABLE
2714 027210  023737  001140  001142    CMP     $GDDAT,$BDDAT
2715 027216  001403                    BEQ     30$             ;BRANCH IF IVC IS OK
2716 027220  010237  001174            MOV     R2,$TMP0        ;SAVE FUNCTION CODE FOR MSG
2717 027224  104160                    EMT     160
2718
2719                            ;ADVANCE FUNCTION CODE AND REPEAT TEST IF NOT DONE
2720 027226                     30$:
2721 027226  062702  000002            ADD     #2,R2
2722 027232  022702  000076            CMP     #ILF76,R2
2723 027236  103401                    BLO     40$             ;BRANCH IF DONE TEST
2724 027240  000703                    BR      10$
2725
2726 027242                     40$:                            ;END OF TEST
2727
2728                            ;;*****************************************************************
                                ;*TEST 51        SET LSC TEST

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

```
                                    TST51:
027242                                      SCOPE                   ;SCOPE CALL
027242  000004                              NOP
027244  000240                              MOV     #STACK,SP       ;LOAD THE STACK POINTER
027246  012706  001100                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
027252  013700  001276                      MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
027256  013701  001462                      MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
027262  012737  000051  001226
2729
2730 027270  010037  001136                MOV     R0,$BDADR
2731 027274  062737  000042  001136        ADD     #RMER2,$BDADR
2732
2733                                ;INITIALIZE AND VERIFY THAT LOSS OF SYSTEM CLOCK, 'LSC', IS RESET
2734 027302  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2735 027306  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2736 027314  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
2737 027322  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
2738 027330  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
2739 027336  016037  000042  001142        MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2740 027344  042737  173777  001142        BIC     #^CLSC,$BDDAT
2741 027352  001403                        BEQ     10$             ;BRANCH IF LSC IS ZERO
2742 027354  005037  001140                CLR     $GDDAT
2743 027360  104161                        EMT     161
2744
2745                                ;WITH DEBUG CLOCK ENABLED, SET GO AND WAIT FOR ONE SHOT TO SET
2746 027362                         10$:
027362  012760  000001  000000            MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
2747 027370  012737  000001  001530        MOV     #1,WATCH        ;SET WATCHDOG TIMER VALUE
027376  004777  152130                    JSR     PC,@CLOCK       ;START THE CLOCK
2748 027402  005737  001530         20$:   TST     WATCH
2749 027406  001375                        BNE     20$             ;WAIT FOR WATCH ZERO
2750 027410  004777  152120                JSR     PC,@STOP        ;STOP THE CLOCK
2751
2752                                ;ONE SHOT SHOULD BE SET-DISABLE DIAGNOSTIC CLOCK AND LSC SHOULD SET.
2753 027414  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2754 027422  016037  000042  001142        MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2755 027430  042737  173777  001142        BIC     #^CLSC,$BDDAT
2756 027436  001004                        BNE     30$             ;BRANCH IF LSC SET
2757 027440  012737  004000  001140        MOV     #LSC,$GDDAT
2758 027446  104162                        EMT     162
2759
2760 027450                         30$:                            ;END OF TEST
2761
2762                                ;;*****************************************************************
                                    ;*TEST 52        DECODE TEST

                                    ;;*****************************************************************
027450                              TST52:
027450  000004                              SCOPE                   ;SCOPE CALL
027452  000240                              NOP
027454  012706  001100                      MOV     #STACK,SP       ;LOAD THE STACK POINTER
027460  013700  001276                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
027464  013701  001462                      MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
027470  012737  000052  001226              MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2763
2764 027476  005037  001422                CLR     RMER10          ;NO ERROR FIRST TEST
2765 027502                         5$:
2766 027502  004737  030062                JSR     PC,100$         ;INITIALIZE
```

```
                                        ;EXECUTE A PACK ACKNOWLEDGE AND CHECK VOLUME VALID
027506  013760  001422  000014            MOV     RMER10,RMER1(R0)        ;LOAD RMFR1
027514  012760  000023  000000            MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
027522  012703  000003                    MOV     #3,R3
027526                                  10$:
027526  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
027534  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
027542  005303                            DEC     R3
027544  001370                            BNE     10$             ;ISSUE NEXT CLOCK IF COUNT NOT 0
027546  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
027554  042737  177677  001142            BIC     #^CVV,$BDDAT
027562  001414                            BEQ     20$             ;BRANCH IF VV IS ZERO
027564  005737  001422                    TST     RMER10
027570  001427                            BEQ     70$             ;BRANCH IF VV SHOULD BE SET
027572  005037  001140                    CLR     $GDDAT          ;SETUP ERROR MESSAGE
027576  010037  001136                    MOV     R0,$BDADR
027602  062737  000012  001136            ADD     #RMDS,$BDADR
027610  104163                            EMT     163
027612  000522                            BR      80$             ;SKIP
027614                                  20$:
027614  004737  030062                    JSR     PC,100$         ;INITIALIZE AND SET DIAGNOSTIC MODE

                                        ;EXECUTE A READ IN PRESET AND CHECK VOLUME VALID
027620  013760  001422  000014            MOV     RMER10,RMER1(R0)        ;LOAD RMER1
027626  012760  000021  000000            MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
027634  012703  000003                    MOV     #3,R3           ;R3=CLOCK COUNT
027640                                  30$:
027640  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
027646  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
027654  005303                            DEC     R3
027656  001370                            BNE     30$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
027660  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
027666  042737  177677  001142            BIC     #^CVV,$BDDAT
027674  001414                            BEQ     40$             ;BRANCH IF VOLUME VALID NOT SET
027676  005737  001422                    TST     RMER10
027702  001462                            BEQ     70$             ;BRANCH IF VOLUME VALID SHOULD BE SET
027704  005037  001140                    CLR     $GDDAT          ;SETUP ERROR MESSAGE
027710  010037  001136                    MOV     R0,$BDADR
027714  062737  000012  001136            ADD     #RMDS,$BDADR
027722  104163                            EMT     163
027724  000455                            BR      80$             ;SKIP
027726                                  40$:
027726  004737  030062                    JSR     PC,100$         ;INITIALIZE AND SET DIAGNOSTIC MODE

                                        ;EXECUTE A WRITE CHECK DATA AND CHECK OCCUPIED
027732  013760  001422  000014            MOV     RMER10,RMER1(R0)        ;LOAD RMER1
027740  012760  000051  000000            MOV     #WCD!GO,RMCS1(R0)       ;LOAD RMCS1
027746  012703  000002                    MOV     #2,R3           ;R3=CLOCK COUNT
027752                                  50$:
027752  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
027760  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
027766  005303                            DEC     R3
027770  001370                            BNE     50$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
027772  016037  000024  001142            MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
030000  042737  077777  001142            BIC     #^COCC,$BDDAT
030006  001414                            BEQ     60$             ;BRANCH IF OCCUPIED IS RESET
```

```
2821 030010  005737  001422                TST     RMER10
2822 030014  001415                        BEQ     70$             ;BRANCH IF OCCUPIED SHOULD BE SET
2823 030016  005037  001140                CLR     $GDDAT          ;SETUP ERROR MESSAGE
2824 030022  010037  001136                MOV     R0,$BDADR
2825 030026  062737  000024  001136        ADD     #RMMR1,$BDADR
2826 030034  104164                        EMT     164
2827 030036  000410                        BR      80$
2828
2829                                ;VOLUME VALID AND OCCUPIED DID NOT SET-SEE IF COMP ERROR WAS ACTIVE
2830 030040                         60$:
2831 030040  005737  001422                TST     RMER10
2832 030044  001005                        BNE     80$             ;BRANCH IF COMP ERROR WAS SET
2833
2834                                ;COULD NOT SET VV OR OCCUPIED-SUSPECT DECODE FLOP NOT SETTING
2835 030046  104165                        EMT     165
2836
2837                                ;REPEAT TEST WITH COMPOSITE ERROR ACTIVE-VERIFY THAT DECODE FLOP
2838                                ;DOES NOT SET, AS INDICATED BY VOLUME VALID AND OCCUPIED.
2839 030050                         70$:
2840 030050  012737  040000  001422        MOV     #UNS,RMER10     ;USE UNSAFE TO SET COMP ERROR
2841 030056  000611                        BR      5$
2842
2843 030060  000510                80$:    BR      200$            ;END OF TEST
2844
2845                                ;;*****************************************************************
2846                                ;SUBROUTINE USED DURING TEST
2847                                ;;*****************************************************************
2848
2849                                ;USING DIAGNOSTIC MODE, RESET VOLUME VALID AND COMPOSITE ERROR.
2850                                ;VERIFY THAT VV, ERR, AND OCC ARE ZERO.
2851 030062                         100$:
2852 030062  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2853 030066  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2854 030074  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2855 030102  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
2856 030110  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
2857 030116  005037  001140                CLR     $GDDAT          ;SETUP FOR ERROR MSG
2858 030122  010037  001136                MOV     R0,$BDADR
2859 030126  062737  000012  001136        ADD     #RMDS,$BDADR
2860 030134  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2861 030142  042737  137777  001142        BIC     #^CERR,$BDDAT
2862 030150  001402                        BEQ     110$            ;BRANCH IF COMP ERROR ZERO
2863 030152  104143                        EMT     143
2864 030154  000447                        BR      140$            ;SKIP TEST
2865 030156                         110$:
     030156  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2866 030164  042737  177677  001142        BIC     #^CVV,$BDDAT
2867 030172  001402                        BEQ     120$            ;BRANCH IF VOLUME VALID ZERO
2868 030174  104135                        EMT     135
2869 030176  000436                        BR      140$            ;SKIP TEST
2870 030200                         120$:
     030200  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
2871 030206  042737  077777  001142        BIC     #^COCC,$BDDAT
2872 030214  001407                        BEQ     130$            ;BRANCH IF OCCUPIED ZERO
2873 030216  010037  001136                MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
2874 030222  062737  000024  001136        ADD     #RMMR1,$BDADR
2875 030230  104166                        EMT     166
```

```
2876 030232  000420                        BR      140$            ;SKIP TEST
2877
2878                                ;TO VERIFY THAT THE DECODE FLOP IS RESET, LOAD AN ILLEGAL FUNCTION
2879                                ;IN RMCS1 AND VERIFY THAT ILF DOES NOT SET.
2880 030234                         130$:
2881 030234  012760  000024  000000        MOV     #ILF24,RMCS1(R0)        ;LOAD RMCS1
2882 030242  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
2883 030250  042737  177776  001142        BIC     #^CILF,$BDDAT
2884 030256  001410                        BEQ     150$            ;BRANCH IF ILF IS ZERO
2885 030260  010037  001136                MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
2886 030264  062737  000014  001136        ADD     #RMER1,$BDADR
2887 030272  104167                        EMT     167
2888 030274  012716  030302        140$:   MOV     #200$,(SP)      ;DONT GO BACK TO TEST
2889
2890 030300  000207                150$:   RTS     PC              ;RETURN TO TEST OR EXIT TEST
2891
2892 030302                         200$:
2893
2894                                ;;*****************************************************************
                                    ;*TEST 53        SET/RESET VOLUME VALID TEST

                                    ;;*****************************************************************
     030302                         TST53:
     030302  000004                        SCOPE                   ;SCOPE CALL
     030304  000240                        NOP
     030306  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     030312  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     030316  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     030322  012737  000053  001226        MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2895
2896 030330  010037  001136                MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2897 030334  062737  000012  001136        ADD     #RMDS,$BDADR
2898 030342  012702  030546                MOV     #100$,R2        ;R2=TABLE POINTER
2899
2900                                ;INITIALIZE AND USE DIAGNOSTIC MODE TO RESET VOLUME VALID
2901 030346                         10$:
2902 030346  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2903 030352  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2904 030360  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2905 030366  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
2906 030374  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
2907 030402  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2908 030410  042737  177677  001142        BIC     #^CVV,$BDDAT
2909 030416  001403                        BEQ     20$             ;BRANCH IF VOLUME VALID ZERO
2910 030420  005037  001140                CLR     $GDDAT
2911 030424  104135                        EMT     135
2912
2913                                ;EXECUTE THE FUNCTION CODE IN THE TABLE
2914 030426                         20$:
2915 030426  111203                        MOVB    (R2),R3         ;GET FUNCTION CODE
2916 030430  042703  177701                BIC     #^CILF76,R3
2917 030434  052703  000001                BIS     #GO,R3
2918 030440  010360  000000                MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2919 030444  116204  000001                MOVB    1(R2),R4        ;GET CLOCK COUNT
2920 030450  042704  177400                BIC     #^C377,R4
2921 030454                         30$:
     030454  012760  141001  000024        MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
```

```
 2922 030462  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
 2923 030470  005304                          DEC     R4
 2924 030472  001370                          BNE     30$             ;ISSUE COCKS TIL R4 ZERO
 2925 030474  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 2926 030502  042737  177677  001142          BIC     #^CVV,$BDDAT
 2927 030510  001007                          BNE     40$             ;BRANCH IF VOLUME VALID SET
 2928 030512  010337  001174                  MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
 2929 030516  012737  000100  001140          MOV     #VV,$GDDAT
 2930 030524  104170                          EMT     170
 2931 030526  000406                          BR      50$
 2932
 2933                                 ;ADVANCE THE TABLE POINTER, EXIT IF DONE
 2934 030530                          40$:
 2935 030530  062702  000002                  ADD     #2,R2
 2936 030534  105762  000001                  TSTB    1(R2)
 2937 030540  100401                          BMI     50$             ;EXIT IF COUNT IS NEGATIVE
 2938 030542  000701                          BR      10$
 2939 030544  000403                  50$:    BR      200$            ;JUMP OVER TABLE
 2940
 2941                                 ;TABLE OF FUNCTION CODES AND CLOCK COUNTS
 2942 030546                          100$:
 2943 030546      020                         .BYTE   RIP             ;READ IN PRESET COMMAND
 2944 030547      003                         .BYTE   3
 2945
 2946 030550      022                         .BYTE   PAKACK          ;PACK ACKNOWLEDGE COMMAND
 2947 030551      003                         .BYTE   3
 2948
 2949 030552      000                         .BYTE                   ;END OF TABLE
 2950 030553      377                         .BYTE   -1
 2951
 2952 030554                          200$:                           ;END OF TEST
 2953
 2954                                 ;;*****************************************************************
                                      ;*TEST 54        ILLEGAL FUNCTION TEST
                                      ;;*****************************************************************
      030554                          TST54:
      030554  000004                         SCOPE                   ;SCOPE CALL
      030556  000240                         NOP
      030560  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      030564  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      030570  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      030574  012737  000054  001226          MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2955
 2956 030602  005002                          CLR     R2              ;INITIALIZE FUNCTION CODE VALUE
 2957 030604                          10$:
 2958 030604  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      030610  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
      030612  104000                          EMT
 2959 030614  000460                          BR      50$             ;SKIP TEST IF ERROR
 2960 030616  012704  000002          20$:    MOV     #2,R4           ;R4=CLOCK COUNT
 2961
 2962                                 ;EXECUTE THE TEST FUNCTION CODE AND VERIFY ILF
 2963 030622  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 2964 030630  010203                          MOV     R2,R3           ;SETUP FUNCTION CODE IN R3
 2965 030632  052703  000001                  BIS     #GO,R3
 2966 030636  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
```

```
                                        30$:
030642                                          
030642  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
030650  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
030656  005304                                  DEC     R4
030660  001370                                  BNE     30$
030662  016037  000014  001142                  MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
030670  042737  177776  001142                  BIC     #^CILF,$BDDAT   ;SETUP ACTUAL ILF STATUS
030676  016237  062752  001140                  MOV     FNCDTB(R2),$GDDAT  ;GET EXPECTED ILF STATUS
030704  042737  177776  001140                  BIC     #^CILF,$GDDAT
030712  023737  001140  001142                  CMP     $GDDAT,$BDDAT
030720  001410                                  BEQ     40$                     ;BRANCH IF ILF IS OK
030722  010037  001136                          MOV     R0,$BDADR               ;SETUP FOR ERROR MSG
030726  062737  000014  001136                  ADD     #RMER1,$BDADR
030734  010237  001174                          MOV     R2,$TMP0
030740  104171                                  EMT     171

                                        ;ADVANCE TO THE NEXT FUNCTION CODE AND REPEAT TEST
                                        40$:
030742                                          
030742  062702  000002                          ADD     #2,R2
030746  022702  000076                          CMP     #ILF76,R2
030752  103401                                  BLO     50$
030754  000713                                  BR      10$
030756                                  50$:                                    ;END OF TEST

                                        ;;*******************************************************************
                                        ;*TEST 55       OCCUPIED TEST

                                        ;;*******************************************************************
030756                                  TST55:
030756  000004                                  SCOPE                           ;SCOPE CALL
030760  000240                                  NOP
030762  012706  001100                          MOV     #STACK,SP               ;LOAD THE STACK POINTER
030766  013700  001276                          MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
030772  013701  001462                          MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
030776  012737  000055  001226                  MOV     #55,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

031004  005002                                  CLR     R2                              ;INITIALIZE FUNCTION CODE

                                        ;GET THE DEVICE READY
031006                                  10$:
031006  004737  055370                          JSR     PC,SETVV                ;GO SET VOLUME VALID
031012  000402                                  BR      20$                     ;BRANCH TO 20$ IF NO ERROR
031014  104000                                  EMT
031016  000464                                  BR      50$

                                        ;ENABLE DEBUG CLOCK AND LOAD THE FUNCTION CODE
031020                                  20$:
031020  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
031026  010203                                  MOV     R2,R3                           ;ASSEMBLE FUNCTION CODE AND
031030  052703  000001                          BIS     #GO,R3                          ;GO BIT IN R3
031034  010360  000000                          MOV     R3,RMCS1(R0)    ;LOAD RMCS1
031040  012704  000002                          MOV     #2,R4                           ;R4=CLOCK COUNT

                                        ;STEP THE DEBUG CLOCK UNTIL SET PULSE IS ACTIVE
031044                                  30$:
031044  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
031052  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
3011 031060  005304                        DEC     R4
3012 031062  001370                        BNE     30$                     ;ISSUE NEXT CLOCK TIL R4 ZERO
3013
3014                                ;VERIFY OCCUPIED STATUS
3015 031064  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
3016 031072  042737  077777  001142        BIC     #^COCC,$BDDAT
3017 031100  005037  001140                CLR     $GDDAT          ;GENERATE OCC FROM AOE
3018 031104  032762  001000  062752        BIT     #AOE,FNCDTB(R2)
3019 031112  001403                        BEQ     35$
3020 031114  012737  100000  001140        MOV     #OCC,$GDDAT
3021 031122  023737  001140  001142 35$:   CMP     $GDDAT,$BDDAT
3022 031130  001411                        BEQ     40$                     ;BRANCH IF OCC IS OK
3023 031132  010237  001174                MOV     R2,$TMP0                ;SAVE FUNCTION CODE
3024 031136  010037  001136                MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS
3025 031142  062737  000024  001136        ADD     #RMMR1,$BDADR
3026 031150  104173                        EMT     173
3027 031152  000406                        BR      50$
3028
3029                                ;ADVANCE TO NEXT FUNCTIONCODE, EXIT IF DONE
3030 031154                         40$:
3031 031154  062702  000002                ADD     #2,R2
3032 031160  022702  000076                CMP     #ILF76,R2
3033 031164  103401                        BLO     50$                     ;EXIT IF DONE
3034 031166  000707                        BR      10$
3035
3036 031170                         50$:                                    ;END OF TEST
3037
3038                                ;;**********************************************************************
                                    ;*TEST 56        READ IN PRESET TEST
                                    ;;**********************************************************************
     031170                         TST56:
     031170  000004                        SCOPE                   ;SCOPE CALL
     031172  000240                        NOP
     031174  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     031200  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     031204  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     031210  012737  000056  001226        MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3039
3040                                ;CLEAR AND ENABLE DEBUG CLOCK - LEAVE VOLUME VALID RESET
3041 031216  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3042 031222  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3043 031230  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3044 031236  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
3045 031244  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
3046
3047                                ;LOAD ALL ONES IN RMDA, RMDC AND RMOF
3048 031252  012760  177777  000006        MOV     #-1,RMDA(R0)    ;LOAD RMDA
3049 031260  012760  177777  000034        MOV     #-1,RMDC(R0)    ;LOAD RMDC
3050 031266  012760  177777  000032        MOV     #-1,RMOF(R0)    ;LOAD RMOF
3051
3052                                ;LOAD READ IN PRESET COMMAND AND STEP THE CLOCK TILL SET PULSE
3053 031274  012760  000021  000000        MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
3054 031302  012702  000003                MOV     #3,R2                   ;R2=CLOCK COUNT
3055 031306                         10$:
3056 031306  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3057 031314  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
 3058 031322  005302                        DEC     R2
 3059 031324  001370                        BNE     10$                     ;ISSUE 3 CLOCKS
 3060
 3061                                ;SEE IF RMDA OR RMDC OR RMOF IS ZERO
 3062 031326  016002  000006                MOV     RMDA(R0),R2     ;STORE RMDA AT R2
 3063 031332  005702                        TST     R2
 3064 031334  001413                        BEQ     20$                     ;BRANCH IF RMDA IS ZERO
 3065 031336  016002  000034                MOV     RMDC(R0),R2     ;STORE RMDC AT R2
 3066 031342  042702  176000                BIC     #XNUDC,R2               ;CLEAR UNUSED BITS
 3067 031346  001406                        BEQ     20$                     ;BRANCH IF RMDC IS ZERO
 3068 031350  016002  000032                MOV     RMOF(R0),R2     ;STORE RMOF AT R2
 3069 031354  042702  161577                BIC     #XNUOF,R2               ;CLEAR UNUSED BITS
 3070 031360  001401                        BEQ     20$                     ;BRANCH IF RMOF IS ZERO
 3071
 3072                                ;READ IN PRESET COMMAND DIDNT CLEAR ANY OF THE 3 REGISTERS
 3073 031362  104174                        EMT     174
 3074
 3075 031364                        20$:                                    ;END OF TEST
 3076
 3077                                ;;*****************************************************************
                                     ;*TEST 57        RIP/RMOF TEST
                                     ;;*****************************************************************
      031364                         TST57:
      031364  000004                        SCOPE                   ;SCOPE CALL
      031366  000240                        NOP
      031370  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
      031374  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      031400  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      031404  012737  000057  001226        MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 3078
 3079 031412  010037  001136                MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS AND
 3080 031416  062737  000032  001136        ADD     #RMOF,$BDADR
 3081 031424  005037  001140                CLR     $GDDAT                  ;EXPECTED RMOF
 3082
 3083                                ;INITIALIZE AND SET BITS IN RMOF
 3084 031430  004737  055614                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 3085 031434  012760  000001  000024        MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 3086 031442  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 3087 031450  012760  000000  000C14        MOV     #0,RMER1(R0)    ;LOAD RMER1
 3088 031456  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
 3089 031464  012760  177777  000032        MOV     #-1,RMOF(R0)    ;LOAD RMOF
 3090
 3091                                ;EXECUTE A READ IN PRESET IN DIAGNOSTIC MODE TILL SET PULSE
 3092 031472  012760  000021  000000        MOV     #RIP'GO,RMCS1(R0)       ;LOAD RMCS1
 3093 031500  012702  000003                MOV     #3,R2                   ;R2=CLOCK COUNT
 3094 031504                        10$:
      031504  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 3095 031512  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 3096 031520  005302                        DEC     R2
 3097 031522  001370                        BNE     10$                     ;ISSUE 3 CLOCKS
 3098
 3099                                ;VERIFY THAT RMOF IS ZERO
 3100 031524  016037  000032  001142        MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
 3101 031532  042737  161577  001142        BIC     #XNUOF,$BDDAT
 3102 031540  001401                        BEQ     20$                     ;BRANCH IF RMOF IS ZERO
 3103 031542  104175                        EMT     175
```

```
3104
3105 031544                                    20$:                                    ;END OF TEST
3106
3107                                           ;;*****************************************************************
                                               ;*TEST 60       RMDA/RMDC/RIP TEST
                                               ;;*****************************************************************
     031544                                    TST60:
     031544  000004                                  SCOPE                             ;SCOPE CALL
     031546  000240                                  NOP
     031550  012706  001100                          MOV     #STACK,SP                 ;LOAD THE STACK POINTER
     031554  013700  001276                          MOV     $BASE,R0                  ;R0 = UNIBUS ADDRESS
     031560  013701  001462                          MOV     TSTQUE,R1                 ;R1 = POINTER TO DEVICE
     031564  012737  000060  001226                  MOV     #60,$TESTN                ;;SET TEST NUMBER IN APT MAIL BOX
3108
3109 031572  005037  001140                          CLR     $GDDAT
3110
3111                                           ;CLEAR, ENABLE DEBUG CLOCK, THEN PRESET RMDA AND RMDC
3112 031576  004737  055614                          JSR     PC,CNTCLR                 ;GO CLEAR CONTROLLER
3113 031602  012760  000001  000024                  MOV     #DMD,RMMR1(R0)            ;LOAD RMMR1
3114 031610  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0)   ;LOAD RMMR1
3115 031616  012760  000000  000014                  MOV     #0,RMER1(R0)              ;LOAD RMER1
3116 031624  012760  000000  000014                  MOV     #0,RMER1(R0)              ;LOAD RMER1
3117 031632  012760  177777  000006                  MOV     #-1,RMDA(R0)              ;LOAD RMDA
3118 031640  012760  177777  000034                  MOV     #-1,RMDC(R0)              ;LOAD RMDC
3119
3120                                           ;EXECUTE READ IN PRESET TILL SET PULSE
3121 031646  012760  000021  000000                  MOV     #RIP!GO,RMCS1(R0)         ;LOAD RMCS1
3122 031654  012702  000003                          MOV     #3,R2
3123 031660                                    10$:
     031660  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)     ;LOAD RMMR1
3124 031666  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0)   ;LOAD RMMR1
3125 031674  005302                                  DEC     R2
3126 031676  001370                                  BNE     10$                       ;ISSUE 3 CLOCKS
3127
3128                                           ;VERIFY RMDA IS ZERO
3129 031700  016037  000006  001142                  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
3130 031706  005737  001142                          TST     $BDDAT
3131 031712  001406                                  BEQ     20$                       ;BRANCH IF RMDA RESET
3132 031714  010037  001136                          MOV     R0,$BDADR
3133 031720  062737  000006  001136                  ADD     #RMDA,$BDADR
3134 031726  104176                                  EMT     176
3135
3136                                           ;VERIFY RMDC IS ZERO
3137 031730                                    20$:
3138 031730  016037  000034  001142                  MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
3139 031736  042737  176000  001142                  BIC     #XNUDC,$BDDAT
3140 031744  001406                                  BEQ     30$                       ;BRANCH IF RMDC RESET
3141 031746  010037  001136                          MOV     R0,$BDADR
3142 031752  062737  000034  001136                  ADD     #RMDC,$BDADR
3143 031760  104177                                  EMT     177
3144
3145 031762                                    30$:                                    ;END OF TEST
3146
3147                                           ;;*****************************************************************
                                               ;*TEST 61       OFFSET COMMAND TEST
```

```
                                ;;*****************************************************************
                                TST61:
        031762
        031762  000004                  SCOPE                   ;SCOPE CALL
        031764  000240                  NOP
        031766  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        031772  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        031776  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        032002  012737  000061  001226  MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  3148
  3149  032010  010037  001136          MOV     R0,$BDADR
  3150  032014  062737  000012  001136  ADD     #RMDS,$BDADR
  3151  032022  012737  000001  001140  MOV     #OM,$GDDAT
  3152
  3153  032030  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
        032034  000402                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
        032036  104000                  EMT
  3154  032040  000433                  BR      40$
  3155  032042                  10$:
        032042  012760  000000  000034  MOV     #0,RMDC(R0)     ;LOAD RMDC
  3156
  3157                          ;ENABLE DEBUG CLOCK AND EXECUTE OFFSET COMMAND
  3158  032050  012760  041001  000024  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  3159  032056  012760  000015  000000  MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
  3160  032064  012702  000002          MOV     #2,R2                   ;R2=CLOCK COUNT
  3161  032070                  20$:
        032070  012760  141001  000024  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
  3162  032076  012760  041001  000024  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  3163  032104  005302                  DEC     R2
  3164  032106  001370                  BNE     20$                     ;ISSUE 2 CLOCKS
  3165
  3166                          ;VERIFY THAT OFFSET MODE IS SET
  3167  032110  016037  000012  001142  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
  3168  032116  042737  177776  001142  BIC     #^COM,$BDDAT
  3169  032124  001001                  BNE     40$                     ;BRANCH IF OM IS SET
  3170  032126  104200                  EMT     200
  3171  032130                  40$:                            ;END OF TEST
  3172
  3173                          ;;*****************************************************************
                                ;*TEST 62        RETURN TO CENTER TEST
                                ;;*****************************************************************
        032130                  TST62:
        032130  000004                  SCOPE                   ;SCOPE CALL
        032132  000240                  NOP
        032134  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        032140  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        032144  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        032150  012737  000062  001226  MOV     #62,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  3174
  3175  032156  010037  001136          MOV     R0,$BDADR
  3176  032162  062737  000012  001136  ADD     #RMDS,$BDADR
  3177
  3178  032170  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
        032174  000402                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
        032176  104000                  EMT
  3179  032200  000465                  BR      60$
  3180
```

```
3181                                        ;SET OFFSET DIRECTION AND OFFSET
3182 032202                                 10$:
3183 032202  012760  000200  000032                MOV     #OFD,RMOF(R0)   ;LOAD RMOF
3184 032210  004737  055512                        JSR     PC,SETOM        ;GO SET OFFSET MODE
     032214  000401                                BR      20$             ;BRANCH TO 20$ IF NO ERROR
     032216  104000                                EMT
3185
3186                                        ;ENABLE DEBUG CLOCK AND EXECUTE RETURN TO CENTER COMMAND
3187 032220                                 20$:
3188 032220  004737  055370                        JSR     PC,SETVV        ;GO SET VOLUME VALID
     032224  000402                                BR      30$             ;BRANCH TO 30$ IF NO ERROR
     032226  104000                                EMT
3189 032230  000451                                BR      60$
3190 032232                                 30$:
3191 032232  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3192 032240  012760  000017  000000                MOV     #RTC!GO,RMCS1(R0)       ;LOAD RMCS1
3193 032246  012702  000002                        MOV     #2,R2
3194 032252                                 40$:
     032252  012760  141001  000024                MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3195 032260  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3196 032266  005302                                DEC     R2
3197 032270  001370                                BNE     40$                     ;ISSUE 2 CLOCKS
3198
3199                                        ;VERIFY THAT OFFSET MODE IS RESET
3200 032272  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3201 032300  042737  177776  001142                BIC     #^COM,$BDDAT
3202 032306  001403                                BEQ     50$                     ;BRANCH IF OFFSET MODE RESET
3203 032310  005037  001140                        CLR     $GDDAT
3204 032314  104201                                EMT     201
3205
3206                                        ;VERIFY THAT OFFSET DIRECTION IS RESET
3207 032316                                 50$:
3208 032316  016037  000032  001142                MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
3209 032324  042737  177577  001142                BIC     #^COFD,$BDDAT
3210 032332  001410                                BEQ     60$                     ;BRANCH IF OFD IS RESET
3211 032334  005037  001140                        CLR     $GDDAT
3212 032340  010037  001136                        MOV     R0,$BDADR
3213 032344  062737  000032  001136                ADD     #RMOF,$BDADR
3214 032352  104202                                EMT     202
3215 032354                                 60$:                                    ;END OF TEST
3216
3217                                        ;;*****************************************************************
                                            ;*TEST 63        RMDC CLEAR OFFSET TEST

                                            ;;*****************************************************************
     032354                                 TST63:
     032354  000004                                SCOPE                   ;SCOPE CALL
     032356  000240                                NOP
     032360  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
     032364  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     032370  013701  001462                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     032374  012737  000063  001226                MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3218
3219 032402  010037  001136                        MOV     R0,$BDADR
3220 032406  062737  000012  001136                ADD     #RMDS,$BDADR
3221
3222 032414  004737  055370                        JSR     PC,SETVV        ;GO SET VOLUME VALID
```

```
        032420  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
        032422  104000                          EMT
  3223  032424  000421                          BR      40$                     ;SKIP REST OF TEST
  3224  032426                          10$:
  3225  032426  004737  055512                  JSR     PC,SETOM        ;GO SET OFFSET MODE
        032432  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
        032434  104000                          EMT
  3226
  3227                                  ;WRITE THE DESIRED CYLINDER REGISTER AND VERIFY THAT OFFSET IS ZERO
  3228  032436                          20$:
  3229  032436  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
  3230  032444  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
  3231  032452  042737  177776  001142          BIC     #^COM,$BDDAT
  3232  032460  001403                          BEQ     40$
  3233  032462  005037  001140                  CLR     $GDDAT
  3234  032466  104203                          EMT     203
  3235
  3236  032470                          40$:                            ;END OF TEST
  3237
  3238                                  ;;*******************************************************************
                                        ;*TEST 64        EBL CLEAR OFFSET TEST

                                        ;;*******************************************************************
        032470                          TST64:
        032470  000004                          SCOPE                   ;SCOPE CALL
        032472  000240                          NOP
        032474  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        032500  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        032504  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        032510  012737  000064  001226          MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  3239
  3240  032516  013737  001330  001414          MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
  3241  032524  112737  000037  001414          MOVB    #31.,RMDAO      ;LAST SECTOR
  3242  032532  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER FOR ERROR MSG
  3243  032536  062737  000012  001136          ADD     #RMDS,$BDADR
  3244
  3245  032544  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        032550  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
        032552  104000                          EMT
  3246  032554  000440                          BR      30$                     ;SKIP REST OF TEST IF ERROR
  3247  032556                          10$:
  3248  032556  004737  055512                  JSR     PC,SETOM        ;GO SET OFFSET MODE
        032562  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
        032564  104000                          EMT
  3249  032566                          20$:
  3250  032566  012760  010000  000032          MOV     #FMT16,RMOF(R0) ;LOAD RMOF
  3251  032574  013760  001414  000006          MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
  3252
  3253                                  ;FORCE END OF BLOCK AND VERIFY THAT OFFSET MODE IS CLEARED
  3254  032602  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  3255  032610  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
  3256  032616  012760  061001  000024          MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
  3257  032624  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  3258  032632  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
  3259  032640  042737  177776  001142          BIC     #^COM,$BDDAT
  3260  032646  001403                          BEQ     30$                     ;BRANCH IF OFFSET IS ZERO
  3261  032650  005037  001140                  CLR     $GDDAT
```

```
3262 032654  104204                    EMT     204
3263 032656                        30$:                                ;END OF TEST
3264
3265                               ;;*****************************************************************
                                   ;*TEST 65       RUN AND GO TEST

                                   ;;*****************************************************************
     032656                        TST65:
     032656  000004                        SCOPE                       ;SCOPE CALL
     032660  000240                        NOP
     032662  012706  001100                MOV     #STACK,SP           ;LOAD THE STACK POINTER
     032666  013700  001276                MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
     032672  013701  001462                MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
     032676  012737  000065  001226        MOV     #65,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
3266
3267 032704  005037  001140                CLR     $GDDAT                      ;INITIALIZE EXPECTED RESULT
3268 032710  005002                        CLR     R2                          ;INTIALIZE FUNCTION CODE
3269 032712  010037  001136                MOV     R0,$BDADR
3270 032716  062737  000024  001136        ADD     #RMMR1,$BDADR
3271
3272                               ;CLEAR THE MASSBUS AND ENABLE DEBUG CLOCK
3273 032724                        10$:
3274 032724  004737  055614                JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
3275 032730  012760  000001  000024        MOV     #DMD,RMMR1(R0)      ;LOAD RMMR1
3276 032736  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
3277 032744  012760  000000  000014        MOV     #0,RMER1(R0)        ;LOAD RMER1
3278 032752  012760  000000  000042        MOV     #0,RMER2(R0)        ;LOAD RMER2
3279
3280                               ;LOAD THE FUNCTION CODE AND VERIFY RUN AND GO FLOP
3281 032760  010203                        MOV     R2,R3                       ;ASSEMBLE FUNCTION CODE AND GO
3282 032762  052703  000001                BIS     #GO,R3
3283 032766  012737  000200  001530        MOV     #200,WATCH          ;SET WATCHDOG TIMER VALUE
     032774  004777  146532                JSR     PC,@CLOCK           ;START THE CLOCK
3284 033000  010360  000000                MOV     R3,RMCS1(R0)        ;LOAD RMCS1
3285 033004                        15$:
     033004  016037  000024  001142        MOV     RMMR1(R0),$BDDAT            ;STORE RMMR1 AT $BDDAT
3286 033012  042737  137777  001142        BIC     #^CRG,$BDDAT
3287 033020  023737  001140  001142        CMP     $GDDAT,$BDDAT
3288 033026  001411                        BEQ     20$                         ;BRANCH IF RUN AND GO FLOP OK
3289 033030  005737  001530                TST     WATCH               ;TAKE ANOTHER SAMPLE IF CLOCK NOT ZERO
3290 033034  001363                        BNE     15$
3291 033036  004777  146472                JSR     PC,@STOP            ;STOP THE CLOCK
3292 033042  010237  001174                MOV     R2,$TMP0                    ;SAVE FUNCTION CODE FOR MSG
3293 033046  104205                        EMT     205
3294 033050  000416                        BR      40$                         ;SKIP REST OF
3295 033052                        20$:
3296 033052  004777  146456                JSR     PC,@STOP            ;STOP THE CLOCK
3297
3298                               ;ADVNACE TO NEXT FUNCTION CODE - EXIT IF DONE
3299 033056  062702  000002                ADD     #2,R2
3300 033062  022702  000076                CMP     #ILF76,R2
3301 033066  103407                        BLO     40$                         ;EXIT IF DONE
3302 033070  020227  000050                CMP     R2,#WCD                     ;CHANGE EXPECTED RESULTI IF
3303 033074  103403                        BLO     30$                         ;DATA COMMAND
3304 033076  012737  040000  001140        MOV     #RG,$GDDAT
3305 033104  000707                30$:    BR      10$                         ;REPEAT TEST
3306
```

```
 3307 033106                            40$:                                    ;END OF TEST
 3308
 3309                                   ;;*****************************************************************
                                        ;*TEST 66       SET IAE TEST
                                        ;;*****************************************************************
      033106                            TST66:
      033106  000004                            SCOPE                           ;SCOPE CALL
      033110  000240                            NOP
      033112  012706  001100                    MOV     #STACK,SP               ;LOAD THE STACK POINTER
      033116  013700  001276                    MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
      033122  013701  001462                    MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
      033126  012737  000066  001226            MOV     #66,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
 3310
 3311 033134  012702  033302                    MOV     #100$,R2                        ;R2=TABLE POINTER
 3312 033140                            10$:
 3313 033140  004737  055370                    JSR     PC,SETVV                ;GO SET VOLUME VALID
      033144  000402                            BR      20$                     ;BRANCH TO 20$ IF NO ERROR
      033146  104000                            EMT
 3314 033150  000453                            BR      50$                             ;SKIP REST OF TEST
 3315
 3316                                   ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT TO 18
 3317 033152                            20$:
 3318 033152  012760  177777  000006            MOV     #-1,RMDA(R0)            ;LOAD RMDA
 3319 033160  012760  177777  000034            MOV     #-1,RMDC(R0)            ;LOAD RMDC
 3320 033166  012760  000000  000032            MOV     #0,RMOF(R0)             ;LOAD RMOF
 3321
 3322                                   ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCSI WITH GO ON
 3323 033174  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 3324 033202  111203                            MOVB    (R2),R3
 3325 033204  042703  177701                    BIC     #^CILF76,R3
 3326 033210  052703  000001                    BIS     #GO,R3
 3327 033214  010360  000000                    MOV     R3,RMCS1(R0)    ;LOAD RMCS1
 3328 033220  116204  000001                    MOVB    1(R2),R4                ;GET CLOCK COUNT
 3329 033224  042704  177400                    BIC     #^C377,R4
 3330
 3331                                   ;CLOCK THE COMMAND SEQUENCER
 3332 033230                            30$:
 3333 033230  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 3334 033236  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 3335 033244  005304                            DEC     R4
 3336 033246  001370                            BNE     30$
 3337
 3338                                   ;SEE IF IAE HAS SET
 3339 033250  016004  000014                    MOV     RMER1(R0),R4    ;STORE RMER1 AT R4
 3340 033254  042704  175777                    BIC     #^CIAE,R4
 3341 033260  001007                            BNE     50$                     ;BRANCH IF IAE SET
 3342
 3343                                   ;IAE DID NOT SET - TRY ANOTHER FUNCTION CODE
 3344 033262  062702  000002                    ADD     #2,R2
 3345 033266  105762  000001                    TSTB    1(R2)
 3346 033272  100401                            BMI     40$                     ;BRANCH IF ALL CODES TRIED
 3347 033274  000721                            BR      10$
 3348
 3349                                   ;CANNOT SET IAE WITH ANY COMBINATION OF ADDRESS AND FUNCTION CODE
 3350 033276                            40$:
 3351 033276  104206                            EMT     206
```

```
3352
3353 033300  000411                50$:    BR      200$                    ;JUMP OVER TABLE
3354
3355                                ;TABLE OF FUNCTION CODES AND CLOCK COUNTS FOR TEST
3356 033302                         100$:
3357 033302       030                       .BYTE   SEARCH                  ;SEARCH COMMAND
3358 033303       002                       .BYTE   2
3359
3360 033304       004                       .BYTE   SEEK                    ;SEEK COMMAND
3361 033305       002                       .BYTE   2
3362
3363 033306       062                       .BYTE   WH                      ;WRITE HEADER COMMAND
3364 033307       002                       .BYTE   2
3365
3366 033310       052                       .BYTE   WCH                     ;WRITE CHECK HEADER COMMAND
3367 033311       002                       .BYTE   2
3368
3369 033312       072                       .BYTE   RH                      ;READ HEADER COMMAND
3370 033313       002                       .BYTE   2
3371
3372 033314       060                       .BYTE   WD                      ;WRITE DATA COMMAND
3373 033315       002                       .BYTE   2
3374
3375 033316       050                       .BYTE   WCD                     ;WRITE CHECK DATA COMMAND
3376 033317       002                       .BYTE   2
3377
3378 033320       070                       .BYTE   RD                      ;READ DATA COMMAND
3379 033321       002                       .BYTE   2
3380
3381 033322       000                       .BYTE                           ;END OF TABLE
3382 033323       377                       .BYTE   -1
3383
3384 033324                         200$:                                   ;END OF TEST
3385
3386                                ;;*****************************************************************
                                    ;*TEST 67        SEARCH, SEEK, READ, WRITE TEST

                                    ;;*****************************************************************
     033324                         TST67:
     033324  000004                        SCOPE                   ;SCOPE CALL
     033326  000240                        NOP
     033330  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     033334  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     033340  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     033344  012737  000067  001226        MOV     #67,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3387
3388 033352  005002                       CLR     R2                      ;INITIALIZE FUNCTION CODE
3389 033354                         10$:
3390 033354  004737  055370              JSR     PC,SETVV        ;GO SET VOLUME VALID
     033360  000402                        BR      20$             ;BRANCH TO 20$ IF NO ERROR
     033362  104000                        EMT
3391 033364  000472                        BR      50$
3392
3393                                ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT
3394                                ;TO 18 BIT MODE
3395 033366                         20$:
3396 033366  012760  177777  000006        MOV     #-1,RMDA(R0)    ;LOAD RMDA
```

```
3397 033374  012760  177777  000034          MOV     #-1,RMDC(R0)     ;LOAD RMDC
3398 033402  012760  000000  000032          MOV     #0,RMOF(R0)      ;LOAD RMOF
3399
3400                                  ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCS1 WITH GO ON
3401 033410  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3402 033416  010203                          MOV     R2,R3                   ;ASSEMBLE CODE AND GO
3403 033420  052703  000001                  BIS     #GO,R3
3404 033424  010360  000000                  MOV     R3,RMCS1(R0)     ;LOAD RMCS1
3405
3406                                  ;CLOCK THE COMMAND SEQUENCER TO SET PULSE
3407 033430  012704  000002                  MOV     #2,R4
3408 033434                           30$:
     033434  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)     ;LOAD RMMR1
3409 033442  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3410 033450  005304                          DEC     R4
3411 033452  001370                          BNE     30$
3412
3413                                  ;VERIFY IAE ACCORDING TO FUNCTION CODE TABLE
3414 033454  016037  000014  001142          MOV     RMER1(R0),$BDDAT         ;STORE RMER1 AT $BDDAT
3415 033462  042737  175777  001142          BIC     #^CIAE,$BDDAT
3416 033470  016237  062752  001140          MOV     FNCDTB(R2),$GDDAT        ;ASSEMBLE EXPECTED IAE
3417 033476  042737  175777  001140          BIC     #^CIAE,$GDDAT
3418 033504  023737  001140  001142          CMP     $GDDAT,$BDDAT
3419 033512  001411                          BEQ     40$                      ;BRANCH IF IAE OK
3420 033514  010037  001136                  MOV     R0,$BDADR                ;SET UP ERROR MSG
3421 033520  062737  000014  001136          ADD     #RMER1,$BDADR
3422 033526  010237  001174                  MOV     R2,$TMP0
3423 033532  104207                          EMT     207
3424 033534  000406                          BR      50$                      ;SKIP REST OF TEST
3425
3426                                  ;ADVANCE TO NEXT FUNCTION CODE - EXIT IF DONE
3427 033536                           40$:
3428 033536  062702  000002                  ADD     #2,R2
3429 033542  023702  000076                  CMP     ILF76,R2
3430 033546  103401                          BLO     50$
3431 033550  000701                          BR      10$
3432 033552                           50$:                                     ;END OF TEST
3433
3434                                  ;;*****************************************************************
                                      ;*TEST 70        INVALID TRACK/SECTOR TEST

                                      ;;*****************************************************************
     033552                           TST70:
     033552  000004                          SCOPE                    ;SCOPE CALL
     033554  000240                          NOP
     033556  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
     033562  013700  001276                  MOV     $BASE,R0         ;R0 = UNIBUS ADDRESS
     033566  013701  001462                  MOV     TSTQUE,R1        ;R1 = POINTER TO DEVICE
     033572  012737  000070  001226          MOV     #70,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3435
3436 033600  013737  001330  001414          MOV     LSTRK,RMDAO              ;INITIALIZE TRACK ADDRESS
3437 033606  105237  001415                  INCB    RMDAO+1                  ;SETUP FIRST INVALID ADDRESS
3438 033612                           10$:
3439 033612  004737  055370                  JSR     PC,SETVV         ;GO SET VOLUME VALID
     033616  000402                          BR      20$              ;BRANCH TO 20$ IF NO ERROR
     033620  104000                          EMT
3440 033622  000477                          BR      100$                     ;SKIP REST OF TEST
```

```
                                        ;CLEAR DESIRED CYLINDER, LOAD INVALID TRACK ADDRESS, AND SET
                                        ;18 BIT FORMAT
033624                                  20$:
033624  012760  000000  000034                MOV     #0,RMDC(R0)         ;LOAD RMDC
033632  013760  001414  000006                MOV     RMDA0,RMDA(R0)      ;LOAD RMDA
033640  012760  000000  000032                MOV     #0,RMOF(R0)         ;LOAD RMOF

                                        ;EXECUTE A SEARCH COMMAND TO WHERE 'SET PULSE' IS ACTIVE
033646  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
033654  012760  000031  000000                MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
033662  012703  000002                        MOV     #2,R3
033666                                  30$:
033666  012760  141001  000024                MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
033674  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
033702  005303                                DEC     R3
033704  001370                                BNE     30$                     ;ISSUE 2 CLOCKS

                                        ;VERIFY IAE IS SET
033706  016037  000014  001142                MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
033714  042737  175777  001142                BIC     #^CIAE,$BDDAT
033722  001015                                BNE     40$                     ;BRANCH IF IAE IS ON
033724  012737  002000  001140                MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
033732  010037  001136                        MOV     R0,$BDADR
033736  062737  000014  001136                ADD     #RMER1,$BDADR
033744  013737  001414  001174                MOV     RMDA0,$TMP0
033752  104210                                EMT     210
033754  000422                                BR      100$

                                        ;ADVANCE TO NEXT RMDA ADDRESS
033756                                  40$:
033756  105737  001415                        TSTB    RMDA0+1                 ;TESTING INVALID SECTORS ?
033762  001411                                BEQ     50$                     ;YES !!
033764  105237  001415                        INCB    RMDA0+1                 ;INCREMENT TRACK ADDRESS
033770  123727  001415  000200                CMPB    RMDA0+1,#128.           ;DONE ?
033776  101705                                BLOS    10$                     ;NO, TEST NEXT TRACK ADDRESS
034000  012737  000035  001414                MOV     #29.,RMDA0              ;LOAD LOAD SECTOR ADDRESS AND
                                                                                ;TRACK 0.
034006  005237  001414                50$:    INC     RMDA0                   ;INCREMENT SECTOR ADDRESS
034012  123727  001414  000200                CMPB    RMDA0,#128.             ;DONE ?
034020  101674                                BLOS    10$                     ;NO, TEST NEXT SECTOR ADDRESS
034022                                  100$:

                                        ;;*************************************************************************
                                        ;*TEST 71        INVALID CYLINDER TEST

                                        ;;*************************************************************************
034022                                  TST71:
034022  000004                                SCOPE                   ;SCOPE CALL
034024  000240                                NOP
034026  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
034032  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
034036  013701  001462                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
034042  012737  000071  001226                MOV     #71,$TESTN      ;;SET TEST NUMBER IN API MAIL BOX

034050  012737  001467  001442                MOV     #823.,RMDC0             ;SET FIRST INVALID CYLINDER
034056                                  10$:
```

```
3488 034056  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
     034062  000402                  BR      20$             ;BRANCH TO 20$ IF NO ERROR
     034064  104000                  EMT
3489 034066  000460                  BR      50$                     ;SKIP IF ERROR
3490 034070                  20$:
3491 034070  013760  001442  000034  MOV     RMDCO,RMDC(R0)  ;LOAD RMDC
3492 034076  012760  000000  000006  MOV     #0,RMDA(R0)     ;LOAD RMDA
3493
3494                         ;ENABLE DEBUG CLOCK AND EXECUTE SEARCH COMMAND
3495 034104  012760  041001  000024  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3496 034112  012760  000031  000000  MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
3497 034120  012703  000002          MOV     #2,R3
3498 034124                  30$:
     034124  012760  141001  000024  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3499 034132  012760  041001  000024  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3500 034140  005303                  DEC     R3
3501 034142  001370                  BNE     30$                     ;ISSUE 2 CLOCKS
3502
3503                         ;VERIFY IAE IS SET
3504 034144  016037  000014  001142  MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3505 034152  042737  175777  001142  BIC     #^CIAE,$BDDAT
3506 034160  001015                  BNE     40$                     ;BRANCH IF IAE IS SET
3507 034162  012737  002000  001140  MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
3508 034170  010037  001136          MOV     R0,$BDADR
3509 034174  062737  000014  001136  ADD     #RMER1,$BDADR
3510 034202  013737  001442  001174  MOV     RMDCO,$TMP0
3511 034210  104211                  EMT     211
3512 034212  000406                  BR      50$
3513
3514                         ;ADVANCE CYLINDER ADDRESS
3515 034214                  40$:
3516 034214  005237  001442          INC     RMDCO           ;INCREMENT CYLINDER ADDRESS
3517 034220  023727  001442  002000  CMP     RMDCO,#1024.    ;DONE ?
3518 034226  103713                  BLO     10$             ;NO, TEST AGAIN
3519 034230                  50$:
3520
3521                         ;;**********************************  ******************************
                             ;*TEST 72        SET AOE TEST

                             ;;*****************************************************************
     034230                  TST72:
     034230  000004                  SCOPE                   ;SCOPE CALL
     034232  000240                  NOP
     034234  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
     034240  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     034244  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     034250  012737  000072  001226  MOV     #72,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3522
3523 034256  005037  001440          CLR     RMOF0           ;18 BIT FORMAT MODE
3524 034262  013737  001330  001414  MOV     LSTRK,RMDA0     ;SETUP LAST TRACK AND
3525 034270  112737  000035  001414  MOVB    #29.,RMDA0      ;LAST SECTOR
3526
3527                         ;ENABLE DEBUG CLOCK AND LOAD LAST SECTOR ADDRESS, MEMORY ADDRESS AND
3528                         ;WORD COUNT, THEN LOAD WRITE DATA COMMAND WITH GO SET
3529 034276                  10$:
3530 034276  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
     034302  000402                  BR      15$             ;BRANCH TO 15$ IF NO ERROR
```

```
        034304  104000                          EMT
3531    034306  000504                          BR      40$             ;SKIP TEST IF ERROR
3532    034310                          15$:
3533    034310  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3534    034316  013760  001440  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
3535    034324  013737  001440  001174          MOV     RMOF0,$TMP0     ;SAVE OFFSET REG FOR ERROR MSG
3536    034332  012760  001466  000034          MOV     #822.,RMDC(R0)  ;LOAD RMDC
3537    034340  013760  001414  000006          MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
3538    034346  012760  177000  000002          MOV     #-512.,RMWC(R0) ;LOAD RMWC
3539    034354  012760  103712  000004          MOV     #BUFFER,RMBA(R0)        ;LOAD RMBA
3540    034362  012760  000061  000000          MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
3541    034370  012702  000014                  MOV     #14,R2          ;R2 = CLOCK COUNT
3542                                            ;CLOCK COUNT 2-05 3-04 14-60...
3543                            ;CLOCK THE COMMAND SEQUENCER TO GENERATE SET PULSE
3544    034374                  20$:
3545    034374  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3546    034402  012760  041001  000024          MOV     #DMD.MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3547    034410  005302                          DEC     R2
3548    034412  001370                          BNE     20$             ;ISSUE 2 CLOCKS
3549
3550                            ;FORCE EBL TO GET TO OVERFLOW ADDRESS
3551    034414  012760  061001  000024          MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
3552    034422  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3553
3554                            ;VERIFY THAT ADDRESS OVERFLOW ERROR IS SET
3555    034430  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3556    034436  042737  176777  001142          BIC     #^CAOE,$BDDAT
3557    034444  001012                          BNE     30$             ;BRANCH IF AOE IS SET
3558    034446  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
3559    034452  062737  000014  001136          ADD     #RMER1,$BDADR
3560    034460  012737  001000  001140          MOV     #AOE,$GDDAT
3561    034466  104212                          EMT     212
3562    034470  000413                          BR      40$
3563    034472                          30$:                            ;END OF TEST
3564    034472  032737  010000  001440          BIT     #FMT16,RMOF0    ;DONE 16 BIT FORMAT TEST ?
3565    034500  001007                          BNE     40$             ;YES !!
3566    034502  012737  010000  001440          MOV     #FMT16,RMOF0    ;SET 16 BIT FORMAT MODE AND
3567    034510  112737  000037  001414          MOVB    #31.,RMDA0      ;LAST SECTOR
3568    034516  000667                          BR      10$
3569    034520                          40$:
3570
3571                            ;;*****************************************************************
                                ;*TEST 73        SET RMR TEST

                                ;;*****************************************************************
        034520                  TST73:
        034520  000004                  SCOPE                           ;SCOPE CALL
        034522  000240                  NOP
        034524  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        034530  013700  001276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        034534  013701  001462          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        034540  012737  000073  001226  MOV     #73,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3572
3573    034546  010037  001136          MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR MSG
3574    034552  062737  000014  001136  ADD     #RMER1,$BDADR
3575    034560  012702  034726          MOV     #100$,R2        ;INITIALIZE TABLE POINTER
3576
```

```
                                        ;CLEAR THE DEVICE AND ENABLE DEBUG CLOCK
                                        10$:
034564  004737  055614                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
034570  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
034576  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
034604  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
034612  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2

                                        ;SET GO THEN WRITE THE REGISTER SPECIFIED BY THE TABLE
034620  012760  000001  000000                  MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
034626  011203                                  MOV     (R2),R3                 ;GENERATE REGISTER ADDRESS
034630  060003                                  ADD     R0,R3
034632  012713  041001                          MOV     #DMD!MUR!DBEN,(R3)      ;WRITE THE REGISTER

                                        ;VERIFY RMR ACCORDING TO TABLE
034636  016037  000014  001142                  MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
034644  042737  177773  001142                  BIC     #^CRMR,$BDDAT
034652  016237  000002  001140                  MOV     2(R2),$GDDAT            ;GET EXPECTED RESULT FROM TABLE
034660  023737  001140  001142                  CMP     $GDDAT,$BDDAT
034666  001411                                  BEQ     30$                     ;BRANCH IF RMR IS OK
034670  011237  001174                          MOV     (R2),$TMP0              ;SAVE TEST REGISTER
034674  032762  000004  000002                  BIT     #RMR,2(R2)
034702  001002                                  BNE     20$                     ;BRANCH IF ERROR SHOULD BE ONE
034704  104213                                  EMT     213
034706  000401                                  BR      30$
034710                                  20$:
034710  104214                                  EMT     214

                                        ;ADVANCE TABLE POINTER, EXIT IF DONE
                                        30$:
034712  062702  000004                          ADD     #4,R2
034716  005712                                  TST     (R2)
034720  100401                                  BMI     40$                     ;EXIT IF ENTRY NEGATIVE
034722  000720                                  BR      10$
034724  000410                          40$:    BR      200$                    ;JUMP OVER TABLE

                                        ;TABLE OF REGISTER ADDRESSES AND RMR VALUES
                                        100$:
034726  000016                                  .WORD   RMAS                    ;ATTENTION SUMMARY REG
034730  000000                                  .WORD   0                       ;RMR = 0

034732  000024                                  .WORD   RMMR1                   ;MAINTENANCE REG
034734  000000                                  .WORD   0                       ;RMR = 0

034736  000006                                  .WORD   RMDA                    ;DISK ADDRESS REG
034740  000004                                  .WORD   RMR                     ;RMR = 1

034742  177777                                  .WORD   -1                      ;END OF TABLE
034744  000000                                  .WORD   0

034746                                  200$:                                   ;END OF TEST

                                        ;;*****************************************************************
                                        ;*TEST 74        PGM STATUS CHECK

                                        ;;*****************************************************************
034746                                  TST74:
```

```
        034746  000004                          SCOPE                   ;SCOPE CALL
        034750  000240                          NOP
        034752  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        034756  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        034762  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        034766  012737  000074  001226          MOV     #74,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  3629
  3630                                  ;CLEAR AND READ DRIVE TYPE AND DRIVE STATUS
  3631  034774  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  3632  035000  016037  000026  001174          MOV     RMDT(R0),$TMP0  ;STORE RMDT AT $TMP0
  3633  035006  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
  3634
  3635                                  ;OMIT TEST IF DRQ IS ON - ELSE VERIFY THAT PGM IS OFF
  3636  035014  032737  004000  001174          BIT     #DRQ,$TMP0
  3637  035022  001014                          BNE     10$                     ;BRANCH IF DRQ IS ON
  3638  035024  042737  176777  001142          BIC     #^CPGM,$BDDAT
  3639  035032  001410                          BEQ     10$                     ;BRANCH IF PGM IS OFF
  3640  035034  005037  001140                  CLR     $GDDAT
  3641  035040  010037  001136                  MOV     R0,$BDADR
  3642  035044  062737  000012  001134          ADD     #RMDS,$GDADR
  3643  035052  104215                          EMT     215
  3644  035054                          10$:                                    ;END OF TEST
  3645
  3646                                  ;;**********************************************************************
                                        ;*TEST 75        DVA/DPR STATUS CHECK
                                        ;;**********************************************************************
        035054                          ;;**********************************************************************
        035054                          TST75:
        035054  000004                          SCOPE                   ;SCOPE CALL
        035056  000240                          NOP
        035060  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        035064  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        035070  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        035074  012737  000075  001226          MOV     #75,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
  3647
  3648                                  ;CLEAR AND VERIFY THAT DVA IS SET
  3649  035102  004737  055614                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
  3650  035106  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
  3651  035114  042737  173777  001142          BIC     #^CDVA,$BDDAT
  3652  035122  001006                          BNE     10$                     ;BRANCH IF DVA IS ON
  3653  035124  012737  004000  001140          MOV     #DVA,$GDDAT
  3654  035132  010037  001136                  MOV     R0,$BDADR
  3655  035136  104216                          EMT     216
  3656
  3657                                  ;VERIFY THAT DPR IS SET
  3658  035140                          10$:
  3659  035140  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDA
  3660  035146  042737  177377  001142          BIC     #^CDPR,$BDDAT
  3661  035154  001011                          BNE     20$
  3662  035156  012737  000400  001140          MOV     #DPR,$GDDAT
  3663  035164  010037  001136                  MOV     R0,$BDADR
  3664  035170  062737  000012  001136          ADD     #RMDS,$BDADR
  3665  035176  104217                          EMT     217
  3666
  3667  035200                          20$:                                    ;END OF TEST
  3668
  3669                                  ;;**********************************************************************
```

```
                                        ;*TEST 76       PORT REQUEST TEST, PART 1

                                        ;;**********************************************************************
        035200                          TST76:
        035200  000004                         SCOPE                   ;SCOPE CALL
        035202  000240                         NOP
        035204  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
        035210  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        035214  013701  001462                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        035220  012737  000076  001226         MOV     #76,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3670
3671    035226  004737  055370                JSR     PC,SETVV        ;GO SET VOLUME VALID
        035232  000402                         BR      10$             ;BRANCH TO 10$ IF NO ERROR
        035234  104000                         EMT
3672    035236  000434                         BR      20$
3673
3674                                    ;EXECUTE A RELEASE TO RESET REQUEST FLOP
3675    035240                          10$:
3676    035240  012760  000013  000000         MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3677
3678                                    ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3679    035246  016002  000040                 MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3680    035252  042702  037777                 BIC     #^C<RQA!RQB>,R2
3681    035256  001024                         BNE     20$
3682
3683                                    ;READ RMCS1 TO SET REQUEST FLOP
3684    035260  016002  000000                 MOV     RMCS1(R0),R2    ;STORE RMCS1 AT R2
3685
3686                                    ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3687    035264  016005  000012                 MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3688    035270  032705  001000                 BIT     #PGM,R5         ;SEE IF PGM IS SET
3689    035274  001415                         BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
3690    035276  016037  000040  001142         MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
3691    035304  042737  037777  001142         BIC     #^C<RQA!RQB>,$BDDAT
3692    035312  001006                         BNE     20$                     ;BRANCH IF REQUEST IS SET
3693    035314  010037  001136                 MOV     R0,$BDADR
3694    035320  062737  000040  001136         ADD     #RMMR2,$BDADR
3695    035326  104220                         EMT     220
3696    035330                          20$:
3697
3698                                    ;;**********************************************************************
                                        ;*TEST 77       PORT REQUEST TEST, PART 2

                                        ;;**********************************************************************
        035330                          TST77:
        035330  000004                         SCOPE                   ;SCOPE CALL
        035332  000240                         NOP
        035334  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
        035340  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        035344  013701  001462                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        035350  012737  000077  001226         MOV     #77,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3699
3700    035356  004737  055370                JSR     PC,SETVV        ;GO SET VOLUME VALID
        035362  000402                         BR      10$             ;BRANCH TO 10$ IF NO ERROR
        035364  104000                         EMT
3701    035366  000435                         BR      20$
3702
```

```
3703                                    ;EXECUTE A RELEASE TO RESET REQUEST FLOP
3704 035370                             10$:
3705 035370  012760  000013  000000             MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3706
3707                                    ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3708 035376  016002  000040                       MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3709 035402  042702  037777                       BIC     #^C<RQA!RQB>,R2
3710 035406  001025                               BNE     20$
3711
3712                                    ;WRITE THE ATTENTION SUMMARY REGISTER TO SET REQUEST FLOP
3713 035410  012760  177777  000016             MOV     #-1,RMAS(R0)    ;LOAD RMAS
3714
3715                                    ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3716 035416  016005  000012                       MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3717 035422  032705  001000                       BIT     #PGM,R5         ;SEE IF PGM IS SET
3718 035426  001415                               BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
3719 035430  016037  000040  001142               MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
3720 035436  042737  037777  001142               BIC     #^C<RQA!RQB>,$BDDAT
3721 035444  001006                               BNE     20$
3722 035446  010037  001136                       MOV     R0,$BDADR
3723 035452  062737  000040  001136               ADD     #RMMR2,$BDADR
3724 035460  104221                               EMT     221
3725 035462                             20$:
3726
3727                                    ;;*****************************************************************
                                        ;*TEST 100      PORT REQUEST TEST, PART 3

                                        ;;*****************************************************************
     035462                             TST100:
     035462  000004                             SCOPE                   ;SCOPE CALL
     035464  000240                             NOP
     035466  012706  001100                     MOV     #STACK,SP       ;LOAD THE STACK POINTER
     035472  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     035476  013701  001462                     MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     035502  012737  000100  001226             MOV     #100,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3728
3729 035510  004737  055370                     JSR     PC,SETVV        ;GO SET VOLUME VALID
     035514  000402                             BR      10$             ;BRANCH TO 10$ IF NO ERROR
     035516  104000                             EMT
3730 035520  000435                             BR      20$
3731
3732                                    ;EXECUTE A RELEASE COMMAND TO RESET REQUEST FLOP
3733 035522                             10$:
3734 035522  012760  000013  000000             MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3735
3736                                    ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3737 035530  016002  000040                       MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3738 035534  042702  037777                       BIC     #^C<RQA!RQB>,R2
3739 035540  001025                               BNE     20$
3740
3741                                    ;WRITE RMDA TO SET REQUEST FLOP
3742 035542  012760  000000  000006             MOV     #0,RMDA(R0)     ;LOAD RMDA
3743
3744                                    ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3745 035550  016005  000012                       MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3746 035554  032705  001000                       BIT     #PGM,R5         ;SEE IF PGM IS SET
3747 035560  001415                               BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
```

```
 3748 035562  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 3749 035570  042737  037777  001142          BIC     #^C<RQA!RQB>,$BDDAT
 3750 035576  001006                          BNE     20$
 3751 035600  010037  001136                  MOV     R0,$BDADR
 3752 035604  062737  000040  001136          ADD     #RMMR2,$BDADR
 3753 035612  104222                          EMT     222
 3754
 3755 035614                          20$:
 3756
 3757                                 ;;*****************************************************************
                                      ;*TEST 101       RELEASE TEST

                                      ;;*****************************************************************
      035614                          TST101:
      035614  000004                          SCOPE                   ;SCOPE CALL
      035616  000240                          NOP
      035620  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      035624  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      035630  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      035634  012737  000101  001226          MOV     #101,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 3758
 3759                                 ;REQUEST FLOP SHOULD SET WHEN WRITTING RMCS1
 3760
 3761 035642  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      035646  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
      035650  104000                          EMT
 3762 035652  000454                          BR      40$
 3763
 3764                                 ;EXECUTE A RELEASE COMMAND
 3765 035654                          10$:
 3766 035654  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
 3767 035662  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
 3768 035670  012702  000002                  MOV     #2,R2
 3769 035674                          20$:
      035674  012760  141001  000024          MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
 3770 035702  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
 3771 035710  005302                          DEC     R2
 3772 035712  001370                          BNE     20$                     ;ISSUE 2 CLOCKS
 3773
 3774                                 ;VERIFY REQUEST FLOPS ARE RESET
 3775 035714  016037  000026  001174          MOV     RMDT(R0),$TMP0  ;STORE RMDT AT $TMP0
 3776 035722  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 3777 035730  042737  037777  001142          BIC     #^C<RQA!RQB>,$BDDAT
 3778 035736  001422                          BEQ     40$                     ;BRANCH IF REQUESTS ARE RESET
 3779 035740  032737  004000  001174          BIT     #DRQ,$TMP0
 3780 035746  001410                          BEQ     30$                     ;BRANCH IF SINGLE PORT DEVICE
 3781 035750  032737  100000  001142          BIT     #RQA,$BDDAT
 3782 035756  001412                          BEQ     40$                     ;BRANCH IF RQA IS RESET
 3783 035760  032737  040000  001142          BIT     #RQB,$BDDAT
 3784 035766  001406                          BEQ     40$                     ;BRANCH IF RQB IS RESET
 3785
 3786                                 ;DRQ IS ZERO AND A REQUEST FLOP IS ON, OR, DRQ IS ONE AND
 3787                                 ;BOTH REQUEST FLOPS ARE ON
 3788 035770                          30$:
 3789 035770  010037  001136                  MOV     R0,$BDADR
 3790 035774  062737  000040  001136          ADD     #RMMR2,$BDADR
 3791 036002  104223                          EMT     223
```

```
                                        40$:                                    ;END OF TEST
                                        ;;**************************************************************
                                        ;*TEST 102      WRITE ATA TEST
                                        ;;**************************************************************
036004                                  TST102:
036004  000004                                SCOPE                   ;SCOPE CALL
036006  000240                                NOP
036010  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
036014  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
036020  013701  001462                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
036024  012737  000102  001226                MOV     #102,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                                        ;CLEAR THE DEVICE, SET DIAGNOSTIC MODE THEN SET UNIT READY
036032  004737  055614                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
036036  012760  000001  000024                MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
036044  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
036052  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
036060  012760  001001  000024                MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
                                        ;WRITE THE ATTENTION SUMMARY REGISTER
036066  111102                                MOVB    (R1),R2                 ;ASSEMBLE THE ATA BIT IN R3
036070  042702  177770                        BIC     #^CUNTMSK,R2
036074  116203  063052                        MOVB    ATNTBL(R2),R3
036100  042703  177400                        BIC     #^CATNMSK,R3
036104  010360  000016                        MOV     R3,RMAS(R0)     ;LOAD RMAS
                                        ;READ RMDS AND VERIFY THAT ATA IS RESET
036110  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
036116  042737  077777  001142                BIC     #^CATA,$BDDAT
036124  001411                                BEQ     10$                     ;BRANCH IF ATA IS RESET
036126  010037  001136                        MOV     R0,$BDADR
036132  062737  000012  001136                ADD     #RMDS,$BDADR
036140  005037  001140                        CLR     $GDDAT
036144  104224                                EMT     224
036146  000417                                BR      20$
                                        ;READ RMAS AND VERIFY ATA IS RESET
036150                                  10$:
036150  016037  000016  001142                MOV     RMAS(R0),$BDDAT ;STORE RMAS AT $BDDAT
036156  005103                                COM     R3
036160  040337  001142                        BIC     R3,$BDDAT
036164  001410                                BEQ     20$                     ;BRANCH IF ATA IS RESET
036166  005037  001140                        CLR     $GDDAT
036172  010037  001136                        MOV     R0,$BDADR
036176  062737  000016  001136                ADD     #RMAS,$BDADR
036204  104225                                EMT     225
036206                                  20$:                                    ;END OF TEST
                                        ;;**************************************************************
                                        ;*TEST 103      RESET ATA BY GO TEST
                                        ;;**************************************************************
036206                                  TST103:
036206  000004                                SCOPE                   ;SCOPE CALL
036210  000240                                NOP
```

```
036212  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
036216  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
036222  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
036226  012737  000103  001226          MOV     #103,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3833
3834 036234  004737  055614            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3835 036240  012760  000001  000024    MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3836 036246  012760  000000  000014    MOV     #0,RMER1(R0)    ;LOAD RMER1
3837 036254  012760  000000  000042    MOV     #0,RMER2(R0)    ;LOAD RMER2
3838 036262  012760  041001  000024    MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3839
3840                                ;WITH DEBUG CLOCK ENABLED, SET GO AND VERIFY THAT ATA IS RESET
3841 036270  012760  000001  000000    MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
3842 036276  016037  000012  001142    MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3843 036304  042737  077777  001142    BIC     #^CATA,$BDDAT
3844 036312  001410                    BEQ     10$                     ;BRANCH IF ATA IS RESET
3845 036314  005037  001140            CLR     $GDDAT
3846 036320  010037  001136            MOV     R0,$BDADR
3847 036324  062737  000012  001136    ADD     #RMDS,$BDADR
3848 036332  104226                    EMT     226
3849
3850 036334                         10$:                                ;END OF TEST
3851
3852                                ;;*******************************************************************
                                    ;*TEST 104      UNIT READY ATA TEST
                                    ;;*******************************************************************
036334                              TST104:
036334  000004                      SCOPE                   ;SCOPE CALL
036336  000240                      NOP
036340  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
036344  013700  001276              MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
036350  013701  001462              MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
036354  012737  000104  001226      MOV     #104,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3853
3854                                ;SET DIAGNOSTIC MODE AND CLEAR ATA
3855 036362  004737  055614            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3856 036366  012760  000001  000024    MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3857 036374  012760  000000  000014    MOV     #0,RMER1(R0)    ;LOAD RMER1
3858 036402  012760  000000  000042    MOV     #0,RMER2(R0)    ;LOAD RMER2
3859 036410  012760  177777  000016    MOV     #-1,RMAS(R0)    ;LOAD RMAS
3860
3861                                ;SET UNIT READY AND VERIFY ATA IS SET
3862 036416  012760  001001  000024    MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
3863 036424  016037  000012  001142    MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3864 036432  042737  077777  001142    BIC     #^CATA,$BDDAT
3865 036440  001011                    BNE     10$                     ;BRANCH IF ATA IS SET
3866 036442  010037  001136            MOV     R0,$BDADR
3867 036446  062737  000012  001136    ADD     #RMDS,$BDADR
3868 036454  012737  100000  001140    MOV     #ATA,$GDDAT
3869 036462  104227                    EMT     227
3870
3871                                ;CLEAR ATA, RESET UNIT READY, AND VERIFY ATA IS SET
3872 036464                         10$:
3873 036464  012760  177777  000016    MOV     #-1,RMAS(R0)    ;LOAD RMAS
3874 036472  012760  000001  000024    MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3875 036500  016037  000012  001142    MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
```

```
 3876 036506  042737  077777  001142          BIC     #^CATA,$BDDAT
 3877 036514  001011                          BNE     20$
 3878 036516  010037  001136                  MOV     R0,$BDADR
 3879 036522  062737  000012  001136          ADD     #RMDS,$BDADR
 3880 036530  012737  100000  001140          MOV     #ATA,$GDDAT
 3881 036536  104230                          EMT     230
 3882
 3883 036540                          20$:                                ;END OF TEST
 3884
 3885                                 ;;*****************************************************************
                                      ;*TEST 105       ERROR ATA TEST
                                      ;;*****************************************************************
      036540                          TST105:
      036540  000004                          SCOPE                       ;SCOPE CALL
      036542  000240                          NOP
      036544  012706  001100                  MOV     #STACK,SP           ;LOAD THE STACK POINTER
      036550  013700  001276                  MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
      036554  013701  001462                  MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
      036560  012737  000105  001226          MOV     #105,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
 3886
 3887                                 ;CLEAR THE DEVICE AND RESET ATTENTION
 3888 036566  004737  055614                  JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
 3889 036572  012760  000001  000024          MOV     #DMD,RMMR1(R0)      ;LOAD RMMR1
 3890 036600  012760  000000  000014          MOV     #0,RMER1(R0)        ;LOAD RMER1
 3891 036606  012760  000000  000042          MOV     #0,RMER2(R0)        ;LOAD RMER2
 3892 036614  012760  177777  000016          MOV     #-1,RMAS(R0)        ;LOAD RMAS
 3893
 3894                                 ;WRITE ONES IN ERROR REGISTER 1 AND VERIFY ATA IS SET
 3895 036622  012760  177777  000014          MOV     #-1,RMER1(R0)       ;LOAD RMER1
 3896 036630  016037  000012  001142          MOV     RMDS(R0),$BDDAT     ;STORE RMDS AT $BDDAT
 3897 036636  042737  077777  001142          BIC     #^CATA,$BDDAT
 3898 036644  001011                          BNE     10$
 3899 036646  012737  100000  001140          MOV     #ATA,$GDDAT
 3900 036654  010037  001136                  MOV     R0,$BDADR
 3901 036660  062737  000012  001136          ADD     #RMDS,$BDADR
 3902 036666  104231                          EMT     231
 3903
 3904 036670                          10$:                                ;END OF TEST
 3905
 3906                                 ;;*****************************************************************
                                      ;*TEST 106       REGISTER TRANSFER ATA TEST
                                      ;;*****************************************************************
      036670                          TST106:
      036670  000004                          SCOPE                       ;SCOPE CALL
      036672  000240                          NOP
      036674  012706  001100                  MOV     #STACK,SP           ;LOAD THE STACK POINTER
      036700  013700  001276                  MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
      036704  013701  001462                  MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
      036710  012737  000106  001226          MOV     #106,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
 3907
 3908 036716  012702  037044                  MOV     #100$,R2                    ;INITIALIZE TABLE ADDRESS
 3909
 3910                                 ;FORCE COMPOSITE ERROR AND RESET ATA
 3911 036722                          5$:
 3912 036722  004737  055614                  JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
```

```
 3913 036726  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 3914 036734  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
 3915 036742  012760  177777  000016          MOV     #-1,RMAS(R0)    ;LOAD RMAS
 3916
 3917                                  ;WRITE THE TEST REGISTER AND VERIFY ATA
 3918 036750  011203                          MOV     (R2),R3                 ;GENERATE REGISTER ADDRESS
 3919 036752  060003                          ADD     R0,R3
 3920 036754  005013                          CLR     (R3)
 3921 036756  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 3922 036764  042737  077777  001142          BIC     #^CATA,$BDDAT
 3923 036772  016237  000002  001140          MOV     2(R2),$GDDAT
 3924 037000  023737  001140  001142          CMP     $GDDAT,$BDDAT
 3925 037006  001410                          BEQ     10$                     ;BRANCH IF ATA IS OK
 3926 037010  010337  001174                  MOV     R3,$TMP0
 3927 037014  010037  001136                  MOV     R0,$BDADR
 3928 037020  062737  000012  001136          ADD     #RMDS,$BDADR
 3929 037026  104232                          EMT     232
 3930
 3931                                  ;MOVE TABLE POINTER - EXIT IF DONE
 3932 037030                           10$:
 3933 037030  062702  000004                  ADD     #4,R2
 3934 037034  005712                          TST     (R2)
 3935 037036  100401                          BMI     20$                     ;EXIT IF ENTRY MINUS
 3936 037040  000730                          BR      5$
 3937 037042  000410                   20$:   BR      200$                    ;JUMP OVER TABLE
 3938
 3939                                  ;TABLE OF REGISTER ADDRESSES AND ATA BITS
 3940 037044                           100$:
 3941 037044  000016                          .WORD   RMAS
 3942 037046  000000                          .WORD
 3943
 3944 037050  000006                          .WORD   RMDA
 3945 037052  100000                          .WORD   ATA
 3946
 3947 037054  000000                          .WORD   RMCS1
 3948 037056  000000                          .WORD
 3949
 3950 037060  177777                          .WORD   -1                      ;END OF TABLE
 3951 037062  000000                          .WORD
 3952
 3953 037064                           200$:                                  ;END OF TEST
 3954
 3955                                  ;;*******************************************************************
                                       ;*TEST 107       P SET ATA TEST
                                       ;;*******************************************************************
      037064                           TST107:
      037064  000004                          SCOPE                   ;SCOPE CALL
      037066  000240                          NOP
      037070  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      037074  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      037100  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      037104  012737  000107  001226          MOV     #107,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 3956
 3957 037112  012702  037254                  MOV     #100$,R2        ;INIT'ALIZE TABLE POINTER
 3958 037116                           10$:
 3959 037116  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
```

```
                037122  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
                037124  104000                          EMT
     3960       037126  000451                          BR      50$
     3961
     3962                                               ;EXECUTE THE COMMAND FROM THE TABLE
     3963       037130                                  20$:
     3964       037130  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
     3965       037136  011203                          MOV     (R2),R3         ;GET FUNCTION CODE
     3966       037140  052703  000001                  BIS     #GO,R3
     3967       037144  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
     3968       037150  012703  000003                  MOV     #3,R3
     3969       037154                                  30$:
     3970       037154  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
     3971       037162  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
     3972       037170  005303                          DEC     R3
     3973       037172  001370                          BNE     30$
     3974
     3975                                               ;VERIFY THAT ATA IS SET
     3976       037174  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
     3977       037202  042737  077777  001142          BIC     #^CATA,$BDDAT
     3978       037210  001013                          BNE     40$
     3979       037212  010037  001136                  MOV     R0,$BDADR
     3980       037216  062737  000012  001136          ADD     #RMDS,$BDADR
     3981       037224  012737  100000  001140          MOV     #ATA,$GDDAT
     3982       037232  011237  001174                  MOV     (R2),$TMP0
     3983       037236  104233                          EMT     233
     3984
     3985                                               ;ADVANCE TABLE POINTER-EXIT IF DONE
     3986       037240                                  40$:
     3987       037240  062702  000002                  ADD     #2,R2
     3988       037244  005712                          TST     (R2)
     3989       037246  100401                          BMI     50$
     3990       037250  000722                          BR      10$
     3991       037252  000403                  50$:    BR      200$            ;JUMP OVER TABLE
     3992
     3993                                               ;TABLE OF FUNCTION CODES
     3994       037254                                  100$:
     3995       037254  000014                          .WORD   OFFSET
     3996       037256  000016                          .WORD   RTC
     3997       037260  177777                          .WORD   -1              ;END OF TABLE
     3998       037262                                  200$:
     3999
     4000                                               ;;*****************************************************************
                                                        ;*TEST 110       SET WLE TEST

                                                        ;;*****************************************************************
                037262                                  TST110:
                037262  000004                          SCOPE                   ;SCOPE CALL
                037264  000240                          NOP
                037266  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
                037272  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                037276  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                037302  012737  000110  001226          MOV     #110,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
     4001
     4002       037310  012702  037504                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
     4003       037314                                  10$:
     4004       037314  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
```

```
        037320  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
        037322  104000                          EMT
   4005 037324  000466                          BR      50$
   4006
   4007                                 ;ENABLE DEBUG CLOCK AND SET WRITE PROTECT ACCORDING TO TABLE
   4008 037326                          20$:
   4009 037326  016203  000002                  MOV     2(R2),R3        ;GET WRITE PROTECT FROM TABLE
   4010 037332  052703  041001                  BIS     #DMD!MUR!DBEN,R3        ;SET OTHER MAINT BITS
   4011 037336  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
   4012
   4013                                 ;LOAD AND EXECUTE THE COMMAND FROM THE TABLE
   4014 037342  011204                          MOV     (R2),R4         ;GET FUNCTION CODE FROM TABLE
   4015 037344  052704  000001                  BIS     #GO,R4          ;SET GO
   4016 037350  010460  000000                  MOV     R4,RMCS1(R0)    ;LOAD RMCS1
   4017 037354  012705  000002                  MOV     #2,R5           ;R5=CLOCK COUNT
   4018 037360  052703  100000          30$:    BIS     #DBCK,R3        ;SET CLOCK
   4019 037364  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
   4020 037370  042703  100000                  BIC     #DBCK,R3        ;RESET CLOCK
   4021 037374  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
   4022 037400  005305                          DEC     R5
   4023 037402  001366                          BNE     30$             ;ISSUE 2 CLOCKS
   4024
   4025                                 ;VERIFY THAT WRITE LOCK ERROR IS ACCORDING TO TABLE
   4026 037404  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
   4027 037412  042737  173777  001142          BIC     #^CWLE,$BDDAT
   4028 037420  026237  000004  001142          CMP     4(R2),$BDDAT
   4029 037426  001417                          BEQ     40$             ;BRANCH IF WLE IS OK
   4030 037430  016237  000004  001140          MOV     4(R2),$GDDAT    ;SAVE DATA FOR ERROR MSG
   4031 037436  010037  001136                  MOV     R0,$BDADR
   4032 037442  062737  000014  001136          ADD     #RMER1,$BDADR
   4033 037450  011237  001174                  MOV     (R2),$TMP0      ;$TMP0=FUNCTION CODE
   4034 037454  016237  000002  001176          MOV     2(R2),$TMP1     ;$TMP1=WRITE PROTECT
   4035 037462  104234                          EMT     234
   4036 037464  000406                          BR      50$
   4037
   4038                                 ;ADVANCE TABLE POINTER TO NEXT FUNCTION CODE-EXIT IF DONE
   4039 037466                          40$:
   4040 037466  062702  000006                  ADD     #6,R2
   4041 037472  005762  000002                  TST     2(R2)
   4042 037476  100401                          BMI     50$
   4043 037500  000705                          BR      10$             ;REPEAT TEST
   4044
   4045 037502  000416                  50$:    BR      200$            ;JUMP OVER TABLE
   4046
   4047                                 ;TABLE OF FUNCTION CODES, WRITE PROTECT AND WRITE LOCK ERRORS
   4048 037504                          100$:
   4049 037504  000060                          .WORD   WD              ;WRITE DATA COMMAND
   4050 037506  000010                          .WORD   MWP             ;WRITE PROTECT ON
   4051 037510  004000                          .WORD   WLE             ;WRITE LOCK ERROR ONE
   4052
   4053 037512  000060                          .WORD   WD              ;WRITE DATA COMMAND
   4054 037514  000000                          .WORD                   ;MWP OFF
   4055 037516  000000                          .WORD                   ;WLE OFF
   4056
   4057 037520  000070                          .WORD   RD              ;READ DATA COMMAND
   4058 037522  000010                          .WORD   MWP             ;MWP ON
   4059 037524  000000                          .WORD                   ;WLE OFF
```

```
                                .WORD   RIP             ;READ IN PRESET COMMAND
                                .WORD   MWP             ;MWP ON
                                .WORD                   ;WLE OFF

                                .WORD                   ;END OF TABLE
                                .WORD   -1

                        200$:                           ;END OF TEST

                        ;;**********************************************************************
                        ;*TEST 111      EXCEPTION TEST

                        ;;**********************************************************************
037540                  TST111:
037540  000004                SCOPE                   ;SCOPE CALL
037542  000240                NOP
037544  012706  001100        MOV     #STACK,SP       ;LOAD THE STACK POINTER
037550  013700  001276        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
037554  013701  001462        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
037560  012737  000111  001226 MOV     #111,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

                        ;WITH OCCUPIED SET, EACH OF THE FOLLOWING ERRORS SHOULD CAUSE AN
                        ;EXCEPTION:
                        ;
                        ;       PAR, IF RUN AND GO IS SET
                        ;       RMR, IF RUN AND GO IS SET
                        ;       ILR, IF RUN AND GO IS SET
                        ;       DCK
                        ;       HCE
                        ;       HCRC
                        ;       FER
                        ;       OPI

037566  012702  040040        MOV     #100$,R2        ;INITIALIZE TABLE POINTER

                        ;INITIALIZE AND VERIFY THAT EXCEPTION IS RESET
037572                  10$:
037572  004737  055614        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
037576  016037  000024  001142 MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
037604  042737  167777  001142 BIC     #^CREX,$BDDAT
037612  001410                BEQ     20$             ;BRANCH IF EXCEPTION IS 0
037614  010037  001136        MOV     R0,$BDADR
037620  062737  000024  001136 ADD     #RMMR1,$BDADR
037626  005037  001140        CLR     $GDDAT
037632  104235                EMT     235
037634                  20$:
037634  004737  055370        JSR     PC,SETVV        ;GO SET VOLUME VALID
037640  000402                BR      30$             ;BRANCH TO 30$ IF NO ERROR
037642  104000                EMT
037644  000474                BR      60$

                        ;EXECUTE A WRITE DATA COMMAND TO SET OCCUPIED, RUN AND GO
037646                  30$:
037646  012760  000000  000006 MOV     #0,RMDA(R0)     ;LOAD RMDA
037654  012760  000000  000034 MOV     #0,RMDC(R0)     ;LOAD RMDC
037662  012760  041001  000024 MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
4105 037670  012760  000061  000000          MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
4106 037676  012703  000002                  MOV     #2,R3
4107 037702                          40$:
     037702  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4108 037710  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4109 037716  005303                          DEC     R3
4110 037720  001370                          BNE     40$             ;ISSUE 2 CLOCKS
4111
4112                                 ;LOAD ERROR REGISTER WITH ENTRY FROM TABLE AND VERIFY THAT EXCEPTION IS SET
4113 037722  011260  000014                  MOV     (R2),RMER1(R0)  ;LOAD RMER1
4114 037726  012737  000200  001530          MOV     #200,WATCH      ;SET WATCHDOG TIMER VALUE
     037734  004777  141572                  JSR     PC,@CLOCK       ;START THE CLOCK
4115 037740                          45$:
     037740  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
4116 037746  042737  167777  001142          BIC     #^CREX,$BDDAT
4117 037754  001021                          BNE     50$
4118 037756  005737  001530                  TST     WATCH           ;HAS CLOCK EXPIRED ??
4119 037762  001366                          BNE     45$             ;NO - TAKE ANOTHER SAMPLE
4120 037764  004777  141544                  JSR     PC,@STOP        ;STOP THE CLOCK
4121 037770  012737  010000  001140          MOV     #REX,$GDDAT
4122 037776  010037  001136                  MOV     R0,$BDADR
4123 040002  062737  000024  001136          ADD     #RMMR1,$B ADR
4124 040010  011237  001174                  MOV     (R2),$TMP0
4125 040014  104236                          EMT     236
4126 040016  000407                          BR      60$
4127
4128                                 ;ADVANCE TABLE POINTER-EXIT IF DONE
4129 040020                          50$:
4130 040020  004777  141510                  JSR     PC,@STOP        ;STOP THE CLOCK
4131 040024  062702  000002                  ADD     #2,R2
4132 040030  005712                          TST     (R2)
4133 040032  001401                          BEQ     60$
4134 040034  000656                          BR      10$
4135 040036  000402                  60$:    BR      200$            ;JUMP OVER TABLE
4136
4137                                 ;PRESENTLY, THE TABLE HAS ONLY ONE ENTRY, THE TEST USING RMR.  THE
4138                                 ;TABLE SHOULD BE EXPANDED TO TEST ALL THE CONDITIONS LISTED ABOVE IF
4139                                 ;A HARDWARE CHANGE IS MADE SUCH THAT IT IS POSSIBLE TO WRITE ERROR
4140                                 ;REGISTER 1 WITH GO SET.
4141 040040                          100$:
4142 040040  000004                          .WORD   RMR             ;RMR IS CAUSED BY HARDWARE
4143
4144 040042  000000                          .WORD                   ;END OF TABLE
4145 040044                          200$:                           ;END OF TEST
4146
4147                                 ;;*****************************************************************
                                     ;*TEST 112       RECALIBRATE TEST

                                     ;;*****************************************************************
     040044                          TST112:
     040044  000004                          SCOPE                   ;SCOPE CALL
     040046  000240                          NOP
     040050  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     040054  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     040060  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     040064  012737  000112  001226          MOV     #112,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
4148
```

```
4149 040072  004737  055370            JSR     PC,SETVV        ;GO SET VOLUME VALID
     040076  000403                    BR      10$             ;BRANCH TO 10$ IF NO ERROR
     040100  104000                    EMT
4150 040102  000137  041400            JMP     330$
4151
4152                                ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4153 040106                         10$:
4154 040106  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4155 040114  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
4156 040122  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
4157 040130  012760  000007  000000        MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4158
4159                                ;STEP COMMAND SEQUENCER TO RECAL COM (2 CLOCKS)
4160 040136  012702  000002            MOV     #2,R2
4161 040142                         20$:
     040142  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4162 040150  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4163 040156  005302                    DEC     R2
4164 040160  001370                    BNE     20$
4165
4166                                ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
4167 040162  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
4168 040170  012702  000002            MOV     #2,R2
4169 040174                         30$:
     040174  012760  140001  000024        MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR
4170 040202  012760  040001  000024        MOV     #DMD:DBEN,RMMR1(R0)     ;LOAD RMMR1
4171 040210  005302                    DEC     R2
4172 040212  001370                    BNE     30$
4173
4174                                ;VERIFY THAT OPI IS SET
4175 040214  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4176 040222  042737  157777  001142        BIC     #^COPI,$BDDAT
4177 040230  001011                    BNE     40$
4178 040232  012737  020000  001140        MOV     #OPI,$GDDAT
4179 040240  010037  001136            MOV     R0,$BDADR
4180 040244  062737  000014  001136        ADD     #RMER1,$BDADR
4181 040252  104241                    EMT     241
4182
4183 040254  012737  040262  001124  40$:  MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
4184
4185                                ;VERIFY THAT RECALIBRATE ABORTS DURING EXECUTION
4186 040262                         50$:
4187 040262  004737  055370            JSR     PC,SETVV        ;GO SET VOLUME VALID
     040266  000403                    BR      60$             ;BRANCH TO 60$ IF NO ERROR
     040270  104000                    EMT
4188 040272  000137  041400            JMP     330$
4189
4190                                ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4191 040276                         60$:
4192 040276  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4193 040304  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
4194 040312  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
4195 040320  012760  000007  000000        MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4196
4197                                ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
4198 040326  012702  000003            MOV     #3,R2
4199 040332                         70$:
```

```
4200 040332  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4201 040340  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4202 040346  005302                          DEC     R2
4203 040350  001370                          BNE     70$
4204
4205                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
4206 040352  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
4207
4208                                 ;STEP 2 CLOCKS AND VERIFY GO IS RESET
4209 040360  012702  000002                  MOV     #2,R2
4210 040364                          80$:
     040364  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
4211 040372  012760  041101  000024          MOV     #DMD!MUR!DBEN.MDF,RMMR1(R0)     ;LOAD RMMR1
4212 040400  005302                          DEC     R2
4213 040402  001370                          BNE     80$
4214 040404  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4215 040412  042737  177776  001142          BIC     #^CGO,$BDDAT
4216 040420  001405                          BEQ     90$
4217 040422  010037  001136                  MOV     R0,$BDADR
4218 040426  005037  001140                  CLR     $GDDAT
4219 040432  104242                          EMT     242
4220
4221 040434  012737  040442  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4222
4223                                 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
4224 040442                          100$:
4225 040442  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     040446  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
     040450  104000                          EMT
4226 040452  000137  041400                  JMP     330$
4227
4228                                 ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4229 040456                          110$:
4230 040456  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4231 040464  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4232 040472  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4233 040500  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4234
4235                                 ;STEP THE COMMAND SEQUENCER
4236 040506  012702  000017                  MOV     #15.,R2
4237 040512                          120$:
4238 040512  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4239 040520  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4240 040526  005302                          DEC     R2
4241 040530  001370                          BNE     120$
4242
4243                                 ;VERIFY THAT OPI IS SET
4244 040532  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4245 040540  042737  157777  001142          BIC     #^COPI,$BDDAT
4246 040546  001011                          BNE     130$
4247 040550  010037  001136                  MOV     R0,$BDADR
4248 040554  062737  000014  001136          ADD     #RMER1,$BDADR
4249 040562  012737  020000  001140          MOV     #OPI,$GDDAT
4250 040570  104243                          EMT     243
4251
4252 040572  012737  040600  001124  130$:    MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4253
```

```
                                        ;VERIFY ATA SETS IF DRIVE COMPLETES RECALIBRATE (ON CYLINDER SETS)
040600                                  150$:
040600  004737  055370                          JSR     PC,SETVV        ;GO SET VOLUME VALID
040604  000403                                  BR      160$            ;BRANCH TO 160$ IF NO ERROR
040606  104000                                  EMT
040610  000137  041400                          JMP     330$

                                        ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
040614                                  160$:
040614  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
040622  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
040630  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
040636  012760  000007  000000                MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1

                                        ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
040644  012702  000015                        MOV     #13.,R2
040650                                  170$:
040650  012760  141401  000024                MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
040656  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
040664  005302                                DEC     R2
040666  001370                                BNE     170$

                                        ;DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
040670  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
040676  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1

                                        ;STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
040704  012702  000003                        MOV     #3,R2
040710                                  180$:
040710  012760  141401  000024                MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
040716  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
040724  005302                                DEC     R2
040726  001370                                BNE     180$

                                        ;VERIFY ATA IS SET
040730  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
040736  042737  077777  001142                BIC     #^CATA,$BDDAT
040744  001011                                BNE     190$
040746  012737  100000  001140                MOV     #ATA,$GDDAT
040754  010037  001136                        MOV     R0,$BDADR
040760  062737  000012  001136                ADD     #RMDS,$BDADR
040766  104244                                EMT     244

040770  012737  040776  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY THAT RECALIBRATE ABORTS AFTER EXECUTION DURING WAIT LOOP
040776                                  200$:
040776  004737  055370                          JSR     PC,SETVV        ;GO SET VOLUME VALID
041002  000402                                  BR      210$            ;BRANCH TO 210$ IF NO ERROR
041004  104000                                  EMT
041006  000574                                  BR      330$

                                        ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
041010                                  210$:
041010  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
041016  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
041024  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
```

```
 4306 041032  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)      ;LOAD RMCS1
 4307
 4308                                  ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
 4309 041040  012702  000015                  MOV     #13.,R2
 4310 041044                           220$:
 4311 041044  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)         ;LOAD RMMR1
 4312 041052  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)      ;LOAD RMMR1
 4313 041060  005302                          DEC     R2
 4314 041062  001370                          BNE     220$
 4315
 4316                                  ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
 4317 041064  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4318
 4319                                  ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
 4320 041072  012702  000007                  MOV     #7,R2
 4321 041076                           230$:
 4322 041076  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)     ;LOAD RMMR1
 4323 041104  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4324 041112  005302                          DEC     R2
 4325 041114  001370                          BNE     230$
 4326
 4327                                  ;VERIFY THAT GO IS STILL SET
 4328 041116  016037  000000  001142          MOV     RMCS1(R0),$BDDAT         ;STORE RMCS1 AT $BDDAT
 4329 041124  042737  177776  001142          BIC     #^CGO,$BDDAT
 4330 041132  001006                          BNE     240$
 4331 041134  012737  000001  001140          MOV     #GO,$GDDAT
 4332 041142  010037  001136                  MOV     R0,$BDADR
 4333 041146  104245                          EMT     245
 4334
 4335                                  ;SET SEEK INCOMPLETE ERROR
 4336 041150                           240$:
      041150  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)     ;LOAD RMMR1
 4337
 4338                                  ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
 4339 041156  012702  000003                  MOV     #3,R2
 4340 041162                           250$:
      041162  012760  141201  000024          MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)        ;LOAD RMMR1
 4341 041170  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)     ;LOAD RMMR1
 4342 041176  005302                          DEC     R2
 4343 041200  001370                          BNE     250$
 4344 041202  016037  000000  001142          MOV     RMCS1(R0),$BDDAT         ;STORE RMCS1 AT $BDDAT
 4345 041210  042737  177776  001142          BIC     #^CGO,$BDDAT
 4346 041216  001405                          BEQ     260$
 4347 041220  010037  001136                  MOV     R0,$BDADR
 4348 041224  005037  001140                  CLR     $GDDAT
 4349 041230  104246                          EMT     246
 4350
 4351 041232  012737  041240  001124  260$:   MOV     #300$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
 4352
 4353                                  ;VERIFY THE TAG BUS DURING RECALIBRATE
 4354 041240                           300$:
 4355 041240  004737  055370                  JSR     PC,SETVV         ;GO SET VOLUME VALID
      041244  000402                          BR      310$             ;BRANCH TO 310$ IF NO ERROR
      041246  104000                          EMT
 4356 041250  000453                          BR      330$
 4357
 4358                                  ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
```

```
                                        310$:
041252                                          
041252  012760  041401  000024                  MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
041260  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
041266  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
041274  012760  000007  000000                  MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
041302  012702  041402                          MOV     #400$,R2        ;INITIALIZE TABLE POINTER
                                        
                                        ;VERIFY TAG BUS ACCORDING TO TABLE
041306                                  315$:
041306  016037  000040  001142                  MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
041314  042737  150000  001142                  BIC     #RQA!RQB!TST,$BDDAT
041322  021237  001142                          CMP     (R2),$BDDAT
041326  001411                                  BEQ     320$
041330  011237  001140                          MOV     (R2),$GDDAT
041334  010037  001136                          MOV     R0,$BDADR
041340  062737  000040  001136                  ADD     #RMMR2,$BDADR
041346  104247                                  EMT     247
041350  000413                                  BR      330$
                                        
                                        ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
041352                                  320$:
041352  062702  000002                          ADD     #2,R2
041356  005712                                  TST     (R2)
041360  100407                                  BMI     330$            ;EXIT IF ENTRY NEGATIVE
                                        
                                        ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
041362  012760  151001  000024                  MOV     #DMD!DBEN!MUR!MOV'DBCK,RMMR1(R0)        ;LOAD RMMR1
041370  012760  041401  000024                  MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
041376  000743                                  BR      315$
041400  000416                          330$:   BR      500$            ;JUMP OVER TABLE
                                        
                                        ;TABLE OF TAG BUS CONTROL AND BIT VALUES
041402                                  400$:
041402  001777                                  .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
041404  001777                                  .WORD   1777
041406  006100                                  .WORD   CC!CH!BB06      ;CONTROL BITS ENABLED, BIT 6 ON
041410  006100                                  .WORD   CC!CH!BB06
041412  026100                                  .WORD   TAG!CC!CH!BB06  ;TAG COMES ON
041414  026100                                  .WORD   TAG!CC!CH!BB06
041416  026100                                  .WORD   TAG!CC!CH!BB06
041420  026100                                  .WORD   TAG!CC!CH!BB06
041422  026100                                  .WORD   TAG!CC!CH!BB06
041424  026100                                  .WORD   TAG!CC!CH!BB06
041426  006100                                  .WORD   CC!CH!BB06      ;TAG GOES OFF
041430  006100                                  .WORD   CC!CH!BB06
041432  001777                                  .WORD   1777            ;CONTROL BITS DISABLED
                                        
041434  177777                                  .WORD   -1              ;END OF TABLE
                                        
041436                                  500$:                           ;END OF TEST
                                        
                                        ;;*****************************************************************
                                        ;*TEST 113       SEEK TEST
                                        
                                        ;;*****************************************************************
041436                                  TST113:
041436  000004                                  SCOPE                   ;SCOPE CALL
```

```
        041440  000240                          NOP
        041442  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        041446  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        041452  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        041456  012737  000113  001226          MOV     #113,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
  4410
  4411                                  ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
  4412
  4413  041464  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        041470  000403                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
        041472  104000                          EMT
  4414  041474  000137  043114                  JMP     330$
  4415
  4416                                  ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
  4417                                  ;ADDRESS,  AND LOAD SEEK COMMAND
  4418  041500                          10$:
  4419  041500  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  4420  041506  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
  4421  041514  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
  4422  041522  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
  4423  041530  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
  4424  041536  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
  4425
  4426                                  ;STEP COMMAND SEQUENCER TO SEEK COM (2 CLOCKS)
  4427  041544  012702  000002                  MOV     #2,R2
  4428  041550                          20$:
        041550  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
  4429  041556  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
  4430  041564  005302                          DEC     R2
  4431  041566  001370                          BNE     20$
  4432
  4433                                  ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
  4434  041570  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
  4435  041576  012702  000002                  MOV     #2,R2
  4436  041602                          30$:
        041602  012760  140001  000024          MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
  4437  041610  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
  4438  041616  005302                          DEC     R2
  4439  041620  001370                          BNE     30$
  4440
  4441                                  ;VERIFY THAT OPI IS SET
  4442  041622  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
  4443  041630  042737  157777  001142          BIC     #^COPI,$BDDAT
  4444  041636  001011                          BNE     40$
  4445  041640  012737  020000  001140          MOV     #OPI,$GDDAT
  4446  041646  010037  001136                  MOV     R0,$BDADR
  4447  041652  062737  000014  001136          ADD     #RMER1,$BDADR
  4448  041660  104250                          EMT     250
  4449
  4450  041662  012737  041670  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
  4451
  4452                                  ;VERIFY THAT SEEK ABORTS DURING EXECUTION
  4453  041670                          50$:
  4454  041670  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        041674  000403                          BR      60$             ;BRANCH TO 60$ IF NO ERROR
        041676  104000                          EMT
  4455  041700  000137  043114                  JMP     330$
```

```
                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEEK COMMAND
041704                                  60$:
041704  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
041712  012760  000000  000006                MOV     #0,RMDA(R0)     ;LOAD RMDA
041720  012760  000000  000034                MOV     #0,RMDC(R0)     ;LOAD RMDC
041726  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
041734  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
041742  012760  000005  000000                MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1

                                        ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
041750  012702  000003                      MOV     #3,R2
041754                                  70$:
041754  012760  141001  000024              MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
041762  012760  041001  000024              MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
041770  005302                          DEC     R2
041772  001370                          BNE     70$

                                        ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
041774  012760  041101  000024              MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1

                                        ;STEP 2 CLOCKS AND VERIFY GO IS RESET
042002  012702  000002                      MOV     #2,R2
042006                                  80$:
042006  012760  141101  000024              MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
042014  012760  041101  000024              MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
042022  005302                          DEC     R2
042024  001370                          BNE     80$
042026  016037  000000  001142              MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
042034  042737  177776  001142              BIC     #^CGO,$BDDAT
042042  001405                          BEQ     90$
042044  010037  001136                      MOV     R0,$BDADR
042050  005037  001140                      CLR     $GDDAT
042054  104251                          EMT     251

042056  012737  042064  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
042064                                  100$:
042064  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
042070  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
042072  104000                          EMT
042074  000137  043114                  JMP     330$

                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEEK COMMAND
042100                                  110$:
042100  012760  041001  000024              MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
042106  012760  000000  000006              MOV     #0,RMDA(R0)     ;LOAD RMDA
042114  012760  000000  000034              MOV     #0,RMDC(R0)     ;LOAD RMDC
042122  012760  000000  000014              MOV     #0,RMER1(R0)    ;LOAD RMER1
042130  012760  000000  000042              MOV     #0,RMER2(R0)    ;LOAD RMER2
042136  012760  000005  000000              MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1

                                        ;STEP THE COMMAND SEQUENCER
042144  012702  000017                      MOV     #15.,R2
```

```
 4510 042150                                  120$:
 4511 042150  012760  141001  000024                 MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4512 042156  012760  041001  000024                 MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4513 042164  005302                                 DEC     R2
 4514 042166  001370                                 BNE     120$
 4515
 4516                                         ;VERIFY THAT OPI IS SET
 4517 042170  016037  000014  001142                 MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 4518 042176  042737  157777  001142                 BIC     #^COPI,$BDDAT
 4519 042204  001011                                 BNE     130$
 4520 042206  010037  001136                         MOV     R0,$BDADR
 4521 042212  062737  000014  001136                 ADD     #RMER1,$BDADR
 4522 042220  012737  020000  001140                 MOV     #OPI,$GDDAT
 4523 042226  104252                                 EMT     252
 4524
 4525 042230  012737  042236  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4526
 4527                                         ;VERIFY ATA SETS IF DRIVE COMPLETES SEEK (ON CYLINDER SETS)
 4528 042236                                  150$:
 4529 042236  004737  055370                         JSR     PC,SETVV        ;GO SET VOLUME VALID
      042242  000403                                 BR      160$            ;BRANCH TO 160$ IF NO ERROR
      042244  104000                                 EMT
 4530 042246  000137  043114                         JMP     330$
 4531
 4532                                         ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4533                                         ;ADDRESS,  AND LOAD SEEK COMMAND
 4534 042252                                  160$:
 4535 042252  012760  041401  000024                 MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4536 042260  012760  000000  000006                 MOV     #0,RMDA(R0)     ;LOAD RMDA
 4537 042266  012760  000000  000034                 MOV     #0,RMDC(R0)     ;LOAD RMDC
 4538 042274  012760  000000  000014                 MOV     #0,RMER1(R0)    ;LOAD RMER1
 4539 042302  012760  000000  000042                 MOV     #0,RMER2(R0)    ;LOAD RMER2
 4540 042310  012760  000005  000000                 MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
 4541
 4542                                         ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
 4543 042316  012702  000015                         MOV     #13.,R2
 4544 042322                                  170$:
      042322  012760  141401  000024                 MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
 4545 042330  012760  041401  000024                 MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4546 042336  005302                                 DEC     R2
 4547 042340  001370                                 BNE     170$
 4548
 4549                                         ;DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
 4550 042342  012760  041001  000024                 MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4551 042350  012760  041401  000024                 MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4552
 4553                                         ;STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
 4554 042356  012702  000003                         MOV     #3,R2
 4555 042362                                  180$:
 4556 042362  012760  141401  000024                 MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 4557 042370  012760  041401  000024                 MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4558 042376  005302                                 DEC     R2
 4559 042400  001370                                 BNE     180$
 4560
 4561                                         ;VERIFY ATA IS SET
 4562 042402  016037  000012  001142                 MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 4563 042410  042737  077777  001142                 BIC     #^CATA,$BDDAT
```

```
 4564 042416  001011                          BNE     190$
 4565 042420  012737  100000  001140          MOV     #ATA,$GDDAT
 4566 042426  010037  001136                  MOV     R0,$BDADR
 4567 042432  062737  000012  001136          ADD     #RMDS,$BDADR
 4568 042440  104253                          EMT     253
 4569
 4570 042442  012737  042450  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4571
 4572                                 ;VERIFY THAT SEEK ABORTS AFTER EXECUTION DURING WAIT LOOP
 4573 042450                          200$:
 4574 042450  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      042454  000403                          BR      210$            ;BRANCH TO 210$ IF NC ERROR
      042456  104000                          EMT
 4575 042460  000137  043114                  JMP     330$
 4576
 4577                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4578                                 ;ADDRESS  AND LOAD SEEK COMMAND
 4579 042464                          210$:
 4580 042464  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4581 042472  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 4582 042500  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 4583 042506  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 4584 042514  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 4585 042522  012760  000005  000000          MOV     #SEEK.GO,RMCS1(R0)      ;LOAD RMCS1
 4586
 4587                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
 4588 042530  012702  000015                  MOV     #13.,R2
 4589 042534                          220$:
 4590 042534  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 4591 042542  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4592 042550  005302                          DEC     R2
 4593 042552  001370                          BNE     220$
 4594
 4595                                 ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
 4596 042554  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4597
 4598                                 ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
 4599 042562  012702  000007                  MOV     #7,R2
 4600 042566                          230$:
 4601 042566  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4602 042574  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4603 042602  005302                          DEC     R2
 4604 042604  001370                          BNE     230$
 4605
 4606                                 ;VERIFY THAT GO IS STILL SET
 4607 042606  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 4608 042614  042737  177776  001142          BIC     #^CGO,$BDDAT
 4609 042622  001006                          BNE     240$
 4610 042624  012737  000001  001140          MOV     #GO,$GDDAT
 4611 042632  010037  001136                  MOV     R0,$BDADR
 4612 042636  104254                          EMT     254
 4613
 4614                                 ;SET SEEK INCOMPLETE ERROR
 4615 042640                          240$:
      042640  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
 4616
 4617                                 ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
```

```
 4618 042646  012702  000003                    MOV     #3,R2
 4619 042652                          250$:
      042652  012760  141201  000024            MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
 4620 042660  012760  041201  000024            MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
 4621 042666  005302                            DEC     R2
 4622 042670  001370                            BNE     250$
 4623 042672  016037  000000  001142            MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 4624 042700  042737  177776  001142            BIC     #^CGO,$BDDAT
 4625 042706  001405                            BEQ     260$
 4626 042710  010037  001136                    MOV     R0,$BDADR
 4627 042714  005037  001140                    CLR     $GDDAT
 4628 042720  104255                            EMT     255
 4629
 4630 042722  012737  042734  001124  260$:    MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4631 042730  012703  000001                    MOV     #1,R3           ,INITIALIZE CYLINDER ADDRESS
 4632
 4633                                  ;VERIFY THE TAG BUS DURING SEEK
 4634 042734                          300$:
 4635 042734  004737  055370                    JSR     PC,SETVV        ;GO SET VOLUME VALID
      042740  000402                            BR      310$            ;BRANCH TO 310$ IF NO ERROR
      042742  104000                            EMT
 4636 042744  000463                            BR      330$
 4637
 4638                                  ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4639                                  ;ADDRESS  AND LOAD SEEK COMMAND
 4640 042746                          310$:
 4641 042746  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4642 042754  012760  000000  000006            MOV     #0,RMDA(R0)     ;LOAD RMDA
 4643 042762  010360  000034                    MOV     R3,RMDC(R0)     ;LOAD RMDC
 4644 042766  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
 4645 042774  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
 4646 043002  012760  000005  000000            MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
 4647 043010  012702  043130                    MOV     #400$,R2        ;INITIALIZE TABLE POINTER
 4648
 4649                                  ;VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
 4650 043014                          315$:
      043014  016037  000040  001142            MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 4651 043022  042737  150000  001142            BIC     #RQA!RQB!TST,$BDDAT
 4652 043030  011237  001140                    MOV     (R2),$GDDAT
 4653 043034  050337  001140                    BIS     R3,$GDDAT       ;OR CYLINDER ADDRESS IN
 4654 043040  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
 4655 043046  001407                            BEQ     320$            ;BRANCH IF TAG BUS OK
 4656 043050  010037  001136                    MOV     R0,$BDADR
 4657 043054  062737  000040  001136            ADD     #RMMR2,$BDADR
 4658 043062  104256                            EMT     256
 4659 043064  000413                            BR      330$
 4660
 4661                                  ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
 4662 043066                          320$:
 4663 043066  062702  000002                    ADD     #2,R2
 4664 043072  005712                            TST     (R2)
 4665 043074  100407                            BMI     330$            ;EXIT IF ENTRY NEGATIVE
 4666
 4667                                  ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
 4668 043076  012760  151001  000024            MOV     #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0)        ;LOAD RMMR1
 4669 043104  012760  041401  000024            MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
 4670 043112  000740                            BR      315$
```

```
                                        ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
043114                                  330$:
043114  006303                                  ASL     R3              ;SHIFT TO NEXT CYLINDER
043116  020327  002000                          CMP     R3,#1024.
043122  103001                                  BHIS    340$            ;EXIT IF WAS DONE
043124  000703                                  BR      300$            ;TEST NEXT CYLINDER
043126  000416                          340$:   BR      500$            ;JUMP OVER TABLE

                                        ;TABLE OF TAG BUS CONTROL AND BIT VALUES
043130                                  400$:
043130  001777                                  .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
043132  001777                                  .WORD   1777
043134  004000                                  .WORD   CC              ;CONTROL BITS ENABLED, BIT 6 ON
043136  004000                                  .WORD   CC
043140  024000                                  .WORD   TAG!CC          ;TAG COMES ON
043142  024000                                  .WORD   TAG!CC
043144  024000                                  .WORD   TAG!CC
043146  024000                                  .WORD   TAG!CC
043150  024000                                  .WORD   TAG!CC
043152  024000                                  .WORD   TAG!CC
043154  004000                                  .WORD   CC              ;TAG GOES OFF
043156  004000                                  .WORD   CC
043160  001777                                  .WORD   1777            ;CONTROL BITS DISABLED

043162  177777                                  .WORD   -1              ;END OF TABLE

043164                                  500$:                           ;END OF TEST

                                        ;;*****************************************************************
                                        ;*TEST 114      SEARCH TEST

                                        ;;****************************** ************************************
043164                                  TST114:
043164  000004                                  SCOPE                   ;SCOPE CALL
043166  000240                                  NOP
043170  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
043174  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
043200  013701  001462                          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
043204  012737  000114  001226                  MOV     #114,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

                                        ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION

043212  004737  055370                          JSR     PC,SETVV        ;GO SET VOLUME VALID
043216  000403                                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
043220  104000                                  EMT
043222  000137  045260                          JMP     330$

                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEARCH COMMAND
043226                                  10$:
043226  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043234  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
043242  012760  000000  000034                  MOV     #0,RMDC(R0)     ;LOAD RMDC
043250  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
043256  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
043264  012760  000031  000000                  MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
```

```
                                        ;STEP COMMAND SEQUENCER TO SEARCH COM (2 CLOCKS)
043272  012702  000002                          MOV     #2,R2
043276                                  20$:
043276  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
043304  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043312  005302                                  DEC     R2
043314  001370                                  BNE     20$

                                        ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
043316  012760  040001  000024                  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
043324  012702  000002                          MOV     #2,R2
043330                                  30$:
043330  012760  140001  000024                  MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
043336  012760  040001  000024                  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
043344  005302                                  DEC     R2
043346  001370                                  BNE     30$

                                        ;VERIFY THAT OPI IS SET
043350  016037  000014  001142                  MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
043356  042737  157777  001142                  BIC     #^COPI,$BDDAT
043364  001011                                  BNE     40$
043366  012737  020000  001140                  MOV     #OPI,$GDDAT
043374  010037  001136                          MOV     R0,$BDADR
043400  062737  000014  001136                  ADD     #RMER1,$BDADR
043406  104257                                  EMT     257

043410  012737  043416  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY THAT SEARCH ABORTS DURING EXECUTION
043416                                  50$:
043416  004737  055370                          JSR     PC,SETVV        ;GO SET VOLUME VALID
043422  000403                                  BR      60$             ;BRANCH TO 60$ IF NO ERROR
043424  104000                                  EMT
043426  000137  045260                          JMP     330$

                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEARCH COMMAND
043432                                  60$:
043432  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043440  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
043446  012760  000000  000034                  MOV     #0,RMDC(R0)     ;LOAD RMDC
043454  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
043462  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
043470  012760  000031  000000                  MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1

                                        ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
043476  012702  000003                          MOV     #3,R2
043502                                  70$:
043502  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
043510  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043516  005302                                  DEC     R2
043520  001370                                  BNE     70$

                                        ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
043522  012760  041101  000024                  MOV     #DMD!MUR.DBEN.MDF,RMMR1(R0)     ;LOAD RMMR1
```

```
                                        ;STEP 2 CLOCKS AND VERIFY GO IS RESET
043530  012702  000002                          MOV     #2,R2
043534                                  80$:
043534  012760  141101  000024                  MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
043542  012760  041101  000024                  MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
043550  005302                                  DEC     R2
043552  001370                                  BNE     80$
043554  016037  000000  001142                  MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
043562  042737  177776  001142                  BIC     #^CGO,$BDDAT
043570  001405                                  BEQ     90$
043572  010037  001136                          MOV     R0,$BDADR
043576  005037  001140                          CLR     $GDDAT
043602  104260                                  EMT     260

                                        ;(THE OTHER TWO ABORT TESTS IN THE COMMAND SEQUENCER ARE TESTED
                                        ;DURING DATA COMMAND TESTS)

043604  012737  043612  001124        90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
043612                                  100$:
043612  004737  055370                          JSR     PC,SETVV        ;GO SET VOLUME VALID
043616  000403                                  BR      110$            ;BRANCH TO 110$ IF NO ERROR
043620  104000                                  EMT
043622  000137  045260                          JMP     330$

                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEARCH COMMAND
043626                                  110$:
043626  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043634  012760  000000  000006                  MOV     #0,RMDA(R0)     ;LOAD RMDA
043642  012760  000000  000034                  MOV     #0,RMDC(R0)     ;LOAD RMDC
043650  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
043656  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
043664  012760  000031  000000                  MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1

                                        ;STEP THE COMMAND SEQUENCER
043672  012702  000023                          MOV     #19.,R2
043676                                  120$:
043676  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
043704  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043712  005302                                  DEC     R2
043714  001370                                  BNE     120$

                                        ;VERIFY THAT OPI IS SET
043716  016037  000014  001142                  MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
043724  042737  157777  001142                  BIC     #^COPI,$BDDAT
043732  001011                                  BNE     130$
043734  010037  001136                          MOV     R0,$BDADR
043740  062737  000014  001136                  ADD     #RMER1,$BDADR
043746  012737  020000  001140                  MOV     #OPI,$GDDAT
043754  104261                                  EMT     261

043756  012737  043764  001124        130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY ATA SETS IF DRIVE COMPLETES SEARCH (ON CYLINDER AND
                                        ;SECTOR COMPARE SETS)
```

```
 4823 043764                          150$:
 4824 043764  004737  055370                JSR     PC,SETVV        ;GO SET VOLUME VALID
      043770  000403                        BR      160$            ;BRANCH TO 160$ IF NO ERROR
      043772  104000                        EMT
 4825 043774  000137  045260                JMP     330$
 4826
 4827                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4828                                 ;ADDRESS,  AND LOAD SEARCH COMMAND
 4829 044000                          160$:
 4830 044000  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4831 044006  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
 4832 044014  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
 4833 044022  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
 4834 044030  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
 4835 044036  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 4836
 4837                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
 4838 044044  012702  000021                MOV     #17.,R2
 4839 044050                          170$:
      044050  012760  141401  000024        MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
 4840 044056  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4841 044064  005302                        DEC     R2
 4842 044066  001370                        BNE     170$
 4843
 4844                                 ;DROP ON CYLINDER TO RESET LATCH,AND RAISE INDEX PULSE
 4845                                 ;TO SET FORMAT CHANGE FLOP
 4846 044070  012760  041005  000024        MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
 4847
 4848                                 ;RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
 4849 044076  012760  051401  000024        MOV     #DMD!MUR!DBEN!MOC.DTO,RMMR1(R0) ;LOAD RMMR1
 4850
 4851                                 ;STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
 4852 044104  012702  000002                MOV     #2,R2
 4853 044110                          180$:
 4854 044110  012760  151401  000024        MOV     #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4855 044116  012760  051401  000024        MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
 4856 044124  005302                        DEC     R2
 4857 044126  001370                        BNE     180$
 4858
 4859                                 ;FORCE SECTOR COMPARE BY CLOCKING SECTOR PULSE WITH SECTOR COMPARE
 4860                                 ;ACTIVE
 4861 044130  012760  051403  000024        MOV     #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(R0)     ;LOAD RMMR1
 4862 044136  012760  051443  000024        MOV     #DMD!MUR!MOC!DBEN!DTO!MSC!MS,RMMR1(R0)  ;LOAD RMMR1
 4863 044144  012760  051403  000024        MOV     #DMD!MUR!MOC!DBEN!DTO!MSC,RMMR1(R0)     ;LOAD RMMR1
 4864
 4865                                 ;CLOCK SEQUENCER TO SET ATA (3 CLOCKS)
 4866 044152  012702  000003                MOV     #3,R2           ;R2 = CLOCK COUNT
 4867 044156                          185$:
      044156  012760  151401  000024        MOV     #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4868 044164  012760  051401  000024        MOV     #DMD!MUR!MOC!DBEN!DTO,RMMR1(R0) ;LOAD RMMR1
 4869 044172  005302                        DEC     R2
 4870 044174  001370                        BNE     185$
 4871
 4872                                 ;VERIFY ATA IS SET
 4873 044176  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 4874 044204  042737  077777  001142        BIC     #^CATA,$BDDAT
 4875 044212  001011                        BNE     190$
```

```
 4876 044214  012737  100000  001140          MOV     #ATA,$GDDAT
 4877 044222  010037  001136                  MOV     R0,$BDADR
 4878 044226  062737  000012  001136          ADD     #RMDS,$BDADR
 4879 044234  104262                          EMT     262
 4880
 4881 044236  012737  044244  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4882
 4883                                 ;VERIFY THAT SEARCH ABORTS AFTER EXECUTION DURING SEARCH SEEK LOOP
 4884 044244                          200$:
 4885 044244  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      044250  000403                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
      044252  104000                          EMT
 4886 044254  000137  045260                  JMP     330$
 4887
 4888                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4889                                 ;ADDRESS  AND LOAD SEARCH COMMAND
 4890 044260                          210$:
 4891 044260  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4892 044266  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 4893 044274  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 4894 044302  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 4895 044310  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 4896 044316  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 4897
 4898                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
 4899 044324  012702  000021                  MOV     #17.,R2
 4900 044330                          220$:
 4901 044330  012760  141401  000024          MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 4902 044336  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4903 044344  005302                          DEC     R2
 4904 044346  001370                          BNE     220$
 4905
 4906                                 ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
 4907 044350  012760  041001  000024          MOV     #DMD'MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4908
 4909                                 ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
 4910 044356  012702  000007                  MOV     #7,R2
 4911 044362                          230$:
 4912 044362  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4913 044370  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4914 044376  005302                          DEC     R2
 4915 044400  001370                          BNE     230$
 4916
 4917                                 ;VERIFY THAT GO IS STILL SET
 4918 044402  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 4919 044410  042737  177776  001142          BIC     #^CGO,$BDDAT
 4920 044416  001006                          BNE     240$
 4921 044420  012737  000001  001140          MOV     #GO,$GDDAT
 4922 044426  010037  001136                  MOV     R0,$BDADR
 4923 044432  104263                          EMT     263
 4924
 4925                                 ;SET SEEK INCOMPLETE ERROR
 4926 044434                          240$:
      044434  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
 4927
 4928                                 ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
 4929 044442  012702  000003                  MOV     #3,R2
```

```
 4930 044446                          250$:
      044446  012760  141201  000024          MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
 4931 044454  012760  041201  000024          MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
 4932 044462  005302                          DEC     R2
 4933 044464  001370                          BNE     250$
 4934 044466  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 4935 044474  042737  177776  001142          BIC     #^CGO,$BDDAT
 4936 044502  001405                          BEQ     260$
 4937 044504  010037  001136                  MOV     R0,$BDADR
 4938 044510  005037  001140                  CLR     $GDDAT
 4939 044514  104264                          EMT     264
 4940
 4941 044516  012737  044524  001124  260$:  MOV     #265$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4942
 4943                                 ;VERIFY THAT SEARCH ABORTS DURING SECTOR COMPARE LOOP
 4944 044524                          265$:
 4945 044524  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      044530  000403                          BR      270$            ;BRANCH TO 270$ IF NO ERROR
      044532  104000                          EMT
 4946 044534  000137  045260                  JMP     330$
 4947
 4948                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 4949                                 ;ADDRESS,  AND LOAD SEARCH COMMAND
 4950 044540                          270$:
 4951 044540  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)      ;LOAD RMMR1
 4952 044546  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 4953 044554  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 4954 044562  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 4955 044570  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 4956 044576  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 4957
 4958                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
 4959 044604  012702  000021                  MOV     #17.,R2
 4960 044610                          275$:
      044610  012760  141401  000024          MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)         ;LOAD RMMR1
 4961 044616  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4962 044624  005302                          DEC     R2
 4963 044626  001370                          BNE     275$
 4964
 4965                                 ;DROP ON CYLINDER TO RESET LATCH, AND RAISE INDEX PULSE
 4966                                 ;TO SET FORMAT CHANGE FLOP
 4967 044630  012760  041005  000024          MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
 4968
 4969                                 ;RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
 4970 044636  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
 4971
 4972                                 ;STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
 4973 044644  012702  000002                  MOV     #2,R2
 4974 044650                          280$:
 4975 044650  012760  151401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4976 044656  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
 4977 044664  005302                          DEC     R2
 4978 044666  001370                          BNE     280$
 4979
 4980                                 ;VERIFY THAT SEARCH ENABLE IS ON DURING SECTOR COMPARE LOOP
 4981 044670  012702  000004                  MOV     #4,R2           ;R2 = CLOCK COUNT
 4982 044674                          281$:
```

```
 4983 044674  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 4984 044702  042737  173777  001142          BIC     #^CESRC,$BDDAT
 4985 044710  001411                          BEQ     282$            ;BRANCH IF SEARCH NOT ENABLED
 4986 044712  012760  151401  000024          MOV     #DMD!MUR!MOC!DBEN!DTO!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4987 044720  012760  051401  000024          MOV     #DMD!MUR!MOC!DBEN!DTO,RMMR1(R0) ;LOAD RMMR1
 4988 044726  005302                          DEC     R2
 4989 044730  001361                          BNE     281$
 4990 044732  000411                          BR      283$
 4991 044734  012737  004000  001140  282$:   MOV     #ESRC,$GDDAT
 4992 044742  010037  001136                  MOV     R0,$BDADR
 4993 044746  062737  000024  001136          ADD     #RMMR1,$BDADR
 4994 044754  104265                          EMT     265
 4995
 4996                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
 4997 044756                          283$:
      044756  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC.DTO!MDF,RMMR1(R0)     ;LOAD RMMR1
 4998
 4999                                 ;STEP 2 CLOCKS AND VERIFY GO IS RESET
 5000 044764  012702  000002                  MOV     #2,R2           ;R2 = CLOCK COUNT
 5001 044770                          285$:
      044770  012760  151501  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5002 044776  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC!DTO!MDF,RMMR1(R0)     ;LOAD RMMR1
 5003 045004  005302                          DEC     R2
 5004 045006  001370                          BNE     285$
 5005 045010  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 5006 045016  042737  177776  001142          BIC     #^CGO,$BDDAT
 5007 045024  001406                          BEQ     290$
 5008 045026  012737  000000  001140          MOV     #0,$GDDAT
 5009 045034  010037  001136                  MOV     R0,$BDADR
 5010 045040  104260                          EMT     260
 5011
 5012 045042  012737  045054  001124  290$:   MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 5013 045050  012703  000001                  MOV     #1,R3           ;INITIALIZE CYLINDER ADDRESS
 5014
 5015                                 ;VERIFY THE TAG BUS DURING SEARCH
 5016 045054                          300$:
 5017 045054  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      045060  000402                          BR      310$            ;BRANCH TO 310$ IF NO ERROR
      045062  104000                          EMT
 5018 045064  000475                          BR      330$
 5019
 5020                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 5021                                 ;ADDRESS  AND LOAD SEARCH COMMAND
 5022 045066                          310$:
 5023 045066  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5024 045074  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5025 045102  010360  000034                  MOV     R3,RMDC(R0)     ;LOAD RMDC
 5026 045106  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5027 045114  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5028 045122  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 5029
 5030                                 ;;*****************************************************************
 5031                                 ;       MOV     #400$,R2        ;INITIALIZE TABLE POINTER
 5032
 5033                                                                 ;HARDWARE ECO CHANGE TO THE PLA OF THE
 5034                                                                 ;CS BOARD.
 5035 045130  012702  000011                  MOV     #9.,R2          ;CLOCK THE SEQUENCER THRU THE FIRST
```

```
                                                        ;9. COMMAND SEQUENCES TO ALLOW THE PROGRAM
                                                        ;TO RUN WITH OR WITHOUT THE ECO.
                                        312$:
                                                MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                                MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
                                                DEC     R2                      ;DONE 9. CLOCKS ?
                                                BNE     312$                    ;NO !!
                                                MOV     #450$,R2                ;INITIALIZE NEW TABLE POINTER
                                        ;;**************************************************************
                                        ;VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
                                        315$:
                                                MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
                                                BIC     #RQA!RQB!TST,$BDDAT
                                                MOV     (R2),$GDDAT
                                                BIS     R3,$GDDAT               ;OR CYLINDER ADDRESS IN
                                                CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED AND RECEIVED
                                                BEQ     320$                    ;BRANCH IF TAG BUS OK
                                                MOV     R0,$BDADR
                                                ADD     #RMMR2,$BDADR
                                                EMT     266
                                                BR      340$

                                        ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
                                        320$:
                                                ADD     #2,R2
                                                TST     (R2)
                                                BMI     330$                    ;EXIT IF ENTRY NEGATIVE

                                        ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
                                                MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                                MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
                                                BR      315$

                                        ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
                                        330$:
                                                ASL     R3                      ;SHIFT TO NEXT CYLINDER
                                                CMP     R3,#1024.
                                                BHIS    340$                    ;EXIT IF WAS DONE
                                                BR      300$                    ;TEST NEXT CYLINDER
                                        340$:   BR      500$                    ;JUMP OVER TABLE

                                        ;TABLE OF TAG BUS CONTROL AND BIT VALUES
                                        400$:
                                                .WORD   1777                    ;BUS BITS AT HIGH IMPEDENCE STATE
                                                .WORD   1777
                                                .WORD   1777
                                                .WORD   1777
                                                .WORD   1777
                                                .WORD   CC                      ;CONTROL BITS ENABLED, BIT 6 ON
                                                .WORD   CC
                                                .WORD   TAG!CC                  ;TAG COMES ON
                                                .WORD   TAG!CC
                                        450$:                                   ;START TABLE HERE FOR HARDWARE ECO CHANGE
                                                .WORD   TAG!CC
                                                .WORD   TAG!CC
                                                .WORD   TAG!CC
```

```
 5092 045324  024000                        .WORD   TAG!CC
 5093 045326  177777                        .WORD   -1              ;END TABLE HERE FOR HARDWARE ECO CHANGE
 5094
 5095                                ;       .WORD   CC              ;TAG GOES OFF
 5096 045330  004000                        .WORD   CC
 5097 045332  001777                        .WORD   1777            ;CONTROL BITS DISABLED
 5098 045334  001777                        .WORD   1777
 5099 045336  001777                        .WORD   1777
 5100 045340  001777                        .WORD   1777
 5101
 5102 045342  177777                        .WORD   -1              ;END OF TABLE
 5103
 5104 045344                         500$:                           ;END OF TEST
 5105
 5106                                ;;*****************************************************************
                                     ;*TEST 115       SEARCH TIMEOUT TEST
                                     ;;*****************************************************************
      045344                         TST115:
      045344  000004                        SCOPE                   ;SCOPE CALL
      045346  000240                        NOP
      045350  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
      045354  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      045360  013701  001462                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      045364  012737  000115  001226        MOV     #115,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 5107
 5108 045372  004737  055370                JSR     PC,SETVV        ;GO SET VOLUME VALID
      045376  000402                        BR      10$             ;BRANCH TO 10$ IF NO ERROR
      045400  104000                        EMT
 5109 045402  000550                        BR      90$
 5110
 5111                                ;ENABLE DEBUG CLOCK AND LOAD SEARCH COMMAND
 5112 045404                         10$:
 5113 045404  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5114 045412  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
 5115 045420  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
 5116 045426  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
 5117 045434  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
 5118 045442  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 5119
 5120                                ;EXECUTE SEARCH TO TEST FOR ON LATCH RESET (17 CLOCKS)
 5121 045450  012702  000021                MOV     #17.,R2
 5122 045454                         20$:
      045454  012760  141401  000024        MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5123 045462  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5124 045470  005302                        DEC     R2
 5125 045472  001370                        BNE     20$
 5126
 5127                                ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER
 5128 045474  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5129 045502  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5130
 5131                                ;STEP COMMAND SEQUENCER TO SECTOR COMPARE LOOP (2 CLOCKS)
 5132 045510  012702  000002                MOV     #2,R2
 5133 045514                         30$:
      045514  012760  151401  000024        MOV     #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(R0)    ;LOAD RMMR1
 5134 045522  012760  051401  000024        MOV     #DMD!DBEN!MUR!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
```

```
 5135 045530  005302                          DEC     R2
 5136 045532  001370                          BNE     30$
 5137
 5138                                 ;THE COMMAND SEQUENCER IS NOW IN THE SECTOR COMPARE LOOP.  STEP
 5139                                 ;THROUGH THE LOOP AND VERIFY SEARCH IS ENABLED.
 5140 045534  012702  000005                  MOV     #5,R2
 5141 045540                          40$:
      045540  012760  151401  000024          MOV     #DMD!DBEN!MUR!MOC!DBCK!DTO,RMMR1(R0)    ;LOAD RMMR1
 5142 045546  012760  051401  000024          MOV     #DMD!DBEN!MUR!MOC!DTO,RMMR1(R0) ;LOAD RMMR1
 5143 045554  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5144 045562  042737  173777  001142          BIC     #^CESRC,$BDDAT
 5145 045570  001403                          BEQ     50$             ;BRANCH IF SEARCH NOT ENABLED
 5146 045572  005302                          DEC     R2
 5147 045574  001361                          BNE     40$
 5148 045576  000412                          BR      60$
 5149 045600  012737  004000  001140  50$:    MOV     #ESRC,$GDDAT    ;SETUP ERROR MSG
 5150 045606  010037  001136                  MOV     R0,$BDADR
 5151 045612  062737  000024  001136          ADD     #RMMR1,$BDADR
 5152 045620  104265                          EMT     265
 5153 045622  000440                          BR      90$
 5154
 5155                                 ;DROP DTO TO ENABLE SEARCH TIMEOUT AND WAIT FOR OPI TO SET.
 5156 045624                          60$:
 5157 045624  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5158 045632  012737  070000  001530          MOV     #70000,WATCH    ;SET WATCHDOG TIMER VALUE
      045640  004777  133666                  JSR     PC,@CLOCK       ;START THE CLOCK
 5159 045644                          70$:
      045644  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 5160 045652  042737  157777  001142          BIC     #^COPI,$BDDAT
 5161 045660  001017                          BNE     80$
 5162 045662  005737  001530                  TST     WATCH
 5163 045666  001366                          BNE     70$
 5164 045670  004777  133640                  JSR     PC,@STOP        ;STOP THE CLOCK
 5165 045674  012737  020000  001140          MOV     #OPI,$GDDAT     ;SETUP ERROR MSG
 5166 045702  010037  001136                  MOV     RC,$BDADR
 5167 045706  062737  000014  001136          ADD     #RMER1,$BDADR
 5168 045714  104267                          EMT     267
 5169 045716  000402                          BR      90$
 5170 045720                          80$:
      045720  004777  133610                  JSR     PC,@STOP        ;STOP THE CLOCK
 5171
 5172 045724                          90$:                            ;END OF TEST
 5173
 5174                                 ;;*****************************************************************
                                      ;*TEST 116       DATA COMMAND TESTS (1)

                                      ;;*****************************************************************
      045724                          TST116:
      045724  000004                          SCOPE                   ;SCOPE CALL
      045726  000240                          NOP
      045730  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      045734  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      045740  013701  001462                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      045744  012737  000116  001226          MOV     #116,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 5175
 5176                                 ;VERIFY DATA COMMAND SETS OPI IF DRIVE NOT READY
 5177
```

```
 5178 045752  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
      045756  000402                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
      045760  104000                  EMT
 5179 045762  000471                  BR      40$
 5180
 5181                          ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
 5182 045764                   10$:
 5183 045764  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5184 045772  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5185 046000  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5186 046006  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5187 046014  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 5188 046022  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
 5189
 5190                          ;STEP COMMAND SEQUENCER TO UNIT READY TEST (3 CLOCKS)
 5191 046030  012702  000003          MOV     #3,R2
 5192 046034                   20$:
      046034  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5193 046042  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5194 046050  005302                  DEC     R2
 5195 046052  001370                  BNE     20$
 5196
 5197                          ;DROP UNIT READY
 5198 046054  012760  040401  000024          MOV     #DMD!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
 5199
 5200                          ;STEP SEQUENCER AND VERIFY OPI SETS (2 CLOCKS)
 5201 046062  012702  000002          MOV     #2,R2
 5202 046066                   30$:
      046066  012760  140401  000024          MOV     #DMD!MOC!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 5203 046074  012760  040401  000024          MOV     #DMD!MOC!DBEN,RMMR1(R0) ;LOAD RMMR1
 5204 046102  005302                  DEC     R2
 5205 046104  001370                  BNE     30$
 5206 046106  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 5207 046114  042737  157777  001142          BIC     #^COPI,$BDDAT
 5208 046122  001011                  BNE     40$
 5209 046124  012737  020000  001140          MOV     #OPI,$GDDAT
 5210 046132  010037  001136          MOV     R0,$BDADR
 5211 046136  062737  000014  001136          ADD     #RMER1,$BDADR
 5212 046144  104270                  EMT     270
 5213
 5214 046146  012737  046154  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR TEST
 5215
 5216                          ;VERIFY DATA COMMAND ABORTS AT LOCATION 129
 5217 046154                   50$:
 5218 046154  004737  055370          JSR     PC,SETVV        ;GO SET VOLUME VALID
      046160  000402                  BR      60$             ;BRANCH TO 60$ IF NO ERROR
      046162  104000                  EMT
 5219 046164  000576                  BR      150$
 5220
 5221                          ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
 5222 046166                   60$:
 5223 046166  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5224 046174  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5225 046202  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5226 046210  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5227 046216  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 5228 046224  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
```

```
                                        ;STEP COMMAND SEQUENCER TO ABORT TEST AT LOCATION 129 (4 CLOCKS)
046232  012702  000004                          MOV     #4,R2
                                        70$:
046236                                          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
046236  012760  141401  000024
046244  012760  041401  000024                  MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
046252  005302                                  DEC     R2
046254  001370                                  BNE     70$

                                        ;SET DEVICE FAULT TO CAUSE ABORT CONDITION
046256  012760  041501  000024                  MOV     #DMD!MUR!MOC.DBEN!MDF,RMMR1(R0) ;LOAD RMMR1

                                        ;STEP THE SEQUENCER THROUGH THE TEST FOR ABORT (1 CLOCK)
046264  012702  000001                          MOV     #1,R2
                                        80$:
046270                                          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
046270  012760  141501  000024
046276  012760  041501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
046304  005302                                  DEC     R2
046306  001370                                  BNE     80$

                                        ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
                                        ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
046310  012702  000020                          MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
                                        85$:
046314                                          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
046314  012760  045501  000024
046322  012760  041501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
046330  016037  000024  001142                  MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
046336  042737  157777  001142                  BIC     #^CEBL,$BDDAT
046344  001014                                  BNE     90$             ;BRANCH IF EBL IS SET

046346  005302                                  DEC     R2
046350  001361                                  BNE     85$             ;CONTINUE BIT CLOCKS IF COUNT NOT 0
046352  012737  020000  001140                  MOV     #EBL,$GDDAT
046360  010037  001136                          MOV     R0,$BDADR
046364  062737  000024  001136                  ADD     #RMMR1,$BDADR
046372  104271                                  EMT     271
046374  000472                                  BR      150$

                                        ;STEP THE SEQUENCER THROUGH ITS TEST FOR EBL (2 CLOCKS)
046376                                  90$:
046376  012702  000002                          MOV     #2,R2
046402                                  100$:
046402  012760  141501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
046410  012760  041501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
046416  005302                                  DEC     R2
046420  001370                                  BNE     100$

                                        ;ABORT EBL SHOULD NOW BE INACTIVE - FORCE BIT CLOCK USING THE
                                        ;MAINTENANCE REGISTER TO RESET EBL (16 BIT CLOCKS)
046422  012702  000020                          MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
046426                                  110$:
046426  012760  045501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
046434  012760  041501  000024                  MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
046442  005302                                  DEC     R2
046444  001370                                  BNE     110$            ;ISSUE 16 BIT CLOCKS THEN TEST

                                        ;VERIFY EBL IS NOW RESET
```

```
                                        120$:
046446
046446  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
046454  042737  157777  001142          BIC     #^CEBL,$BDDAT
046462  001411                          BEQ     130$            ;BRANCH IF EBL IS RESET
046464  005037  001140                  CLR     $GDDAT
046470  010037  001136                  MOV     R0,$BDADR
046474  062737  000024  001136          ADD     #RMMR1,$BDADR
046502  104273                          EMT     273
046504  000426                          BR      150$

                                        ;VERIFY GO RESETS WITHIN 4 CLOCK CYCLES
046506                                  130$:
046506  012702  000004                  MOV     #4,R2
046512                                  140$:
046512  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
046520  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
046526  005302                          DEC     R2
046530  001370                          BNE     140$
046532  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
046540  042737  177776  001142          BIC     #^CGO,$BDDAT
046546  001405                          BEQ     150$
046550  005037  001140                  CLR     $GDDAT
046554  010037  001136                  MOV     R0,$BDADR
046560  104274                          EMT     274

046562  012737  046570  001124  150$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY SEQUENCER BRANCHES TO SEEK WHEN RUN AND GO FLOP SETS
046570                                  200$:
046570  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
046574  000402                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
046576  104000                          EMT
046600  000512                          BR      250$

                                        ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
046602                                  210$:
046602  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
046610  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
046616  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
046624  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
046632  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
046640  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                        ;MOVE SEQUENCER TO TEST FOR RUN AND GO AT LOCATION 130 (5 CLOCKS)
046646  012702  000005                  MOV     #5,R2
046652                                  220$:
046652  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
046660  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
046666  005302                          DEC     R2
046670  001370                          BNE     220$

                                        ;VERIFY RUN AND GO IS SET
046672  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
046700  042737  137777  001142          BIC     #^CRG,$BDDAT
046706  001012                          BNE     230$
046710  012737  040000  001140          MOV     #RG,$GDDAT
046716  010037  001136                  MOV     R0,$BDADR
```

```
 5335 046722  062737  000024  001136          ADD     #RMMR1,$BDADR
 5336 046730  104275                          EMT     275
 5337 046732  000435                          BR      250$
 5338
 5339                                 ;VERIFY THAT CYLINDER TAG COMES UP IN ONE CLOCK CYCLE
 5340 046734                          230$:
 5341 046734  012702  000001                  MOV     #1,R2
 5342 046740                          240$:
      046740  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5343 046746  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5344 046754  005302                          DEC     R2
 5345 046756  001370                          BNE     240$
 5346 046760  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 5347
 5348                                 ;;*****************************************************************
 5349                                 ;       BIC     #RQA!RQB!TAG,$BDDAT
 5350                                                                 ;THE FOLLOWING CODE WAS ADDED
 5351                                                                 ;TO ALLOW THE PROGRAM TO RUN WITH
 5352                                                                 ;OR WITHOUT THE ECO TO THE CS BOARD.
 5353
 5354 046766  042737  162000  001142          BIC     #RQA!RQB!TAG!CH,$BDDAT  ;HARDWARE ECO CHANGE CAUSES CC AND
 5355                                                                 ;CH TO SET AT THE SAME TIME
 5356                                 ;;*****************************************************************
 5357
 5358 046774  012737  004000  001140          MOV     #CC,$GDDAT
 5359 047002  023737  001140  001142          CMP     $GDDAT,$BDDAT
 5360 047010  001406                          BEQ     250$
 5361 047012  010037  001136                  MOV     R0,$BDADR
 5362 047016  062737  000040  001136          ADD     #RMMR2,$BDADR
 5363 047024  104276                          EMT     276
 5364
 5365 047026  012737  047040  001124  250$:   MOV     #260$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 5366
 5367                                 ;VERIFY DATA COMMAND ABORTS AT COMMAND SEQUENCER LOCATIONS 144, 145
 5368 047034  012702  000144                  MOV     #144,R2         ;INITIALIZE TEST LOCATION
 5369 047040                          260$:
 5370 047040  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      047044  000402                          BR      270$            ;BRANCH TO 270$ IF NO ERROR
      047046  104000                          EMT
 5371 047050  000553                          BR      320$
 5372
 5373                                 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 5374 047052                          270$:
 5375 047052  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5376 047060  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5377 047066  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5378 047074  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5379 047102  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 5380 047110  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
 5381
 5382                                 ;WAIT FOR RUN AND GO TO SET
 5383 047116  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
      047124  004777  132402                  JSR     PC,@CLOCK       ;START THE CLOCK
 5384 047130                          280$:
      047130  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5385 047136  042737  137777  001142          BIC     #^CRG,$BDDAT
 5386 047144  001017                          BNE     290$
```

```
5387 047146  005737  001530                TST     WATCH
5388 047152  001366                        BNE     280$
5389 047154  004777  132354                JSR     PC,@STOP        ;STOP THE CLOCK
5390 047160  012737  040000  001140        MOV     #RG,$GDDAT
5391 047166  010037  001136                MOV     R0,$BDADR
5392 047172  062737  000024  001136        ADD     #RMMR1,$BDADR
5393 047200  104275                        EMT     275
5394 047202  000476                        BR      320$
5395 047204                        290$:
     047204  004777  132324                JSR     PC,@STOP        ;STOP THE CLOCK
5396
5397                               ;MOVE COMMAND SEQUENCER TO ABORT TEST (LOCATION 144 OR 145)
5398 047210  012703  000006                MOV     #6,R3           ;SETUP CLOCK COUNT
5399 047214  022702  000144                CMP     #144,R2
5400 047220  001402                        BEQ     300$
5401 047222  012703  000007                MOV     #7,R3
5402 047226                        300$:
     047226  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5403 047234  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5404 047242  005303                        DEC     R3
5405 047244  001370                        BNE     300$
5406
5407                               ;SET DRIVE FAULT TO FORCE ABORT CONDITION
5408 047246  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5409
5410                               ;CLOCK SEQUENCER THROUGH ITS TEST FOR ABORT (1 CLOCK)
5411 047254  012703  000001                MOV     #1,R3
5412 047260                        305$:
     047260  012760  141501  000024        MOV     #DMD!MUR.MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
5413 047266  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5414 047274  005303                        DEC     R3
5415 047276  001370                        BNE     305$            ;ISSUE 2 CLOCKS
5416
5417                               ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
5418                               ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
5419 047300  012702  000020                MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
5420 047304                        306$:
     047304  012760  045501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
5421 047312  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5422 047320  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5423 047326  042737  157777  001142        BIC     #^CEBL,$BDDAT
5424 047334  001013                        BNE     310$            ;BRANCH IF EBL IS SET
5425
5426 047336  005302                        DEC     R2
5427 047340  001361                        BNE     306$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
5428 047342  012737  020000  001140        MOV     #EBL,$GDDAT
5429 047350  010037  001136                MOV     R0,$BDADR
5430 047354  062737  000024  001136        ADD     #RMMR1,$BDADR
5431 047362  104271                        EMT     271
5432 047364  022702  000144        310$:   CMP     #144,R2
5433 047370  001003                        BNE     320$
5434 047372  012702  000145                MOV     #145,R2
5435 047376  000620                        BR      260$
5436
5437 047400  012737  047410  001124  320$: MOV     #330$,$LPERR            ;CHANGE LOOP ON ERROR ADDRESS
5438
5439                               ;VERIFY HEAD TAG DURING DATA COMMAND
```

```
                                        CLR     R2              ;INITIALIZE TRACK ADDRESS = 0
                                330$:
                                        JSR     PC,SETVV        ;GO SET VOLUME VALID
                                        BR      340$            ;BRANCH TO 340$ IF NO ERROR
                                        EMT
                                        BR      400$

                                ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
                                340$:
                                        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
                                        MOV     #0,RMER1(R0)    ;LOAD RMER1
                                        MOV     #0,RMER2(R0)    ;LOAD RMER2
                                        MOV     R2,RMDA(R0)     ;LOAD RMDA
                                        MOV     #0,RMDC(R0)     ;LOAD RMDC
                                        MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                ;WAIT FOR RUN AND GO TO SET
                                        MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
                                        JSR     PC,@CLOCK       ;START THE CLOCK
                                350$:
                                        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
                                        BIC     #^CRG,$BDDAT
                                        BNE     360$
                                        TST     WATCH
                                        BNE     350$
                                        JSR     PC,@STOP        ;STOP THE CLOCK
                                        MOV     #RG,$GDDAT
                                        MOV     R0,$BDADR
                                        ADD     #RMMR1,$BDADR
                                        EMT     275
                                        BR      400$
                                360$:
                                        JSR     PC,@STOP        ;STOP THE CLOCK

                                ;STEP COMMAND SEQUENCER TO HEAD SEQUENCE, LOCATION 156 (17 CLOCKS)
                                        MOV     #17.,R3
                                370$:
                                        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
                                        DEC     R3
                                        BNE     370$

                                ;DROP AND RAISE ON CYLINDER TO RESET ON LATCH
                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1

                                ;;*****************************************************************
                                ;       MOV     #450$,R3        ;INITIALIZE TABLE POINTER

                                                                ;HARDWARE ECO CHANGE TO THE PLA ON THE
                                                                ;CS BOARD.
                                        MOV     #4,R3           ;CLOCK THE SEQUENCER THRU THE FIRST
                                                                ;4 COMMAND SEQUENCES TO ALLOW THE PROGRAM
                                                                ;TO RUN WITH OR WITHOUT THE ECO.
                                372$:
                                        MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                        MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
```

```
5491 047636 005303                          DEC     R3              ;DONE 4 CLOCKS ?
5492 047640 001370                          BNE     372$            ;NO !!
5493 047642 012703 050014                   MOV     #475$,R3        ;INITIALIZE NEW TABLE POINTER
5494                                ;;*************************************************************
5495
5496                                ;VERIFY TAG BUS ACCORDING TO TABLE AND TRACK ADDRESS IN R2
5497 047646                         375$:
5498 047646 016037 000040 001142            MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
5499 047654 042737 150000 001142            BIC     #RQA!RQB!TST,$BDDAT
5500 047662 011337 001140                   MOV     (R3),$GDDAT
5501 047666 010204                          MOV     R2,R4           ;GENERATE EXPECTED TAG BUS
5502 047670 000304                          SWAB    R4
5503 047672 050437 001140                   BIS     R4,$GDDAT
5504 047676 023737 001140 001142            CMP     $GDDAT,$BDDAT
5505 047704 001030                          BNE     390$
5506 047706 012760 141401 000024            MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5507 047714 012760 041401 000024            MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5508
5509                                ;ADVANCE TO NEXT TABLE ENTRY
5510 047722 062703 000002                   ADD     #2,R3
5511 047726 005713                          TST     (R3)
5512 047730 100401                          BMI     380$
5513 047732 000745                          BR      375$
5514
5515                                ;SHIFT TO NEXT TRACK ADDRESS-EXIT LOOP IF DONE
5516 047734                         380$:
5517 047734 062702 000400                   ADD     #TA1,R2         ;ADVANCE TRACK ADDRESS
5518 047740 032737 010000 001330            BIT     #TA16,LSTRK     ;IS IT RM05 ?
5519 047746 001403                          BEQ     385$            ;NO !!
5520 047750 020237 001330                   CMP     R2,LSTRK        ;DONE WITH TRACKS ON RM05 ?
5521 047754 101615                          BLOS    330$            ;NO !!
5522 047756                         385$:
5523 047756 020237 001330                   CMP     R2,LSTRK        ;DONE WITH TRACKS ON RM02/3 ?
5524 047762 101007                          BHI     400$            ;YES, EXIT
5525 047764 000611                          BR      330$
5526
5527                                ;ERROR ON TAG BUS DURING HEAD SEQUENCE
5528 047766                         390$:
5529 047766 010037 001136                   MOV     R0,$BDADR
5530 047772 062737 000040 001136            ADD     #RMMR2,$BDADR
5531 050000 104276                          EMT     276
5532
5533 050002 000440                  400$:   BR      500$            ;JUMP OVER TABLE
5534
5535                                ;TABLE OF TAG BUS DURING HEAD SEQUENCE
5536 050004                         450$:
5537 050004 002000                          .WORD   CH
5538 050006 002000                          .WORD   CH
5539 050010 022000                          .WORD   CH!TAG
5540 050012 022000                          .WORD   CH!TAG
5541 050014                         475$:                           ;START TABLE HERE FOR HARDWARE ECO CHANGE
5542 050014 022000                          .WORD   CH!TAG
5543 050016 022000                          .WORD   CH!TAG
5544 050020 022000                          .WORD   CH!TAG
5545 050022 022000                          .WORD   CH!TAG
5546 050024 177777                          .WORD   -1              ;END TABLE HERE FOR HARDWARE ECO CHANGE
5547
```

```
                                         ;       .WORD   CP
050026  002000                                   .WORD   CH
050030  001777                                   .WORD   1777
050032  001777                                   .WORD   1777
050034  001777                                   .WORD   1777
050036  0C1777                                   .WORD   1777
050040  001777                                   .WORD   1777
050042  001777                                   .WORD   1777
050044  001777                                   .WORD   1777
050046  001777                                   .WORD   1777
050050  001777                                   .WORD   1777
050052  001777                                   .WORD   1777
050054  001777                                   .WORD   1777
050056  001777                                   .WORD   1777
050060  001777                                   .WORD   1777
050062  001777                                   .WORD   1777
050064  001777                                   .WORD   1777
050066  001777                                   .WORD   1777
050070  001777                                   .WORD   1777
050072  001777                                   .WORD   1777
050074  001777                                   .WORD   1777
050076  001777                                   .WORD   1777
050100  001777                                   .WORD   1777

050102  177777                                   .WORD   -1              ;END OF TABLE

050104                                   500$:                           ;END OF TEST

                                         ;;*****************************************************************
                                         ;*TEST 117      DATA COMMAND TESTS (2)

                                         ;;*****************************************************************
050104                                   TST117:
050104  000004                                   SCOPE                   ;SCOPE CALL
050106  000240                                   NOP
050110  012706  001100                           MOV     #STACK,SP       ;LOAD THE STACK POINTER
050114  013700  001276                           MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
050120  013701  001462                           MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
050124  012737  000117  001226                   MOV     #117,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

                                         ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT RESET

050132  004737  055370                           JSR     PC,SETVV        ;GO SET VOLUME VALID
050136  000402                                   BR      10$             ;BRANCH TO 10$ IF NO ERROR
050140  104000                                   EMT
050142  000514                                   BR      60$

                                         ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND DURING CYLINDER SEQUENCE
050144                                   10$:
050144  012760  041401  000024                   MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
050152  012760  000000  000014                   MOV     #0,RMER1(R0)    ;LOAD RMER1
050160  012760  000000  000042                   MOV     #0,?MER2(R0)    ;LOAD RMER2
050166  012760  000000  000036                   MOV     #0,RMDA(R0)     ;LOAD RMDA
050174  012760  000000  000034                   MOV     #0,RMDC(R0)     ;LOAD RMDC
050202  012760  000071  000000                   MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                         ;WAIT FOR RUN AND GO TO SET
```

```
5593 050210  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
     050216  004777  131310                  JSR     PC,@CLOCK       ;START THE CLOCK
5594 050222                          20$:
     050222  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5595 050230  042737  137777  001142          BIC     #^CRG,$BDDAT
5596 050236  001017                          BNE     30$
5597 050240  005737  001530                  TST     WATCH
5598 050244  001366                          BNE     20$
5599 050246  004777  131262                  JSR     PC,@STOP        ;STOP THE CLOCK
5600 050252  012737  040000  001140          MOV     #RG,$GDDAT
5601 050260  010037  001136                  MOV     R0,$BDADR
5602 050264  062737  000024  001136          ADD     #RMMR1,$BDADR
5603 050272  104275                          EMT     275
5604 050274  000437                          BR      60$
5605 050276                          30$:
     050276  004777  131232                  JSR     PC,@STOP        ;STOP THE CLOCK
5606
5607                                 ;STEP COMMAND SEQUENCER AND VERIFY OPI SETS (19 CLOCKS)
5608 050302  012702  000023                  MOV     #19.,R2
5609 050306                          40$:
     050306  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5610 050314  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5611 050322  005302                          DEC     R2
5612 050324  001370                          BNE     40$
5613 050326  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
5614 050334  042737  157777  001142          BIC     #^COPI,$BDDAT
5615 050342  001011                          BNE     50$             ;BRANCH IF OPI SET
5616 050344  012737  020000  001140          MOV     #OPE,$GDDAT
5617 050352  010037  001136                  MOV     R0,$BDADR
5618 050356  062737  000014  001136          ADD     #RMER1,$BDADR
5619 050364  104277                          EMT     277
5620 050366  012737  050374  001124  50$:    MOV     #60$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
5621
5622                                 ;VERIFY DATA COMMAND ABORTS DURING SEEK WAIT LOOP
5623 050374                          60$:
5624 050374  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     050400  000402                          BR      70$             ;BRANCH TO 70$ IF NO ERROR
     050402  104000                          EMT
5625 050404  000567                          BR      140$
5626
5627                                 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
5628 050406                          70$:
5629 050406  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5630 050414  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
5631 050422  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
5632 050430  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5633 050436  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
5634 050444  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5635
5636                                 ;WAIT FOR RUN & GO TO SET
5637 050452  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
     050460  004777  131046                  JSR     PC,@CLOCK       ;START THE CLOCK
5638 050464                          80$:
     050464  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5639 050472  042737  137777  001142          BIC     #^CRG,$BDDAT
5640 050500  001017                          BNE     90$
5641 050502  005737  001530                  TST     WATCH
```

```
                                        BNE     80$
                                        JSR     PC,@STOP        ;STOP THE CLOCK
                                        MOV     #RG,$GDDAT
                                        MOV     R0,$BDADR
                                        ADD     #RMMR1,$BDADR
                                        EMT     275
                                        BR      140$
                                90$:
                                        JSR     PC,@STOP        ;STOP THE CLOCK

                                ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
                                        MOV     #17.,R2
                                100$:
                                        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
                                        DEC     R2
                                        BNE     100$

                                ;DROP ON CYLINDER TO RESET LATCH
                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1

                                ;MOVE COMMAND SEQUENCER TO SEEK WAIT LOOP (31 CLOCKS)
                                        MOV     #31.,R2
                                110$:
                                        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                        DEC     R2
                                        BNE     110$

                                ;STEP THROUGH SEEK WAIT LOOP (6 CLOCKS) 2 TIMES
                                        MOV     #6.,R2
                                120$:
                                        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
                                        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
                                        DEC     R2
                                        BNE     120$

                                ;SET SEEK INCOMPLETE ERROR TO CAUSE ABORT
                                        MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1

                                ;CLOCK THE SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
                                        MOV     #2,R2
                                130$:
                                        MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)        ;LOAD RMMR1
                                        MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
                                        DEC     R2
                                        BNE     130$

                                ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
                                ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
                                        MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
                                135$:
                                        MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
                                        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
                                        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
                                        BIC     #^CEBL,$BDDAT
                                        BNE     140$
```

```
 5693
 5694 050736  005302                          DEC     R2
 5695 050740  001361                          BNE     135$                ;CONTINUE BIT CLOCKS IF COUNT NOT 0
 5696 050742  012737  020000  001140          MOV     #EBL,$GDDAT
 5697 050750  010037  001136                  MOV     R0,$BDADR
 5698 050754  062737  000024  001136          ADD     #RMMR1,$BDADR
 5699 050762  104300                          EMT     300
 5700 050764  012737  050772  001124  140$:   MOV     #150$,$LPERR                ;CHANGE LOOP ON ERROR ADDRESS
 5701
 5702                                 ;VERIFY DATA COMMAND ABORTS DURING OFFSET IF ON CYLINDER LATCH
 5703                                 ;DOESNT RESET
 5704 050772                          150$:
 5705 050772  004737  055370                  JSR     PC,SETVV            ;GO SET VOLUME VALID
      050776  000402                          BR      160$                ;BRANCH TO 160$ IF NO ERROR
      051000  104000                          EMT
 5706 051002  000536                          BR      220$
 5707
 5708                                 ;LOAD TRACK, SECTOR, AND CYLINDER ADDRESSES
 5709 051004                          160$:
 5710 051004  012760  000000  000034          MOV     #0,RMDC(R0)         ;LOAD RMDC
 5711 051012  012760  000000  000006          MOV     #0,RMDA(R0)         ;LOAD RMDA
 5712 051020  004737  055512                  JSR     PC,SETOM            ;GO SET OFFSET MODE
      051024  000402                          BR      170$                ;BRANCH TO 170$ IF NO ERROR
      051026  104000                          EMT
 5713 051030  000523                          BR      220$
 5714
 5715                                 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 5716 051032                          170$:
 5717 051032  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5718 051040  012760  000000  000014          MOV     #0,RMER1(R0)        ;LOAD RMER1
 5719 051046  012760  000000  000042          MOV     #0,RMER2(R0)        ;LOAD RMER2
 5720 051054  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)            ;LOAD RMCS1
 5721
 5722                                 ;WAIT FOR RUN AND GO TO SET
 5723 051062  012737  000310  001530          MOV     #200.,WATCH         ;SET WATCHDOG TIMER VALUE
      051070  004777  130436                  JSR     PC,@CLOCK           ;START THE CLOCK
 5724 051074                          180$:
      051074  016037  000024  001142          MOV     RMMR1(R0),$BDDAT            ;STORE RMMR1 AT $BDDAT
 5725 051102  042737  137777  001142          BIC     #^CRG,$BDDAT
 5726 051110  001017                          BNE     190$
 5727 051112  005737  001530                  TST     WATCH
 5728 051116  001366                          BNE     180$
 5729 051120  004777  130410                  JSR     PC,@STOP            ;STOP THE CLOCK
 5730 051124  012737  040000  001140          MOV     #RG,$GDDAT
 5731 051132  010037  001140                  MOV     R0,$GDDAT
 5732 051136  062737  000024  001136          ADD     #RMMR1,$BDADR
 5733 051144  104275                          EMT     275
 5734 051146  000454                          BR      220$
 5735 051150                          190$:
      051150  004777  130360                  JSR     PC,@STOP            ;STOP THE CLOCK
 5736
 5737                                 ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
 5738 051154  012702  000021                  MOV     #17.,R2
 5739 051160                          200$:
      051160  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5740 051166  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5741 051174  005302                          DEC     R2
```

```
                                        BNE     200$

                                ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
                                ;AT LOCATION 166
051200  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
051206  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1

                                ;MOVE SEQUENCER TO SET OPI AND EBL (39 CLOCKS)
051214  012702  000047                  MOV     #39.,R2
051220                          210$:
051220  012760  141401  000024            MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
051226  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
051234  005302                          DEC     R2
051236  001370                          BNE     210$

                                ;VERIFY OPI IS SET
051240  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
051246  042737  157777  001142          BIC     #^COPI,$BDDAT
051254  001011                          BNE     220$
051256  012737  020000  001140          MOV     #OPI,$GDDAT
051264  010037  001136                  MOV     R0,$BDADR
051270  062737  000014  001136          ADD     #RMER1,$BDADR
051276  104277                          EMT     277
051300  012737  051306  001124  220$:   MOV     #230$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                ;VERIFY DATA COMMAND ABORTS DURING OFFSET WAIT LOOP

051306                          230$:
051306  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
051312  000403                          BR      240$            ;BRANCH TO 240$ IF NO ERROR
051314  104000                          EMT
051316  000137  051726                  JMP     310$

                                ;LOAD SECTOR,TRACK AND CYLINDER ADDRESS
051322                          240$:
051322  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
051330  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
051336  004737  055512                  JSR     PC,SETOM        ;GO SET OFFSET MODE
051342  000402                          BR      245$            ;BRANCH TO 245$ IF NO ERROR
051344  104000                          EMT
051346  000567                          BR      310$

                                ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
051350                          245$:
051350  012760  041401  000024            MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
051356  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
051364  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
051372  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                ;WAIT FOR RUN AND GO TO SET
051400  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
051406  004777  130120                  JSR     PC,@CLOCK       ;START THE CLOCK
051412                          250$:
051412  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
051420  042737  137777  001142          BIC     #^CRG,$BDDAT
051426  001017                          BNE     260$
051430  005737  001530                  TST     WATCH
```

```
 5790 051434  001366                        BNE     250$
 5791 051436  004777  130072                JSR     PC,@STOP        ;STOP THE CLOCK
 5792 051442  012737  040000  001140        MOV     #RG,$GDDAT
 5793 051450  010037  001136                MOV     R0,$BDADR
 5794 051454  062737  000024  001136        ADD     #RMMR1,$BDADR
 5795 051462  104275                        EMT     275
 5796 051464  000520                        BR      310$
 5797 051466                        260$:
      051466  004777  130042                JSR     PC,@STOP        ;STOP THE CLOCK
 5798
 5799                               ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
 5800 051472  012702  000021                MOV     #17.,R2
 5801 051476                        270$:
      051476  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5802 051504  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5803 051512  005302                        DEC     R2
 5804 051514  001370                        BNE     270$
 5805
 5806                               ;DPOP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
 5807                               ;AT LOCATION 166
 5808 051516  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5809 051524  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5810
 5811                               ;MOVE SEQUENCER TO LOCATION  174 (37 CLOCKS)
 5812 051532  012702  000045                MOV     #37.,R2
 5813 051536                        280$:
      051536  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5814 051544  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5815 051552  005302                        DEC     R2
 5816 051554  001370                        BNE     280$
 5817
 5818                               ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER RESET
 5819 051556  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5820
 5821                               ;STEP SEQUENCER THROUGH OFFSET WAIT LOOP TWICE (7 CLOCKS)
 5822 051564  012702  000007                MOV     #7.,R2
 5823 051570                        290$:
      051570  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 5824 051576  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5825 051604  005302                        DEC     R2
 5826 051606  001370                        BNE     290$
 5827
 5828                               ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
 5829 051610  012760  041101  000024        MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
 5830
 5831                               ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
 5832 051616  012702  000002                MOV     #2,R2
 5833 051622                        300$:
      051622  012760  141101  000024        MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5834 051630  012760  041101  000024        MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
 5835 051636  005302                        DEC     R2
 5836 051640  001370                        BNE     300$
 5837
 5838
 5839                               ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
                                    ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
 5840 051642  012702  000020                MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
 5841 051646                        305$:
```

```
                051646  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF.MCLK,RMMR1(R0)    ;LOAD RMMR1
     5842       051654  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
     5843       051662  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
     5844       051670  042737  157777  001142          BIC     #^CEBL,$BDDAT
     5845       051676  001013                          BNE     310$
     5846
     5847       051700  005302                          DEC     R2
     5848       051702  001361                          BNE     305$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
     5849       051704  012737  020000  001140          MOV     #EBL,$GDDAT
     5850       051712  010037  001136                  MOV     R0,$BDADR
     5851       051716  062737  000024  001136          ADD     #RMMR1,$BDADR
     5852       051724  104271                          EMT     271
     5853
     5854       051726  012737  051734  001124  310$:   MOV     #320$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
     5855
     5856                                       ;VERIFY THAT DATA COMMAND ABORTS DURING SECTOR WAIT LOOP AT LOCATION 179
     5857
     5858       051734                          320$:
                051734  004737  055370                  JSR     PC,SETVV        ;GO SET VOLUME VALID
                051740  000403                          BR      330$            ;BRANCH TO 330$ IF NO ERROR
                051742  104000                          EMT
     5859       051744  000137  052372                  JMP     420$
     5860
     5861                                       ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
     5862       051750                          330$:
     5863       051750  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
     5864       051756  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
     5865       051764  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
     5866       051772  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
     5867       052000  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
     5868       052006  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
     5869
     5870                                       ;WAIT FOR RUN AND GO TO SET
     5871       052014  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
                052022  004777  127504                  JSR     PC,@CLOCK       ;START THE CLOCK
     5872       052026                          340$:
                052026  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
     5873       052034  042737  137777  001142          BIC     #^CRG,$BDDAT
     5874       052042  001017                          BNE     350$
     5875       052044  005737  001530                  TST     WATCH
     5876       052050  001366                          BNE     340$
     5877       052052  004777  127456                  JSR     PC,@STOP        ;STOP THE CLOCK
     5878       052056  012737  040000  001140          MOV     #RG,$GDDAT
     5879       052064  010037  001136                  MOV     R0,$BDADR
     5880       052070  062737  000024  001136          ADD     #RMMR1,$BDADR
     5881       052076  104275                          EMT     275
     5882       052100  000534                          BR      420$
     5883       052102                          350$:
                052102  004777  127426                  JSR     PC,@STOP        ;STOP THE CLOCK
     5884
     5885                                       ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
     5886       052106  012702  000021                  MOV     #17.,R2
     5887       052112                          360$:
                052112  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
     5888       052120  012760  041401  000024          MOV     #DMD!MUR.MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
     5889       052126  005302                          DEC     R2
     5890       052130  001370                          BNE     360$
```

```
                                        ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
                                        ;AT LOCATION 166
052132  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1

                                        ;MOVE SEQUENCER TO SECTOR WAIT (34 CLOCKS)
052140  012702  000042                        MOV     #34.,R2
052144                                  370$:
052144  012760  141401  000024                MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
052152  012760  041401  000024                MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
052160  005302                                DEC     R2
052162  001370                                BNE     370$

                                        ;STEP THROUGH SECTOR WAIT LOOP TWICE AND VERIFY SEARCH IS ENABLED
                                        ;DURING THE LOOP (6 CLOCKS)
052164  012702  000006                        MOV     #6.,R2
052170                                  380$:
052170  012760  141401  000024                MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
052176  012760  041401  000024                MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
052204  016037  000024  001142                MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
052212  042737  173777  001142                BIC     #^CESRC,$BDDAT
052220  001403                                BEQ     390$
052222  005302                                DEC     R2
052224  001361                                BNE     380$
052226  000412                                BR      400$
052230  012737  004000  001140  390$:         MOV     #ESRC,$GDDAT
052236  010037  001136                        MOV     R0,$BDADR
052242  062737  000024  001136                ADD     #RMMR1,$BDADR
052250  104301                                EMT     301
052252  000447                                BR      420$

                                        ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
052254                                  400$:
052254  012760  041501  000024                MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1

                                        ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
052262  012702  000002                        MOV     #2,R2
052266                                  410$:
052266  012760  141501  000024                MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)     ;LOAD RMMR1
052274  012760  041501  000024                MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
052302  005302                                DEC     R2
052304  001370                                BNE     410$

                                        ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
                                        ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
052306  012702  000020                        MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
052312                                  415$:
052312  012760  045501  000024                MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)     ;LOAD RMMR1
052320  012760  041501  000024                MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1

                                        ;VERIFY EBL IS SET
052326  016037  000024  001142                MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
052334  042737  157777  001142                BIC     #^CEBL,$BDDAT
052342  001013                                BNE     420$

052344  005302                                DEC     R2
052346  001361                                BNE     415$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
```

```
 5943 052350  012737  020000  001140          MOV     #EBL,$GDDAT
 5944 052356  010037  001136                  MOV     R0,$BDADR
 5945 052362  062737  000024  001136          ADD     #RMMR1,$BDADR
 5946 052370  104271                          EMT     271
 5947
 5948 052372                          420$:                          ;END OF TEST
 5949
 5950                                 ;;*****************************************************************
                                      ;*TEST 120       DATA COMMAND TESTS (3)
                                      ;;*****************************************************************
      052372                          TST120:
      052372  000004                          SCOPE                  ;SCOPE CALL
      052374  000240                          NOP
      052376  012706  001100                  MOV     #STACK,SP      ;LOAD THE STACK POINTER
      052402  013700  001276                  MOV     $BASE,R0       ;R0 = UNIBUS ADDRESS
      052406  013701  001462                  MOV     TSTQUE,R1      ;R1 = POINTER TO DEVICE
      052412  012737  000120  001226          MOV     #120,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
 5951
 5952                                 ;VERIFY THE TAG BUS DURING DATA COMMAND
 5953                                 ;FIRST PART USES OFFSET FORWARD
 5954
 5955                                 ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
 5956 052420  012737  001466  001442          MOV     #822.,RMDC0    ;LAST CYLINDER
 5957 052426  013737  001330  001414          MOV     LSTRK,RMDA0    ;SETUP LAST TRACK AND
 5958 052434  112737  000035  001414          MOVB    #29.,RMDA0     ;LAST SECTOR
 5959 052442                          5$:
 5960 052442  012737  000200  001440          MOV     #0FD,RMOF0     ;FORWARD OFFSET
 5961 052450  012737  000071  001406          MOV     #RD!GO,RMCS10  ;READ DATA
 5962 052456  012737  177400  001410          MOV     #-256.,RMWC0   ;WORD COUNT
 5963 052464  012737  103712  001412          MOV     #BUFFER,RMBA0  ;BUFFER ADDRESS
 5964
 5965                                 ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
 5966 052472  004737  052514                  JSR     PC,10$
 5967
 5968                                 ;SECOND PART USES OFFSET REVERSE
 5969                                 ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
 5970 052476  112737  000000  001440          MOVB    #0,RMOF0       ;REVERSE OFFSET
 5971
 5972                                 ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
 5973 052504  004737  052514                  JSR     PC,10$
 5974 052510  000137  053600                  JMP     300$
 5975
 5976                                 ;;*****************************************************************
 5977                                 ;SUBROUTINE USED DURING TEST
 5978                                 ;;*****************************************************************
 5979
 5980 052514                          10$:
 5981 052514  004737  055370                  JSR     PC,SETVV       ;GO SET VOLUME VALID
      052520  000403                          BR      20$            ;BRANCH TO 20$ IF NO ERROR
      052522  104000                          EMT
 5982 052524  000137  053416                  JMP     160$
 5983
 5984                                 ;LOAD TRACK, SECTOR AND CYLINDER ADDRESS, LOAD OFFSET
 5985 052530                          20$:
 5986 052530  013760  001414  000006          MOV     RMDA0,RMDA(R0) ;LOAD RMDA
 5987 052536  013760  001442  000034          MOV     RMDC0,RMDC(R0) ;LOAD RMDC
```

```
 5988 052544  013760  001440  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
 5989 052552  004737  055512                  JSR     PC,SETOM        ;GO SET OFFSET MODE
      052556  000403                          BR      30$             ;BRANCH TO 30$ IF NO ERROR
      052560  104000                          EMT
 5990 052562  000137  053416                  JMP     160$
 5991
 5992                                 ;LOAD BUFFER ADDRESS AND WORD COUNT
 5993 052566                          30$:
 5994 052566  013760  001410  000002          MOV     RMWC0,RMWC(R0)  ;LOAD RMWC
 5995 052574  013760  001412  000004          MOV     RMBA0,RMBA(R0)  ;LOAD RMBA
 5996
 5997                                 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 5998 052602  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5999 052610  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 6000 052616  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 6001 052624  013760  001406  000000          MOV     RMCS10,RMCS1(R0)        ;LOAD RMCS1
 6002
 6003                                 ;WAIT FOR RUN AND GO TO SET
 6004 052632  012737  000310  001530          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
      052640  004777  126666                  JSR     PC,@CLOCK       ;START THE CLOCK
 6005 052644                          40$:
      052644  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 6006 052652  042737  137777  001142          BIC     #^CRG,$BDDAT
 6007 052660  001020                          BNE     50$
 6008 052662  005737  001530                  TST     WATCH
 6009 052666  001366                          BNE     40$
 6010 052670  004777  126640                  JSR     PC,@STOP        ;STOP THE CLOCK
 6011 052674  012737  040000  001140          MOV     #RG,$GDDAT
 6012 052702  010037  001136                  MOV     R0,$BDADR
 6013 052706  062737  000024  001136          ADD     #RMMR1,$BDADR
 6014 052714  104275                          EMT     275
 6015 052716  000137  053416                  JMP     160$
 6016 052722                          50$:
      052722  004777  126606                  JSR     PC,@STOP        ;STOP THE CLOCK
 6017 052726  012704  053420                  MOV     #200$,R4        ;R4 = TABLE POINTER
 6018
 6019                                 ;STEP SEQUENCER TO HEAD SEQUENCE AT LOCATION 156 (17 CLOCKS)
 6020 052732  012705  000021                  MOV     #17.,R5         ;R5 = CLOCK COUNT
 6021 052736                          60$:
      052736  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 6022
 6023                                 ;;*****************************************************************
 6024 052744  013737  001142  001174          MOV     $BDDAT,$TMP0    ;IF CC AND CH ARE SET AT THE SAME TIME
 6025                                                                 ;THE ECO TO THE CS BOARD IS IMPLEMENTED.
 6026                                   .
 6027 052752  042737  171777  001174          BIC     #^C<CC!CH>,$TMP0        ;SAVE CC AND CH BITS
 6028 052760  022737  006000  001174          CMP     #CC!CH,$TMP0    ;ARE CC AND CH SET ?
 6029 052766  001414                          BEQ     65$             ;YES, BRANCH TO NEW LOCATION
 6030                                 ;;*****************************************************************
 6031
 6032 052770  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
 6033 052776  012437  001140                  MOV     (R4)+,$GDDAT
 6034 053002  053737  001442  001140          BIS     RMDC0,$GDDAT    ;OR CYLINDER ADDRESS
 6035 053010  023737  001140  001142          CMP     $GDDAT,$BDDAT
 6036 053016  001011                          BNE     70$             ;BRANCH IF TAG BUS WRONG
 6037 053020                          65$:
 6038 053020  012760  141401  000024          MOV     #DMD!MUR!MOC.DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
```

```
 6039 053026  012760  041401  000024          MOV     #DMD!MUR.MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
 6040 053034  005305                          DEC     R5
 6041 053036  001337                          BNE     60$
 6042 053040  000407                          BR      80$
 6043 053042  010037  001136          70$:    MOV     R0,$BDADR
 6044 053046  062737  000040  001136          ADD     #RMMR2,$BDADR
 6045 053054  104276                          EMT     276
 6046 053056  000557                          BR      160$
 6047
 6048                                 ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST AT
 6049                                 ;LOCATION 166
 6050 053060                          80$:
 6051 053060  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 6052 053066  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 6053
 6054                                 ;STEP SEQUENCER TO END OF OFFSET AT LOCATION 174 (37 CLOCKS)
 6055 053074  012705  000045          MOV     #37.,R5         ;RELOAD CLOCK COUNT
 6056 053100                          90$:
      053100  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 6057
 6058                                 ;;*****************************************************************
 6059 053106  013737  001142  001174          MOV     $BDDAT,$TMP0    ;IF CC AND CH ARE SET AT SAME TIME
 6060                                                                 ;THE ECO TO THE CS BOARD IS IMPLEMENTED.
 6061
 6062 053114  042737  171777  001174          BIC     #^C<CC!CH>,$TMP0        ;SAVE CC AND CH BITS
 6063 053122  022737  006000  001174          CMP     #CC!CH,$TMP0    ;ARE CC AND CH SET ?
 6064 053130  001003                          BNE     92$             ;NO !!
 6065 053132  162704  000002                  SUB     #2,R4           ;ADJUST THE TABLE ADDRESS
 6066 053136  000441                          BR      115$            ;TO OTHER LOCATION
 6067 053140                          92$:
 6068                                 ;;*****************************************************************
 6069
 6070 053140  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
 6071 053146  011437  001140                  MOV     (R4),$GDDAT
 6072 053152  032714  002000                  BIT     #CH,(R4)
 6073 053156  001425                          BEQ     110$            ;BRANCH IF CONTROL/HEADER NOT ON
 6074 053160  032714  004000                  BIT     #CC,(R4)
 6075 053164  001416                          BEQ     100$            ;BRANCH IF HEADER TAG
 6076
 6077                                 ;CONTROL TAG SHOULD BE ON-SETUP EXPECTED OFFSET
 6078 053166  052737  000010  001140          BIS     #BB03,$GDDAT    ;ASSUME OFD IS NOT SET
 6079 053174  032737  000200  001440          BIT     #OFD,RMOFO
 6080 053202  001406                          BEQ     95$
 6081 053204  042737  000010  001140          BIC     #BB03,$GDDAT    ;RESET BUS BIT 3 - DIRECTION IS REV
 6082 053212  052737  000004  001140          BIS     #BB02,$GDDAT
 6083 053220  000404                  95$:    BR      110$
 6084
 6085                                 ;HEADER TAG SHOULD BE ON - SETUP EXPECTED TRACK ADDRESS
 6086 053222                          100$:
 6087 053222  113703  001415                  MOVB    RMDA0+1,R3      ;GET TRACK
 6088 053226  050337  001140                  BIS     R3,$GDDAT
 6089
 6090                                 ;COMPARE EXPECTED AND RECEIVED TAG BUS DATA
 6091 053232                          110$:
 6092 053232  023737  001140  001142          CMP     $GDDAT,$BDDAT
 6093 053240  001013                          BNE     120$
 6094 053242                          115$:
```

```
        053242  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 6095   053250  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 6096   053256  062704  000002                  ADD     #2,R4                   ;MOVE TABLE POINTER
 6097   053262  005305                          DEC     R5                      ;DECREMENT CLOCK COUNT
 6098   053264  001305                          BNE     90$
 6099   053266  000407                          BR      130$
 6100   053270  010037  001136          120$:   MOV     R0,$BDADR
 6101   053274  062737  000040  001136          ADD     #RMMR2,$BDADR
 6102   053302  104276                          EMT     276
 6103   053304  000444                          BR      160$
 6104
 6105                                  ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER TO PASS TEST AT
 6106                                  ;SEQUENCER LOCATION 175
 6107   053306                         130$:
 6108   053306  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 6109   053314  012760  041401  000024          MOV     #DMD!MUR.MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 6110
 6111                                  ;STEP SEQUENCER TO SECTOR WAIT LOOP (8 CLOCKS)
 6112   053322  012705  000010                  MOV     #8.,R5
 6113   053326                         140$:
        053326  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 6114
 6115                                  ;;**********************************************************************
 6116                                  ;       BIC     #RQA!RQB!TST,$BDDAT
 6117                                                  ;THE FOLLOWING CODE WAS ADDED
 6118                                                  ;TO ALLOW THE PROGRAM TO RUN WITH
 6119                                                  ;OR WITHOUT ECO TO THE CS BOARD.
 6120
 6121   053334  042737  171777  001142          BIC     #^C<CC!CH>,$BDDAT       ;SAVE CC AND CH BITS
 6122   053342  012737  006000  001140          MOV     #CC!CH,$GDDAT   ;GET EXPECTED DATA
 6123                                  ;;**********************************************************************
 6124
 6125   053350  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;GOOD DATA SAME AS LAST CMP
 6126   053356  001011                          BNE     150$            ;BRANCH IF ERROR
 6127   053360  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 6128   053366  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 6129   053374  005305                          DEC     R5
 6130   053376  001353                          BNE     140$
 6131   053400  000406                          BR      160$
 6132   053402  010037  001136          150$:   MOV     R0,$BDADR
 6133   053406  062737  000040  001136          ADD     #RMMR2,$BDADR
 6134   053414  104276                          EMT     276
 6135
 6136   053416  000207                  160$:   RTS     PC
 6137
 6138                                  ;TABLE OF TAG BUS CONTROL AND DATA VALUES
 6139   053420                         200$:
 6140   053420  001777                          .WORD   1777            ;LOCATION 0
 6141   053422  001777                          .WORD   1777            ;LOCATION 25
 6142   053424  001777                          .WORD   1777            ;LOCATION 26
 6143   053426  001777                          .WORD   1777            ;LOCATION 128
 6144   053430  001777                          .WORD   1777            ;LOCATION 129
 6145   053432  001777                          .WORD   1777            ;LOCATION 130
 6146   053434  004000                          .WORD   CC              ;LOCATION 144
 6147   053436  004000                          .WORD   CC              ;LOCATION 145
 6148   053440  024000                          .WORD   CC!TAG          ;LOCATION 146
 6149   053442  024000                          .WORD   CC!TAG          ;LOCATION 147
```

```
                                                                J  1
 6150 053444  024000                    .WORD   CC!TAG          ;LOCATION 148
 6151 053446  024000                    .WORD   CC!TAG          ;LOCATION 149
 6152 053450  024000                    .WORD   CC!TAG          ;LOCATION 150
 6153 053452  024000                    .WORD   CC!TAG          ;LOCATION 151
 6154 053454  004000                    .WORD   CC              ;LOCATION 152
 6155 053456  004000                    .WORD   CC              ;LOCATION 153
 6156 053460  001777                    .WORD   1777            ;LOCATION 154
 6157 053462  002000                    .WORD   CH              ;LOCATION 156
 6158 053464  002000                    .WORD   CH              ;LOCATION 157
 6159 053466  022000                    .WORD   CH!TAG          ;LOCATION 158
 6160 053470  022000                    .WORD   CH!TAG          ;LOCATION 159
 6161 053472  022000                    .WORD   CH!TAG          ;LOCATION 160
 6162 053474  022000                    .WORD   CH!TAG          ;LOCATION 161
 6163 053476  022000                    .WORD   CH!TAG          ;LOCATION 162
 6164 053500  022000                    .WORD   CH!TAG          ;LOCATION 163
 6165 053502  002000                    .WORD   CH              ;LOCATION 164
 6166 053504  002000                    .WORD   CH              ;LOCATION 165
 6167 053506  001777                    .WORD   1777            ;LOCATION 232
 6168 053510  001777                    .WORD   1777            ;LOCATION 233
 6169 053512  001777                    .WORD   1777            ;LOCATION 234
 6170 053514  001777                    .WORD   1777            ;LOCATION 235
 6171 053516  001777                    .WORD   1777            ;LOCATION 236
 6172 053520  001777                    .WORD   1777            ;LOCATION 237
 6173 053522  001777                    .WORD   1777            ;LOCATION 238
 6174 053524  001777                    .WORD   1777            ;LOCATION 239
 6175 053526  001777                    .WORD   1777            ;LOCATION 240
 6176 053530  001777                    .WORD   1777            ;LOCATION 241
 6177 053532  001777                    .WORD   1777            ;LOCATION 242
 6178 053534  001777                    .WORD   1777            ;LOCATION 243
 6179 053536  001777                    .WORD   1777            ;LOCATION 244
 6180 053540  001777                    .WORD   1777            ;LOCATION 245
 6181 053542  001777                    .WORD   1777            ;LOCATION 246
 6182 053544  001777                    .WORD   1777            ;LOCATION 247
 6183 053546  001777                    .WORD   1777            ;LOCATION 248
 6184 053550  001777                    .WORD   1777            ;LOCATION 249
 6185 053552  001777                    .WORD   1777            ;LOCATION 250
 6186 053554  001777                    .WORD   1777            ;LOCATION 251
 6187 053556  001777                    .WORD   1777            ;LOCATION 252
 6188 053560  001777                    .WORD   1777            ;LOCATION 166
 6189 053562  006000                    .WORD   CC!CH           ;LOCATION 169
 6190 053564  006000                    .WORD   CC!CH           ;LOCATION 170
 6191 053566  026000                    .WORD   CC!CH!TAG       ;LOCATION 171
 6192 053570  026000                    .WORD   CC!CH!TAG       ;LOCATION 172
 6193 053572  026000                    .WORD   CC!CH!TAG       ;LOCATION 173
 6194 053574  026000                    .WORD   CC!CH!TAG       ;LOCATION 174
 6195 053576  026000                    .WORD   CC!CH!TAG       ;LOCATION 175,ETC
 6196
 6197 053600                    300$:                           ;END OF TEST
 6198
```

```
                                .SBTTL  END OF SUB-PASS ROUTINE
                                ;THIS IS THE END OF SUB-PASS ROUTINE. THIS ROUTINE IS USED TO
                                ;TERMINATE THE OPERATION OF THE CURRENT DEVICE UNDER TEST AND
                                ;SELECT THE NEXT DEVICE FOR TEST. IF THERE ARE NO MORE DEVICES
                                ;TO TEST, EXIT IS MADE TO '$EOP' ROUTINE. OTHERWISE, RETURN
                                ;IS MADE TO 'READY' ROUTINE.

053600  000004                  $EOSP:  SCOPE
053602  000240                          NOP
053604  013700  001462                  MOV     TSTQUE,R0       ;GET POINTER TO TSTQUE
053610  062700  000002                  ADD     #2,R0           ;ADJUST POINTER TO NEXT DEVICE
053614  010037  001462                  MOV     R0,TSTQUE       ;SAVE POINTER TO TSTQUE
053620  005710                          TST     (R0)            ;ANY MORE DEVICES FOR TEST ?
053622  001402                          BEQ     1$              ;BR IF NO
053624  000137  007152                  JMP     READY           ;YES, JUMP TO READY
053630  012737  001464  001462  1$:     MOV     #TSTQUE+2,TSTQUE        ;INITIALIZE POINTER TO FIRST DEVICE IN
                                                                ;TEST QUE TABLE

                                .SBTTL  END OF PASS ROUTINE

                                ;;**************************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY''
                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO READY

053636                          $EOP:
053636  000240                          NOP
053640  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
053644  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
053650  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
053654  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
053662  005327                          DEC     (PC)+           ;;LOOP?
053664  000001                  $EOPCT: .WORD   1
053666  003063                          BGT     $DOAGN          ;;YES
053670  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
053672  000001                  $ENDCT: .WORD   1
053674  053664                          $EOPCT
053676  104401  053704                  TYPE    ,65$            ;;TYPE ASCIZ STRING
053702  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
053722                          64$:
053722  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE PASS NUMBER
053726  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
053730  104401  053736                  TYPE    ,67$            ;;TYPE ASCIZ STRING
053734  000421                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
054000                          66$:
054000  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                ;;TOTAL NUMBER OF ERRORS
054004  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
054006  104401  001217                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
054012  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
054016  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
```

```
054022  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
054024  000005                          RESET                   ;;CLEAR THE WORLD
054026  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
054030  000240                          NOP                     ;;SAVE ROOM
054032  000240                          NOP                     ;;FOR
054034  000240                          NOP                     ;;ACT11
054036                          $DOAGN:
054036  000137                          JMP     @(PC)+          ;;RETURN
054040  007152                  $RTNAD: .WORD   READY
054042     377      377      000 $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
                                        .EVEN
    21
```

```
                                        .SBTTL  SUBROUTINES
                                        ;;******************************************************************
                                        .SBTTL  ERROR TYPEOUT ROUTINE

                                        ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
                                        ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
                                        ;*
                                        ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
                                        ;*PRINTED ON THE FIRST LINE;
                                        ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
                                        ;*ONE OR MORE SUCCEEDING LINES;
                                        ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
                                        ;*ARE PRINTED AFTER THE ERROR MESSAGE.

054046                                  ERRTYP:
054046  104414                                SAVREG
054050  032777  020000  125076                BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
054056  001402                                BEQ     1$              ;NO!!
054060  000137  054662                        JMP     25$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
054064  104401  001217                1$:     TYPE    ,$CRLF
054070  104401  054676                        TYPE    ,ERTY00         ;TYPE 'UNIT#''
054074  013746  001234                        MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
054100  104403                                TYPOS                   ;;GO TYPE--OCTAL ASCII
054102     003                              .BYTE   3               ;;TYPE 3 DIGIT(S)
054103     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        ;TYPE 'DRIVE TYPE' RM05, RM03 OR RM02 FOR UNIT UNDER TEST
054104  016000  000026                        MOV     RMDT(R0),R0     ;STORE RMDT AT R0
054110  042700  177740                        BIC     #177740,R0      ;SAVE DRIVE TYPE BITS AND
054114  012737  054745  054166                MOV     #$RM03,3$       ;GET ASCII DRIVE TYPE
054122  022700  000024                        CMP     #24,R0          ;IS DEVICE AN RM03 ?
054126  001414                                BEQ     2$              ;YES !!

054130  012737  054740  054166                MOV     #$RM02,3$       ;SAVE ASCII DRIVE TYPE
054136  022700  000025                        CMP     #25,R0          ;IS DEVICE AN RM02 ?
054142  001406                                BEQ     2$              ;YES !!

054144  012737  054752  054166                MOV     #$RM05,3$       ;SAVE ASCII DRIVE TYPE
054152  022700  000027                        CMP     #27,R0          ;IS DEVICE AN RM05 ?
054156  001004                                BNE     4$              ;NO !!
054160  104401  054734                2$:     TYPE    ,ERTY05         ;TYPE '' - ''
054164  104401                                TYPE                    ;TYPE DRIVE TYPE
054166  000000                        3$:     .WORD   0               ;DRIVE TYPE MESAAGE IS STORED HERE

                                        ;TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
054170  005037  054666                4$:     CLR     TSTNMB          ;LOAD TEST NUMBER FOR
054174  013737  001226  054666                MOV     $TESTN,TSTNMB
054202  104401  054704                        TYPE    ,ERTY01         ;TYPE 'TST#''
054206  013746  054666                        MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
054212  104403                                TYPOS                   ;;GO TYPE--OCTAL ASCII
054214     003                              .BYTE   3               ;;TYPE 3 DIGIT(S)
054215     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
054216  005037  054670                        CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
054222  113737  001130  054670                MOVB    $ITEMB,ERRNMB   ;TYPEOUT
054230  001406                                BEQ     5$              ;SKIP IF NO ERROR CALLED
```

```
     50 054232  104401  054714                 TYPE    ,ERTY02          ;TYPE 'ERR#'
     51 054236  013746  054670                 MOV     ERRNMB,-(SP)     ;;SAVE ERRNMB FOR TYPEOUT
                                                                         ;;TYPE ERROR NUMBER
        054242  104403                         TYPOS                    ;;GO TYPE--OCTAL ASCII
        054244     003                         .BYTE   3                ;;TYPE 3 DIGIT(S)
        054245     000                         .BYTE   0                ;;SUPPRESS LEADING ZEROS
     52 054246  104401  054723           5$:   TYPE    ,ERTY03          ;TYPE 'PC='
     53 054252  013746  001132                 MOV     $ERRPC,-(SP)     ;;SAVE $ERRPC FOR TYPEOUT
                                                                         ;;TYPE PROGRAM COUNTER
        054256  104403                         TYPOS                    ;;GO TYPE--OCTAL ASCII
        054260     006                         .BYTE   6                ;;TYPE 6 DIGIT(S)
        054261     001                         .BYTE   1                ;;TYPE LEADING ZEROS
     54                                 ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
     55 054262  005737  054670           6$:   TST     ERRNMB           ;WAS AN ERROR CALLED?
     56 054266  001575                         BEQ     25$              ;NO!!
     57 054270  104401  001217                 TYPE    ,$CRLF           ;YES-TYPE CRLF
     58 054274  105037  054674                 CLRB    BOTFLG           ;CLEAR BOT FLAG
     59 054300  105037  054675                 CLRB    CHRCNT           ;CLEAR CHARACTER COUNTER
     60 054304  013700  054670                 MOV     ERRNMB,R0        ;R0 POINTS TO FIRST OF
     61 054310  006300                         ASL     R0               ;FOUR ENTRIES IN ERROR
     62 054312  006300                         ASL     R0               ;TABLE
     63 054314  006300                         ASL     R0
     64 054316  062700  001526                 ADD     #$ERRTB-8.,R0
     65 054322  011001                         MOV     (R0),R1          ;R1 POINTS TO ERROR MESSAGE
     66                                                                  ;TABLE
     67 054324  001507                         BEQ     16$              ;BRANCH IF NO ERROR MESSAGE
     68                                 ;TYPE THE ERROR MESSAGE
     69 054326  012102                   7$:   MOV     (R1)+,R2         ;R2=ADDRESS OF MESSAGE STRING
     70 054330  001505                         BEQ     16$              ;BRANCH IF END OF MESSAGE
     71 054332  010237  054500                 MOV     R2,15$           ;LOAD ADDRESS OF STRING
     72 054336  005037  054672                 CLR     BOTADR           ;CLEAR BOT ADDRESS
     73 054342  112203                   8$:   MOVB    (R2)+,R3         ;END OF STRING??
     74 054344  001454                         BEQ     14$              ;YES!!
     75 054346  122703  000015                 CMPB    #CR,R3           ;CARRIAGE RETURN??
     76 054352  001003                         BNE     9$               ;NO!!
     77 054354  105037  054675                 CLRB    CHRCNT           ;YES-CLEAR CHAR COUNT
     78 054360  000770                         BR      8$               ;GET NEXT CHARACTER
     79 054362  122703  000012           9$:   CMPB    #LF,R3           ;LINE FEED??
     80 054366  001765                         BEQ     8$               ;YES-GET NEXT CHARACTER
     81 054370  122703  000011                 CMPB    #HT,R3           ;HORIZONTAL TAB??
     82 054374  001007                         BNE     11$              ;NO!!
     83 054376  105237  054675           10$:  INCB    CHRCNT           ;ADJUST CHARACTER COUNT
     84 054402  132737  000007  054675         BITB    #7,CHRCNT
     85 054410  001372                         BNE     10$
     86 054412  000407                         BR      12$
     87 054414  105237  054675           11$:  INCB    CHRCNT           ;INCREMENT CHARACTER COUNT
     88 054420  122703  000040                 CMPB    #' ,R3           ;SPACE??
     89 054424  001002                         BNE     12$              ;NO!!
     90 054426  010237  054672                 MOV     R2,BOTADR        ;SAVE ADDRESS OF SPACE
     91 054432  122737  000100  054675   12$:  CMPB    #64.,CHRCNT      ;END OF LINE??
     92 054440  103340                         BHIS    8$               ;NO!!
     93 054442  013704  054672                 MOV     BOTADR,R4        ;GET ADDRESS OF LAST SPACE
     94 054446  001007                         BNE     13$              ;BRANCH IF SPACE DETECTED
     95 054450  104401  001217                 TYPE    ,$CRLF           ;TYPE CRLF
     96 054454  105037  054675                 CLRB    CHRCNT           ;CLEAR CHARACTER COUNT
     97 054460  013702  054500                 MOV     15$,R2           ;SET UP R2 FOR TESTING
     98 054464  000726                         BR      8$
```

```
 99 054466  105044                 13$:    CLRB    -(R4)           ;REPLACE SPACE
100 054470  112737  177777  054674          MOVB    #-1,BOTFLG      ;SET BOT FLAG
101 054476  104401              14$:    TYPE                    ;TYPE ERROR MESSAGE STRING
102 054500  000000              15$:    .WORD                   ;STRING ADDRESS GOES HERE
103 054502  105737  054674              TSTB    BOTFLG          ;WAS STRING TRUNCATED??
104 054506  001707                      BEQ     7$              ;NO!!
105 054510  104401  001217              TYPE    ,$CRLF          ;YES-TYPE CRLF
106 054514  105037  054674              CLRB    BOTFLG          ;CLEAR BOT FLAG
107 054520  105037  054675              CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
108 054524  013702  054672              MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
109 054530  010237  054500              MOV     R2,15$          ;SETUP 15$ FOR TYPING
110 054534  112742  000040              MOVB    #' ,-(R2)       ;RESTORE SPACE
111 054540  105722                      TSTB    (R2)+           ;RESTORE R2
112 054542  000677                      BR      8$              ;TYPE REST OF STRING
113
114                              ;TYPE ERROR HEADER AND ERROR DATA
115 054544                       16$:
116 054544  016001  000002      17$:    MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
117 054550  001444                      BEQ     25$             ;BRANCH IF NO HEADER
118 054552  104401  001217              TYPE    ,$CRLF          ;(ASSUME NO DATA)
119 054556  016002  000004              MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
120 054562  016003  000006              MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
121 054566  012137  054576      18$:    MOV     (R1)+,19$       ;PUT HEADER ADDRESS FOR TYPE
122 054572  001433                      BEQ     25$             ;BRANCH IF END OF HEADERS
123                                                             ;(ASSUME END OF DATA)
124 054574  104401                      TYPE
125 054576  000000              19$:    .WORD   0               ;HEADER ADDRESS GOES HERE
126 054600  104401  001217              TYPE    ,$CRLF
127 054604  005702                      TST     R2              ;DATA WITH HEADER??
128 054606  001767                      BEQ     18$             ;NO!!
129 054610  012204                      MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
130 054612  012305                      MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
131 054614  105725              20$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
132 054616  100407                      BMI     22$             ;BINARY
133 054620  001403                      BEQ     21$             ;OCTAL
134 054622  013446                      MOV     @(R4)+,-(SP)    ;DECIMAL
135 054624  104405                      TYPDS
136 054626  000405                      BR      23$
137 054630  013446              21$:    MOV     @(R4)+,-(SP)
138 054632  104402                      TYPOC
139 054634  000402                      BR      23$
140 054636  013446              22$:    MOV     @(R4)+,-(SP)
141 054640  104406                      TYPBN
142 054642  005714              23$:    TST     (R4)            ;MORE DATA??
143 054644  001403                      BEQ     24$             ;NO!!
144 054646  104401  054731              TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
145 054652  000760                      BR      20$             ;AND CONTINUE
146 054654  104401  001217      24$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
147 054660  000742                      BR      18$             ;BEFORE NEXT HEADER
148 054662  104415              25$:    RESREG
149 054664  000207                      RTS     PC
150
151 054666  000000              TSTNMB: .WORD   0               ;TEST NUMBER
152 054670  000000              ERRNMB: .WORD   0               ;ERROR NUMBER
153 054672  000000              BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
154 054674      000              BOTFLG: .BYTE   0               ;BOT FLAG
155 054675      000              CHRCNT: .BYTE   0               ;CHARACTER COUNT
```

```
    156
    157 054676      125     116     111  ERTY00: .ASCIZ  @UNIT#@
    158 054704      054     040     124  ERTY01: .ASCIZ  @, TEST#@
    159 054714      054     040     105  ERTY02: .ASCIZ  @, ERR#@
    160 054723      054     040     120  ERTY03: .ASCIZ  @, PC=@
    161 054731      040     040     000  ERTY04: .ASCIZ  @ @
    162 054734      040     055     040  ERTY05: .ASCIZ  @ - @
    163
    164 054740      122     115     060  $RM02:  .ASCIZ  @RM02@
    165 054745      122     115     060  $RM03:  .ASCIZ  @RM03@
    166 054752      122     115     060  $RM05:  .ASCIZ  @RM05@
    167                                  .EVEN
    168
```

```
                                        .SBTTL  CLOCK SUBROUTINES
                                        ;ROUTINE TO SIZE FOR CLOCKS (KW11-L OR KW11-P)
 054760  000240                 SIZCLK: NOP
 054762  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 054766  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
 054772  012737  055056  000004         MOV     #10$,ERRVEC     ;LOAD 04 TRAP VECTORS
 055000  012737  000300  000006         MOV     #PR6,ERRVEC+2
                                        ;SEE IF A KW11-P CLOCK IS PRESENT - GO TO 10$ IF NOT PRESENT
 055006  005777  124474                 TST     @$LPCSR         ;TEST FOR P CLOCK
 055012  012737  055220  001532         MOV     #PCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
 055020  012737  055342  001534         MOV     #PSTOP,STOP     ;LOAD STOP ADDRESS
 055026  012777  055306  124456         MOV     #PCOUNT,@$LPVEC ;LOAD P CLOCK INTERRUPT VECTOR
 055034  012777  000300  124452         MOV     #PR6,@$LPVEC+2
 055042  013777  001522  124450         MOV     $LLVEC+2,@$LLVEC        ;CLEAR L CLOCK INTERRUPT VECTOR
 055050  005077  124446                 CLR     @$LLVEC+2
 055054  000454                         BR      30$
 055056  012716  055064         10$:    MOV     #15$,(SP)       ;DUMMY RTI ADDRESS
 055062  000002                         RTI                     ;RESTORE PRIORITY
                                        ;NO P CLOCK-SEE IF L CLOCK IS PRESENT-GO TO 20$ IF NOT PRESET
 055064                         15$:
 055064  012737  055142  000004         MOV     #20$,ERRVEC     ;CHANGE 04 TRAP VECTOR
 055072  005777  124420                 TST     @$LLCSR         ;TEST FOR L CLOCK
 055076  012737  055236  001532         MOV     #LCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
 055104  012737  055350  001534         MOV     #LSTOP,STOP     ;LOAD STOP ADDRESS
 055112  012777  055306  124400         MOV     #LCOUNT,@$LLVEC ;LOAD L CLOCK INTERRUPT VECTOR
 055120  012777  000300  124374         MOV     #PR6,@$LLVEC+2
 055126  013777  001514  124356         MOV     $LPVEC+2,@$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
 055134  005077  124354                 CLR     @$LPVEC+2
 055140  000422                         BR      30$
 055142  012716  055150         20$:    MOV     #25$,(SP)       ;DUMMY RTI ADDRESS
 055146  000002                         RTI                     ;RESTORE PRIORITY
                                        ;NO CLOCK AVAILABLE - AUGMENT RETURN ADDRESS
 055150                         25$:
 055150  005037  001532                 CLR     CLOCK           ;CLEAR SUBROUTINE ADDRESS
 055154  012737  001514  001512         MOV     #$LPVEC+2,$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
 055162  005037  001514                 CLR     $LPVEC+2
 055166  012737  001522  001520         MOV     #$LLVEC+2,$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
 055174  005037  001522                 CLR     $LLVEC+2
 055200  062766  000002  000004         ADD     #2,4(SP)        ;CHANGE RETURN ADDRESS
 055206                         30$.
 055206  012637  000006                 MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 055212  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 055216  000207                         RTS     PC
                                        ;ROUTINES TO START THE CLOCK (KW11-L OR KW11-P)
 055220  012777  177777  124262 PCLOCK: MOV     #-1,@$LPCSB     ;LOAD COUNT SET BUFFER
 055226  012777  000135  124252         MOV     #135,@$LPCSR    ;LOAD CONTROL REGISTER
 055234  000403                         BR      PLCLK           ;GO TO COMMON CODE
 055236  012777  000100  124252 LCLOCK: MOV     #100,@$LLCSR    ;LOAD CONTROL REGISTER
```

```
                                ;CLEAR TIMER COUNT
                                ;;PUSH OLD PSW AND PC ON STACK
                                ;;SAVE THE PSW IN R5
                                ;SAVE PRIORITY
                                ;MASK X
                                ;IS PRIORITY TOO HIGH??
                                ;NO!!
                                ;;PUT NEW PS ON STACK
                                ;;PUT NEW PC ON  STACK
                                ;;POP NEW PC AND PS

                                ;ROUTINES TO HANDLE CLOCK INTERRUPTS (KW11-L OR KW11-P)

                                ;ADD 17MS TO ELAPSED TIME
                                ;BRANCH IF NO OVERFLOW
                                ;RESTORE MAXIMUM COUNT
                                ;DECREMENT REMAINING TIME
                                ;BRANCH IF POSITIVE
                                ;CLEAR REMAINING TIME
                                ;RETURN TO USER

                                ;ROUTINES TO STOP THE CLOCK (KW11-L OR KW11-P)

                                ;STOP P CLOCK
                                ;GO TO COMMON STOP CODE

                                ;STOP L CLOCK

                                ;;PUT NEW PS ON STACK
                                ;;PUT NEW PC ON  STACK
                                ;;POP NEW PC AND PS
```

SEQ 0224

```
                                        .SBTTL  SET VOLUME VALID SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE INITIALIZES THE SUBSYSTEM AND SETS VOLUME VALID,
                                        ;RETURNING WITH THE DRIVE STILL IN DIAGNOSTIC MODE.  THE SUBROUTINE
                                        ;RETURNS TO THE WORD FOLLOWING THE CALL, EXCEPT WHEN AN ERROR IS
                                        ;DETECTED, IN WHICH CASE IT RETURNS TO THE SECOND WORD FOLLOWING THE
                                        ;CALL.
                                        ;
                                        ;CALL:  JSR     PC,SETVV        JUMP TO SUBROUTINE
                                        ;       BR      ??              RETURN HERE IF NO ERROR
                                        ;       ERROR                   RETURN HERE IF ERROR
                                        ;
055370                                  SETVV:
055370  004737  055614                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
055374  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
055402  012760  001001  000024                  MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
055410  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
055416  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
055424  012760  000023  000000                  MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
055432  016037  000012  001142                  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
055440  042737  177677  001142                  BIC     #^CVV,$BDDAT
055446  001020                                  BNE     10$             ;BRANCH IF VOLUME VALID SET
055450  010037  001136                          MOV     R0,$BDADR       ;SETUP FOR ERROR MSG
055454  062737  000012  001136                  ADD     #RMDS,$BDADR
055462  012737  000100  001140                  MOV     #VV,$GDDAT
055470  062716  000002                          ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
055474  112776  000170  000000                  MOVB    #170,@(SP)      ;WRITE ERROR NUMBER
055502  012737  000022  001174                  MOV     #PACACK,$TMP0
055510  000207                          10$:    RTS     PC              ;RETURN
```

SEQ 0225

```
                                        .SBTTL  SET OFFSET MODE SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE EXECUTES AN OFFSET COMMAND AND VERIFIES THAT OFFSET
                                        ;MODE SETS.  THE DRIVE SHOULD BE IN DIAGNOSTIC MODE WHEN CALLING THE
                                        ;SUBROUTINE, WHICH WILL LEAVE DMD ON.  THE SUBROUTINE RETURNS TO THE
                                        ;WORD FOLLOWING THE CALL UNLESS THERE IS AN ERROR, IN WHICH CASE IT
                                        ;RETURNS TO THE SECOND WORD FOLLOWING THE CALL
                                        ;
                                        ;CALL:  JSR     PC,SETOM        JUMP TO SUBROUTINE
                                        ;       BR      ??              RETURN HERE IF NO ERROR
                                        ;       ERROR                   RETURN HERE IF ERROR
                                        ;
055512                                  SETOM:
055512  012760  001001  000024                  MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
055520  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
055526  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
055534  012760  000015  000000                  MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
055542  016037  000012  001142                  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
055550  042737  177776  001142                  BIC     #^COM,$BDDAT
055556  001015                                  BNE     10$             ;BRANCH IF OFFSET ON
055560  012737  000001  001140                  MOV     #OM,$GDDAT
055566  010037  001136                          MOV     R0,$BDADR
055572  062737  000012  001136                  ADD     #RMDS,$BDADR
055600  062716  000002                          ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
055604  112776  000200  000000                  MOVB    #200,@(SP)      ;WRITE ERROR NUMBER
055612                                  10$:
055612  000207                                  RTS     PC              ;RETURN TO USER
```

```
                                        .SBTTL  CLEAR CONTROLLER SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
                                        ;AND DRIVES, THEN SELECTS THE DRIVE.
                                        ;CALL:
                                        ;       JSR     PC,CNTCLR       ;CALL TO ROUTINE
                                        ;
 055614                                 CNTCLR:
 055614  010046                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 055616  013700  001276                         MOV     $BASE,R0        ;R0 = UNIBUS BASE ADDRESS
 055622  012760  000040  000010                 MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
 055630  117760  123626  000010                 MOVB    @TSTQUE,RMCS2(R0)       ;SELECT DEVICE UNDER TEST
 055636  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
 055640  000207                                 RTS     PC              ;RETURN
                                        ;
                                        .SBTTL  SHUTDOWN SUBROUTINE
                                        ;
 055642                                 SHUT:
 055642  104401  055650                         TYPE    ,65$            ;;TYPE ASCIZ STRING
 055646  000411                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>@TEST WAS HALTED@<CRLF>
 055672                                 64$:
 055672  005737  000042                         TST     @#42            ;RUNNING CHAIN MODE OR ACT11 AUTO ACCEPT ?
 055676  001402                                BEQ     10$             ;NO !!
 055700  000137  053636                         JMP     $EOP            ;YES - GO TO END OF PASS
                                        ;
 055704  000137  004626                 10$:    JMP     START           ;GO TO START
                                        ;
```

```
                                .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

055710                          $SAVREG:
055710  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
055712  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
055714  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
055716  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
055720  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
055722  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
055724  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
055730  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
055734  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
055740  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
055744  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
055746                          $RESREG:
055746  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
055752  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
055756  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
055762  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
055766  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
055770  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
055772  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
055774  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
055776  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
056000  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
056002  000002                          RTI
2                               .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

056004  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
056006  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
056012  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
```

```
056014  112737  000060  056056  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII '0'.
056022  006101                          ROL     R1              ;;GET THIS BIT
056024  001406                          BEQ     2$              ;;DONE?
056026  105537  056056                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
056032  104401  056056                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
056036  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
056040  000765                          BR      1$              ;;GO DO THE NEXT BIT
056042  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
056044  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
056052  012616                          MOV     (SP)+,(SP)
056054  000002                          RTI                     ;;RETURN TO USER
056056     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
    3                           .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

056060                          $TYPDS:
056060  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
056062  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
056064  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
056066  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
056070  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
056072  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
056076  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
056102  100004                          BPL     1$              ;;BR IF INPUT IS POS.
056104  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
056106  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
056114  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
056116  012703  056274                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
056122  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
056126  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
056130  016001  056264                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
056134  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
056136  002402                          BLT     4$              ;;BR IF DONE
056140  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
056142  000774                          BR      3$
056144  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
056146  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
056150  001002                          BNE     5$              ;;FALL THROUGH IF 0
056152  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
056154  100407                          BMI     7$              ;;BR IF YES
056156  106316                  5$:     ASLB    (SP)            ;;MSD?
056160  103003                          BCC     6$              ;;BR IF NO
056162  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
056170  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
056174  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
056200  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
056202  005720                          TST     (R0)+           ;;JUST INCREMENTING
056204  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
```

```
056210  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
056212  003002                          BGT     8$              ;;GO TO EXIT
056214  010502                          MOV     R5,R2           ;;GET THE LSD
056216  000764                          BR      6$              ;;GO CHANGE TO ASCII
056220  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
056222  100003                          BPL     9$              ;;BR IF NO
056224  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
056232  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
056234  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
056236  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
056240  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
056242  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
056244  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
056246  104401  056274                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
056252  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
056260  012616                          MOV     (SP)+,(SP)
056262  000002                          RTI                     ;;RETURN TO USER
056264  023420                  $DTBL:  10000.
056266  001750                          1000.
056270  000144                          100.
056272  000012                          10.
056274                          $DBLK:  .BLKW   4
    4                           .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

056304  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
056310  116637  000001  056527          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
056316  112637  056531                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
056322  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
056326  000406                          BR      $TYPON
056330  112737  000001  056527  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
056336  112737  000006  056531          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
056344  112737  000005  056526  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
056352  010346                          MOV     R3,-(SP)        ;;SAVE R3
056354  010446                          MOV     R4,-(SP)        ;;SAVE R4
```

```
056356  010546                          MOV     R5,-(SP)        ;;SAVE R5
056360  113704  056531                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
056364  005404                          NEG     R4
056366  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
056372  110437  056530                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
056376  113704  056527                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
056402  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
056406  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
056410  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
056412  000404                          BR      3$              ;;GO DO MSB
056414  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
056416  006105                          ROL     R5
056420  006105                          ROL     R5
056422  010503                          MOV     R5,R3
056424  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
056426  105337  056530                  DECB    $OMODE          ;;TYPE THIS DIGIT?
056432  100016                          BPL     7$              ;;BR IF NO
056434  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
056440  001002                          BNE     4$              ;;TEST FOR 0
056442  005704                          TST     R4              ;;SUPPRESS THIS 0?
056444  001403                          BEQ     5$              ;;BR IF YES
056446  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
056450  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
056454  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
056460  110337  056524                  MOVB    R3,8$           ;;SAVE FOR TYPING
056464  104401  056524                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
056470  105337  056526          7$:     DECB    $OCNT           ;;COUNT BY 1
056474  003347                          BGT     2$              ;;BR IF MORE TO DO
056476  002402                          BLT     6$              ;;BR IF DONE
056500  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
056502  000744                          BR      2$              ;;GO DO THE LAST DIGIT
056504  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
056506  012604                          MOV     (SP)+,R4        ;;RESTORE R4
056510  012603                          MOV     (SP)+,R3        ;;RESTORE R3
056512  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
056520  012616                          MOV     (SP)+,(SP)
056522  000002                          RTI                     ;;RETURN
056524     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
056525     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
056526     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
056527     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
056530  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
5                                       .SBTTL  TYPE ROUTINE

                                ;;******************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
```

```
                                ;*
056532  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
056536  100002                          BPL     1$              ;;BR IF YES
056540  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
056542  000430                          BR      3$              ;;LEAVE
056544  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
056546  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
056552  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
056560  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
056562  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
056570  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
056572  010037  056602                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
056576  004737  061672                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
056602  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
056604  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
056612  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
056614  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
056616  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
056620  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
056622  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
056624  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
056630  000002                          RTI                     ;;RETURN
056632  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
056636  001430                          BEQ     8$
056640  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
056644  001006                          BNE     5$
056646  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
056650  104401                          TYPE                    ;;TYPE A CR AND LF
056652  001217                          $CRLF
056654  105037  057062                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
056660  000755                          BR      2$              ;;GET NEXT CHARACTER
056662  004737  056744          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
056666  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
056672  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
056674  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
056700  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
056704  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
056706  004737  056744                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
056712  105337  057062                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
056716  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

056720  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
056724  004737  056744          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
056730  132737  000007  057062          BITB    #7,$CHARCNT     ;,BRANCH IF NOT AT
056736  001372                          BNE     9$              ;;TAB STOP
056740  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
056742  000724                          BR      2$              ;;GET NEXT CHARACTER
056744                          $TYPEC:
056744  105777  122210                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
056750  100022                          BPL     10$             ;;BR IF NOT
056752  017746  122204                  MOV     @$TKB,-(SP)     ;;GET CHAR
056756  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
056762  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
```

```
056766  001012                          BNE     102$            ;;BR IF NOT
056770                          101$:
056770  105777  122164                  TSTB    @$TKS           ;;WAIT FOR CHAR
056774  100375                          BPL     101$
056776  117716  122160                  MOVB    @$TKB,(SP)      ;;GET CHAR
057002  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT
057006  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?
057012  001366                          BNE     101$            ;;BR IF NOT
057014                          102$:
057014  005726                          TST     (SP)+           ;;FIX STACK
057016                          10$:
057016  105777  122142                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
057022  100375                          BPL     10$
057024  116677  000002  122134          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
057032  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
057040  001003                          BNE     1$              ;;BRANCH IF NO
057042  105037  057062                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
057046  000406                          BR      $TYPEX          ;;EXIT
057050  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
057056  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
057060  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
057062  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
057064  000207                  $TYPEX: RTS     PC

     6                          .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

057066                          $SCOPE:
057066  104410                          CKSWR                           ;;TEST  FOR CHANGE IN SOFT-SWR
057070  032777  040000  122056  1$:     BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
057076  001131                          BNE     $OVER                   ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
057100  000416                  $XTSTR: BR      6$                      ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
057102  013746  000004                  MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
057106  012737  057126  000004          MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
057114  005737  177060                  TST     @#177060                ;;TIME OUT ON XOR?
057120  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
057124  000500                          BR      $SVLAD                  ;;GO TO THE NEXT TEST
057126  022626                  5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
057130  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
057134  000440                          BR      7$                      ;;LOOP ON THE PRESENT TEST
057136                          6$:;#####END OF CODE FOR THE XOR TESTER#####
057136  032777  000400  122010          BIT     #BIT08,@SWR             ;;LOOP ON SPEC. TEST?
057144  001421                          BEQ     2$                      ;;BR IF NO
057146  005046                          CLR     -(SP)                   ;;CLEAR A TEMP. LOCATION
```

```
057150  117716  122000                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
057154  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
057156  022716  000120                  CMP     #120,(SP)       ;;CHECK THE NUMBER IN THE SWR
057162  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
057164  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
057170  005316                          DEC     (SP)            ;;BACKUP BY ONE
057172  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
057174  062716  057400                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
057200  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
057204  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
057206  005726          8$:             TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
057210  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
057214  001421                          BEQ     3$              ;;BR IF NO
057216  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
057224  101015                          BHI     3$              ;;BR IF NO
057226  032777  001000  121720          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
057234  001404                          BEQ     4$              ;;BR IF NO
057236  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
057244  000446                          BR      $OVER
057246  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
057252  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
057256  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
057260  032777  004000  121666  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
057266  001011                          BNE     1$              ;;BR IF YES
057270  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
057274  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
057276  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
057302  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
057310  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
057312  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
057320  013737  057376  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
057326  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
057332  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
057340  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
057344  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
057350  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
057354  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
057362  013777  001116  121566  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
057370  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
057374  000002                          RTI                     ;;FIXES PS
057376  000012          $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
057400                  $SW08TBL:
057400  007310                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
057402  007620                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
057404  010002                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
057406  010152                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
057410  010302                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
057412  011366                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
057414  012436                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
057416  012574                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
057420  012660                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
057422  013120                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
057424  013466                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
057426  014250                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
057430  014350                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
057432  014674                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
057434  015406                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
```

```
057436  015510                  .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
057440  015774                  .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
057442  016112                  .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
057444  016240                  .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
057446  016512                  .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
057450  017012                  .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
057452  017434                  .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
057454  017530                  .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
057456  020016                  .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
057460  020332                  .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
057462  020622                  .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
057464  021340                  .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
057466  021652                  .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
057470  022144                  .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
057472  022466                  .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
057474  023026                  .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
057476  023502                  .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
057500  023766                  .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
057502  024250                  .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
057504  024652                  .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
057506  025022                  .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
057510  025402                  .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
057512  026112                  .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
057514  026406                  .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
057516  027010                  .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
057520  027244                  .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
057522  027452                  .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
057524  030304                  .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
057526  030556                  .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
057530  030760                  .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
057532  031172                  .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
057534  031366                  .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
057536  031546                  .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
057540  031764                  .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
057542  032132                  .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
057544  032356                  .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
057546  032472                  .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
057550  032660                  .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
057552  033110                  .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
057554  033326                  .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
057556  033554                  .WORD   TST70+2         ;;STARTING ADDRESS OF TEST 70
057560  034024                  .WORD   TST71+2         ;;STARTING ADDRESS OF TEST 71
057562  034232                  .WORD   TST72+2         ;;STARTING ADDRESS OF TEST 72
057564  034522                  .WORD   TST73+2         ;;STARTING ADDRESS OF TEST 73
057566  034750                  .WORD   TST74+2         ;;STARTING ADDRESS OF TEST 74
057570  035056                  .WORD   TST75+2         ;;STARTING ADDRESS OF TEST 75
057572  035202                  .WORD   TST76+2         ;;STARTING ADDRESS OF TEST 76
057574  035332                  .WORD   TST77+2         ;;STARTING ADDRESS OF TEST 77
057576  035464                  .WORD   TST100+2        ;;STARTING ADDRESS OF TEST 100
057600  035616                  .WORD   TST101+2        ;;STARTING ADDRESS OF TEST 101
057602  036006                  .WORD   TST102+2        ;;STARTING ADDRESS OF TEST 102
057604  036210                  .WORD   TST103+2        ;;STARTING ADDRESS OF TEST 103
057606  036336                  .WORD   TST104+2        ;;STARTING ADDRESS OF TEST 104
057610  036542                  .WORD   TST105+2        ;;STARTING ADDRESS OF TEST 105
057612  036672                  .WORD   TST106+2        ;;STARTING ADDRESS OF TEST 106
057614  037066                  .WORD   TST107+2        ;;STARTING ADDRESS OF TEST 107
057616  037264                  .WORD   TST110+2        ;;STARTING ADDRESS OF TEST 110
```

```
057620  037542                          .WORD   TST111+2                ;;STARTING ADDRESS OF TEST 111
057622  040046                          .WORD   TST112+2                ;;STARTING ADDRESS OF TEST 112
057624  041440                          .WORD   TST113+2                ;;STARTING ADDRESS OF TEST 113
057626  043166                          .WORD   TST114+2                ;;STARTING ADDRESS OF TEST 114
057630  045346                          .WORD   TST115+2                ;;STARTING ADDRESS OF TEST 115
057632  045726                          .WORD   TST116+2                ;;STARTING ADDRESS OF TEST 116
057634  050106                          .WORD   TST117+2                ;;STARTING ADDRESS OF TEST 117
057636  052374                          .WORD   TST120+2                ;;STARTING ADDRESS OF TEST 120
      7                         .SBTTL  ERROR HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

057640                          $ERROR:
057640  104410                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
057642  105237  001117          7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
057646  001775                          BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
057650  013777  001116  121300          MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
057656  032777  002000  121270          BIT     #BIT10,@SWR             ;;BELL ON ERROR?
057664  001402                          BEQ     1$                      ;;NO - SKIP
057666  104401  001212                  TYPE    ,$BELL                  ;;RING BELL
057672  005237  001126          1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
057676  011637  001132                  MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
057702  162737  000002  001132          SUB     #2,$ERRPC
057710  117737  121216  001130          MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
057716  032777  020000  121230          BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
057724  001004                          BNE     20$                     ;;SKIP TYPEOUTS
057726  004737  054046                  JSR     PC,ERRTYP               ;;GO TO USER ERROR ROUTINE
057732  104401  001217                  TYPE    ,$CRLF
057736                          20$:
057736  122737  000001  001242          CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
057744  001007                          BNE     2$                      ;;NO,SKIP APT ERROR REPORT
057746  113737  001130  057760          MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
057754  004737  061702                  JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
057760     000                  21$:    .BYTE   0
057761     000                          .BYTE   0
057762  000777                  22$:    BR      22$                     ;;APT ERROR LOOP
057764  005777  121164          2$:     TST     @SWR                    ;;HALT ON ERROR
057770  100002                          BPL     3$                      ;;SKIP IF CONTINUE
057772  000000                          HALT                            ;;HALT ON ERROR!
057774  104410                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
057776  032777  001000  121150  3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
060004  001402                          BEQ     4$                      ;;BR IF NO
060006  013716  001124                  MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
060012  005737  001210          4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
060016  001402                          BEQ     5$                      ;;BR IF NONE
060020  013716  001210                  MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
060024                          5$:
```

```
060024  022737  054026  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
060032  001001                          BNE     6$              ;;BRANCH IF NO
060034  000000                          HALT                    ;;YES
060036                          6$:
060036  000002                          RTI                     ;;RETURN
      8                         .SBTTL  TTY INPUT ROUTINE

                                ;;**************************************************************************
                                .ENABL  LSB
060040  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
060042  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
060044  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
060046                          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        060047                  $TKQEND=.
                                .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
060050  005037  060040          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
060054  012737  060046  060042          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
060062  013737  060042  060044          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
060070  012737  060120  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
060076  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;'BR' LEVEL 4
060104  005777  121052                  TST     @$TKB           ;;CLEAR DONE FLAG
060110  012777  000100  121042          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
060116  000207                          RTS     PC              ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A 'CONTROL-C' (^C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (SHUT)
                                ;
060120  117746  121036          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
060124  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
060130  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
060134  001007                          BNE     1$              ;;BRANCH IF NO
060136  104401  061234                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
060142  004737  060050                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
060146  005726                          TST     (SP)+           ;;CLEAN UP STACK
060150  000137  055642                  JMP     SHUT            ;;CONTROL C RESTART
060154  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
060160  001004                          BNE     2$              ;;BRANCH IF NO
060162  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
060170  001500                          BEQ     6$              ;;GO TO SWR CHANGE

060172                          2$:
060172  022737  000001  060040          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
060200  001004                          BNE     3$              ;;BRANCH IF NO
060202  104401  001212                  TYPE    ,$BELL          ;;RING THE TTY BELL
```

```
060206  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
060210  000451                          BR      5$              ;;EXIT
060212  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
060216  001021                          BNE     32$             ;;BRANCH IF NO
060220  005077  120734                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
060224  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
060226  105777  120726          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
060232  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
060234  117746  120722                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
060240  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
060244  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
060250  001366                          BNE     31$             ;;BRANCH IF NO
060252  012777  000100  120700          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
060260  000002                          RTI                     ;;RETURN
060262  005237  060040          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
060266  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
060272  002405                          BLT     4$              ;;BRANCH IF YES
060274  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
060300  003002                          BGT     4$              ;;BRANCH IF YES
060302  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
060306  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
060312  005237  060042                  INC     $TKQIN          ;;UPDATE THE POINTER
060316  023727  060042  060047          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
060324  001003                          BNE     5$              ;;BRANCH IF NO
060326  012737  060046  060042          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
060334  000002                  5$:     RTI                     ;;RETURN

                                        ;;**********************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                        ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
060336  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
060344  001124                          BNE     15$             ;;EXIT IF NOT
060346  105777  120606                  TSTB    @$TKS           ;;IS A CHAR WAITING?
060352  100121                          BPL     15$             ;;IF NOT, EXIT
060354  117746  120602                  MOVB    @$TKB,-(SP)     ;;YES
060360  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
060364  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
060370  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                        ;;**********************************************************************
                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
060372  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
060400  001674                          BEQ     2$              ;;BRANCH IF YES
060402  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
060404  004737  060050                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
060410  005077  120544                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
060414  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

060422  104401  061246                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
060426  104401  061253  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
060432  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
060436  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
060440  104401  061264              TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
060444  005046          19$:        CLR     -(SP)           ;;CLEAR COUNTER
060446  005046                      CLR     -(SP)           ;;THE NEW SWR
060450  105777  120504      7$:     TSTB    @$TKS           ;;CHAR THERE?
060454  100375                      BPL     7$              ;;IF NOT TRY AGAIN

060456  117746  120500              MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
060462  042716  177600              BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

060466  021627  000003              CMP     (SP),#3         ;;IS IT A CONTROL-C?
060472  001015                      BNE     9$              ;;BRANCH IF NOT
060474  104401  061234              TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
060500  062706  000006              ADD     #6,SP           ;;CLEAN UP STACK
060504  123727  001151  000001      CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
060512  001003                      BNE     8$              ;;BRANCH IF NO
060514  012777  000100  120436      MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
060522  000137  055642      8$:     JMP     SHUT            ;;CONTROL-C RESTART


060526  021627  000025      9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
060532  001005                      BNE     10$             ;;BRANCH IF NOT
060534  104401  061241              TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
060540  062706  000006      20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
060544  000737                      BR      19$             ;;LET'S TRY IT AGAIN


060546  021627  000015      10$:    CMP     (SP),#15        ;;IS IT A <CR>?
060552  001022                      BNE     16$             ;;BRANCH IF NO
060554  005766  000004              TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
060560  001403                      BEQ     11$             ;;BRANCH IF YES
060562  016677  000002  120364      MOV     2(SP),@SWR      ;;SAVE NEW SWR
060570  062706  000006      11$:    ADD     #6,SP           ;;CLEAR UP STACK
060574  104401  001217      14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
060600  123727  001151  000001      CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
060606  001003                      BNE     15$             ;;BRANCH IF NOT
060610  012777  000100  120342      MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
060616  000002          15$:        RTI                     ;;RETURN
060620  004737  056744      16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
060624  021627  000060              CMP     (SP),#60        ;;CHAR < 0?
060630  002420                      BLT     18$             ;;BRANCH IF YES
060632  021627  000067              CMP     (SP),#67        ;;CHAR > 7?
060636  003015                      BGT     18$             ;;BRANCH IF YES
060640  042726  000060              BIC     #60,(SP)+       ;;STRIP-OFF ASCII
060644  005766  000002              TST     2(SP)           ;;IS THIS THE FIRST CHAR
060650  001403                      BEQ     17$             ;;BRANCH IF YES
060652  006316                      ASL     (SP)            ;;NO, SHIFT PRESENT
060654  006316                      ASL     (SP)            ;;   CHAR OVER TO MAKE
060656  006316                      ASL     (SP)            ;;   ROOM FOR NEW ONE.
060660  005266  000002      17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
060664  056616  177776              BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
060670  000667                      BR      7$              ;;GET THE NEXT ONE
060672  104401  001216      18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
060676  000720                      BR      20$             ;;SIMULATE CONTROL-U
                          .DSABL    LSB

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

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

060700  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
060702  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
060710  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
060714  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
060716  012746  060724                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
060722  000002                          RTI                     ;;POP NEW PC AND PS
060724                          64$:
060724  005737  060040          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
060730  001775                          BEQ     1$
060732  005337  060040                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
060736  117766  177102  000004        MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
060744  005237  060044                  INC     $TKQOUT         ;;UPDATE THE POINTER
060750  023727  060044  060047        CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
060756  001003                          BNE     2$              ;;BRANCH IF NO
060760  012737  060046  060044        MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
060766  000002                  2$:     RTI                     ;;RETURN
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

060770  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
060772  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
060774  012703  061224          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
061000  022703  061234          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
061004  101456                          BLOS    4$              ;;BR IF YES
061006  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
061010  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
061012  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
061016  001022                          BNE     5$              ;;BR IF NO
061020  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
061022  001007                          BNE     6$              ;;BR IF NO
061024  112737  000134  061222        MOVB    #'\,9$          ;;TYPE A BACK SLASH
061032  104401  061222                  TYPE    ,9$
061036  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
061042  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
061044  020327  061224                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
061050  103434                          BLO     4$              ;;BR IF YES
061052  111337  061222                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
061056  104401  061222                  TYPE    ,9$             ;;GO TYPE
061062  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
061064  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
061066  001406                          BEQ     7$              ;;BR IF NO
061070  112737  000134  061222        MOVB    #'\,9$          ;;TYPE A BACK SLASH
061076  104401  061222                  TYPE    ,9$
061102  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
061104  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
061110  001003                          BNE     8$              ;;BR IF NO
```

```
061112  104401  061241                  TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
061116  000726                          BR      1$              ;;GO START OVER
061120  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
061124  001011                          BNE     3$              ;;BRANCH IF NO
061126  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
061130  104401  001217                  TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
061134  104401  061224                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
061140  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
061142  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
061146  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
061150  111337  061222          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
061154  104401  061222                  TYPE    ,9$
061160  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
061164  001305                          BNE     2$              ;;LOOP IF NOT RETURN
061166  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
061172  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
061176  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
061200  012603                          MOV     (SP)+,R3        ;;RESTORE R3
061202  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
061204  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
061212  012766  061224  000004          MOV     #$TTYIN,4(SP)
061220  000002                          RTI                     ;;RETURN
061222     000          9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
061223     000                  .BYTE   0               ;;TERMINATOR
061224          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
061234     136     103     015  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
061241     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
061246     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
061253     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
061264     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
                                .EVEN
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

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

061276  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
061300  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
061306  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
061310  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
061312  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
061314  104412          1$:     RDLIN                   ;;READ AN ASCIZ LINE
061316  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
061320  005001                          CLR     R1              ;;CLEAR DATA WORD
061322  005002                          CLR     R2
061324  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
061326  001412                          BEQ     3$              ;;IF ZERO GET OUT
061330  006301                          ASL     R1              ;;*2
061332  006102                          ROL     R2
061334  006301                          ASL     R1              ;;*4
061336  006102                          ROL     R2
061340  006301                          ASL     R1              ;;*8
```

```
061342  006102                          ROL     R2
061344  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
061350  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
061352  000764                          BR      2$              ;;LOOP
061354  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
061356  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
061362  010237  061376                  MOV     R2,$HIOCT
061366  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
061370  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
061372  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
061374  000002                          RTI                     ;;RETURN
061376  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
10                                      .SBTTL  TRAP DECODER

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

061400  016646  000002          $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
061404  042716  000020                  BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
061410  012746  061416                  MOV     #1$,-(SP)       ;; T-BIT TRAPS
061414  000002                          RTI                     ;;SET THE NEW STATUS
061416  010046                  1$:     MOV     R0 -(SP)        ;;SAVE R0
061420  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
061424  005740                          TST     -(R0)           ;;BACKUP BY 2
061426  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
061430  006300                          ASL     R0              ;;POSITION FOR INDEXING
061432  016000  061452                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
061436  000200                          RTS     R0              ;;GO TO ROUTINE


                                        ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO

061440  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
061442  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
061450  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE ''TRAP'' INSTRUCTION.

                                        ;       ROUTINE
                                        ;       -------
061452  061440                  $TRPAD: .WORD   $TRAP2
061454  056532                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
061456  056330                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
061460  056304                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
061462  056344                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
061464  056060                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
061466  056004                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

061470  060426                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING

061472  060336                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
```

```
061474  060700                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
061476  060770                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
061500  061276                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
061502  055710                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
061504  055746                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
    11                          .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
061506  012737  061646  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
061514  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
061522  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
061524  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
061526  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
061530  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
061532  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
061534  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
061536  017746  117412                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
061542  010637  061652                  MOV     SP,$SAVR6       ;;SAVE SP
061546  012737  061560  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
061554  000000                          HALT
061556  000776                          BR      .-2             ;;HANG UP

                                ;;**********************************************************************
                                ;POWER UP ROUTINE
061560  012737  061646  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
061566  013706  061652                  MOV     $SAVR6,SP       ;;GET SP
061572  005037  061652                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
061576  005237  061652          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
061602  001375                          BNE     1$              ;;OF  WORD
061604  012677  117344                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
061610  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
061612  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
061614  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
061616  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
061620  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
061622  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
061624  012737  061506  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
061632  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
061640  104401                          TYPE                    ;REPORT THE POWER FAILURE
061642  061654          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
061644  000002                  RTI
061646  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
061650  000776                  BR      .-2                     ;; BEFOPE THE POWER DOWN WAS COMPLETE
061652  000000          $SAVR6: 0                               ;;PUT THE SP HERE
061654     015     012     120  $POWER: .ASCIZ  <15><12>'POWER'
                                .EVEN
    12                          .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;********************************** **********************************
061664  112737  000001  062130  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
061672  112737  000001  062126  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
061700  000403                  BR      $ATYC
061702  112737  000001  062130  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
061710                          $ATYC:
061710  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
061712  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
061714  105737  062126              TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
061720  001450                      BEQ     5$              ;;IF NOT:  BR
061722  122737  000001  001242      CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
061730  001031                      BNE     3$              ;;IF NOT:  BR
061732  132737  000100  001243      BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
061740  001425                      BEQ     3$              ;;IF NOT:  BR
061742  017600  000004              MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
061746  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
061754  005737  001222          1$: TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
061760  001375                      BNE     1$              ;;IF NOT:  WAIT
061762  010037  001236              MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
061766  105720                  2$: TSTB    (R0)+           ;;FIND END OF MESSAGE
061770  001376                      BNE     2$
061772  163700  001236              SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
061776  006200                      ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
062000  010037  001240              MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
062004  012737  000004  001222      MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
062012  000413                      BR      5$
062014  017637  000004  062040  3$: MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
062022  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
062030  013746  177776              MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
062034  004737  056532              JSR     PC,$TYPE        ;;CALL TYPE MACRO
062040  000000                  4$: .WORD   0
062042                          5$:
062042  105737  062130          10$:TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
062046  001416                      BEQ     12$             ;;IF NOT:  BR
062050  005737  001242              TST     >ENV            ;;RUNNING UNDER APT?
062054  001413                      BEQ     12$             ;;IF NOT:  BR
062056  005737  001222          11$:TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
062062  001375                      BNE     11$             ;;IF NOT:  WAIT
062064  017637  000004  001224      MOV     @4(SP),$FATAL   ;;GET ERROR #
062072  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
062100  005237  001222              INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
062104  105037  062130          12$:CLRB    $FFLG           ;;CLEAR FATAL FLAG
062110  105037  062127              CLRB    $LFLG           ;;CLEAR LOG FLAG
062114  105037  062126              CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
062120  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
062122  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
062124  000207                      RTS     PC              ;;RETURN
062126     000              $MFLG:  .BYTE   0               ;;MESSG. FLAG
062127     000              $LFLG:  .BYTE   0               ;;LOG FLAG
062130     000              $FFLG:  .BYTE   0               ;;FATAL FLAG
                                    .EVEN
           000200           APTSIZE = 200
           000001           APTENV  = 001
           000100           APTSPOOL= 100
           000040           APTCSUP = 040
```

```
                                        .SBTTL  CONSOLE MESSAGES
                                
062132  075     000             EQUALS: .ASCIZ  @=@
062134  101     114     114     ALL:    .ASCIZ  @ALL@<CRLF>
062141  040     077     040     QUES:   .ASCIZ  @ ? @
062145  054     040     000     COMMA:  .ASCIZ  @, @
062150  200     124     131     MSHELP: .ASCIZ  <CRLF>@TYPE HELP TEXT (Y/N) ? @
062201                          UBUSQST:
062201  200     103     110             .ASCIZ  <CRLF>@CHANGE ADDRESSES (Y/N) ? @
062234  200     125     123     CNSL00: .ASCIZ  <CRLF>@USE SAME DEVICES (Y/N) ? @
062267  200     102     125     CNSL01: .ASCIZ  <CRLF>@BUS ADDRESS @
062305  040     114     111     CNSL02: .ASCIZ  @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
062347  126     105     103     CNSL03: .ASCIZ  @VECTOR ADDRESS @
062367  040     114     111     CNSL04: .ASCIZ  @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
062423  102     122     040     CNSL05: .ASCIZ  @BR LEVEL @
062435  040     114     111     CNSL06: .ASCIZ  @ LIMITS - LO= 0, HI= 7@<CRLF><LF>
062466  200                     CNSL07: .ASCII  <CRLF>
062467  200     124     131             .ASCII  <CRLF>@TYPE "A" TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
062554  200     101     116             .ASCIZ  <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
062631  200                     CNSL08: .ASCII  <CRLF>
062632  040     077     111     CNSL09: .ASCIZ  @ ?ILLEGAL INPUT@<CRLF>
062653  200     104     122     MSDRVS: .ASCIZ  <CRLF>/DRIVE(S): /
062667  104     122     111     MSGDRV: .ASCIZ  /DRIVE/
062675  040     111     123     LODEV:  .ASCIZ  / IS LOAD DEVICE/
062715  040     116     117     NOTPRS: .ASCIZ  / NOT PRESENT/
062732  040     116     117     NOTAVL: .ASCIZ  / NOT AVAILABLE/

                                        .EVEN
```

```
                                .SBTTL  FUNCTION CODE TABLE

                                ;THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
                                ;EACH FUNCTION CODE.  BIT USAGE IS AS FOLLOWS:

                                ;       ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF ATA SHOULD BE SET WHEN THE FUNCTION CODE IS EXECUTED, OTHERWISE,
                                ;BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
                                ;NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
                                ;IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

                                ;       WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

                                ;       OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                                ;       IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                                ;       WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                                ;THE WRITE ERRORS WHICH ARE ENABLED ARE 'WLE', 'WCF', 'DPE', 'UPE'.

                                ;       IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

                                ;       AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
                                ;COMMAND.  THE ERRORS ENABLED BY THIS BIT ARE 'TRE', 'DLT', 'NEM',
                                ;'MXF', 'LBT', AND 'AOE'.

                                ;               BIT 08 IS NOT USED.

                                ;       HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                                ;HEADER ERRORS INCLUDE 'HCRC', 'HCE', 'FER', AND 'BSE'.

                                ;       ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                                ;IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
                                ;COMMAND.  THESE ERRORS INCLUDE 'MDPE', 'DCK', AND 'ECH'.

                                ;               BIT 05 IS NOT USED.

                                ;               BIT 04 IS NOT USED.

                                ;               BIT 03 IS NOT USED.

                                ;               BIT 02 IS NOT USED.

                                ;               BIT 01 IS NOT USED.

                                ;       ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

 0 2752                         FNCDTB:                                 ;FUNCTION CODE TABLE

 062752  020000                 .WORD   OPI                             ;NOP
```

```
     58 062754  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (22)
     59 062756  132000                .WORD   ATA!OPI!IVC!IAE                 ;SEEK
     60 062760  130000                .WORD   ATA!OPI!IVC                     ;RECALIBRATE
     61 062762  020000                .WORD   OPI                             ;DRIVE CLEAR
     62 062764  030000                .WORD   OPI!IVC                         ;RELEASE
     63 062766  130000                .WORD   OPI!ATA!IVC                     ;OFFSET
     64 062770  130000                .WORD   OPI!ATA!IVC                     ;RETURN TO CENTERLINE
     65 062772  020000                 WORD   OPI                             ;READ IN PRESET
     66 062774  020000                .WORD   OPI                             ;PACK ACKNOWLEDGE
     67 062776  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (24)
     68 063000  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (26)
     69 063002  132000                .WORD   ATA!OPI!IVC!IAE                 ;SEARCH
     70 063004  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (32)
     71 063006  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (34)
     72 063010  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (36)
     73 063012  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (40)
     74 063014  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (42)
     75 063016  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (44)
     76 063020  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (46)
     77 063022  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK DATA
     78 063024  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK HEADER AND DATA
     79 063026  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (54)
     80 063030  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (56)
     81 063032  037200                .WORD   OPI!IVC!WLE!IAE!AOE!HCE         ;WRITE DATA
     82 063034  037000                .WORD   OPI!IVC!WLE!IAE!AOE             ;WRITE HEADER AND DATA
     83 063036  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (64)
     84 063040  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (66)
     85 063042  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ DATA
     86 063044  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ HEADER AND DATA
     87 063046  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (74)
     88 063050  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (76)
     89
```

SEQ 0247

```
                                .SBTTL  ATTENTION (ATA) TABLE

                        ATNTBL: .BYTE   1.
                                .BYTE   2.
                                .BYTE   4.
                                .BYTE   8.
                                .BYTE   16.
                                .BYTE   32.
                                .BYTE   64.
                                .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

                        RGDTPT:
                        MIXED:
063062                          .WORD   0.
063062  000000                  .WORD   0.
063064  000001                  .WORD   1.
063066  000003                  .WORD   3.
063070  000007                  .WORD   7.
063072  000017                  .WORD   15.
063074  000037                  .WORD   31.
063076  000077                  .WORD   63.
063100  000177                  .WORD   127.
063102  000377                  .WORD   255.
063104  000777                  .WORD   511.
063106  001777                  .WORD   1023.
063110  003777                  .WORD   2047.
063112  007777                  .WORD   4095.
063114  017777                  .WORD   8191.
063116  037777                  .WORD   16383.
063120  077777                  .WORD   32767.
063122  177777          ONES:   .WORD   65535.
063124  177777                  .WORD   65535.
063126  077777                  .WORD   32767.
063130  037777                  .WORD   16383.
063132  017777                  .WORD   8191.
063134  007777                  .WORD   4095.
063136  003777                  .WORD   2047.
063140  001777                  .WORD   1023.
063142  000777                  .WORD   511.
063144  000377                  .WORD   255.
063146  000177                  .WORD   127.
063150  000077                  .WORD   63.
063152  000037                  .WORD   31.
063154  000017                  .WORD   15.
063156  000007                  .WORD   7.
063160  000003                  .WORD   3.
063162  000001                  .WORD   1.
063164  000000          ZEROS:  .WORD   0.
063166  000000                  .WORD   0.
063170  000001                  .WORD   1.
063172  000002                  .WORD   2.
063174  000004                  .WORD   4.
063176  000010                  .WORD   8.
063200  000020                  .WORD   16.
063202  000040                  .WORD   32.
063204  000100                  .WORD   64.
063206  000200                  .WORD   128.
063210  000400                  .WORD   256.
063212  001000                  .WORD   512.
063214  002000                  .WORD   1024.
063216  004000                  .WORD   2048.
063220  010000                  .WORD   4096.
063222  020000                  .WORD   8192.
063224  040000                  .WORD   16384.
063226  100000                  .WORD   32768.
063230  100000                  .WORD   32768.
063232  040000                  .WORD   16384.
```

```
 58 063234  020000                    .WORD   8192.
 59 063236  010000                    .WORD   4096.
 60 063240  004000                    .WORD   2048.
 61 063242  002000                    .WORD   1024.
 62 063244  001000                    .WORD   512.
 63 063246  000400                    .WORD   256.
 64 063250  000200                    .WORD   128.
 65 063252  000100                    .WORD   64.
 66 063254  000040                    .WORD   32.
 67 063256  000020                    .WORD   16.
 68 063260  000010                    .WORD   8.
 69 063262  000004                    .WORD   4.
 70 063264  000002                    .WORD   2.
 71 063266  000001                    .WORD   1.
 72 063270  000000                    .WORD   0.
 73 063272  177777                    .WORD   65535.
 74 063274  177776                    .WORD   65534.
 75 063276  177774                    .WORD   65532.
 76 063300  177770                    .WORD   65528.
 77 063302  177760                    .WORD   65520.
 78 063304  177740                    .WORD   65504.
 79 063306  177700                    .WORD   65472.
 80 063310  177600                    .WORD   65408.
 81 063312  177400                    .WORD   65280.
 82 063314  177000                    .WORD   65024.
 83 063316  176000                    .WORD   64512.
 84 063320  174000                    .WORD   63488.
 85 063322  170000                    .WORD   61440.
 86 063324  160000                    .WORD   57344.
 87 063326  140000                    .WORD   49152.
 88 063330  100000                    .WORD   32768.
 89 063332  000000                    .WORD   0.
 90 063334  000000                    .WORD   0.
 91 063336  100000                    .WORD   32768.
 92 063340  140000                    .WORD   49152.
 93 063342  160000                    .WORD   57344.
 94 063344  170000                    .WORD   61440.
 95 063346  174000                    .WORD   63488.
 96 063350  176000                    .WORD   64512.
 97 063352  177000                    .WORD   65024.
 98 063354  177400                    .WORD   65280.
 99 063356  177600                    .WORD   65408.
100 063360  177700                    .WORD   65472.
101 063362  177740                    .WOF    65504.
102 063364  177760                    .WORD   65520.
103 063366  177770                    .WORD   65528.
104 063370  177774                    .WORD   65532.
105 063372  177776                    .WORD   65534.
106 063374  177777                    .WORD   65535.
107 063376  125252                    .WORD   43690.
108 063400  152525                    .WORD   43690./2
109 063402  125252                    .WORD   43690.
110 063404  177777                    .WORD   65535.
111 063406  177776                    .WORD   65534.
112 063410  177775                    .WORD   65533.
113 063412  177773                    .WORD   65531.
114 063414  177767                    .WORD   65527.
```

```
    115 063416  177757                      .WORD   65519.
    116 063420  177737                      .WORD   65503.
    117 063422  177677                      .WORD   65471.
    118 063424  177577                      .WORD   65407.
    119 063426  177377                      .WORD   65279.
    120 063430  176777                      .WORD   65023.
    121 063432  175777                      .WORD   64511.
    122 063434  173777                      .WORD   63487.
    123 063436  167777                      .WORD   61439.
    124 063440  157777                      .WORD   57343.
    125 063442  137777                      .WORD   49151.
    126 063444  077777                      .WORD   32767.
    127 063446  077777                      .WORD   32767.
    128 063450  137777                      .WORD   49151.
    129 063452  157777                      .WORD   57343.
    130 063454  167777                      .WORD   61439.
    131 063456  173777                      .WORD   63487.
    132 063460  175777                      .WORD   64511.
    133 063462  176777                      .WORD   65023.
    134 063464  177377                      .WORD   65279.
    135 063466  177577                      .WORD   65407.
    136 063470  177677                      .WORD   65471.
    137 063472  177737                      .WORD   65503.
    138 063474  177757                      .WORD   65519.
    139 063476  177767                      .WORD   65527.
    140 063500  177773                      .WORD   65531.
    141 063502  177775                      .WORD   65533.
    142 063504  177776                      .WORD   65534.
    143 063506  177777                      .WORD   65535.
    144 063510                      ENRGDT:
    145                             .EVEN
    146
```

```
                                .SBTTL  ERROR MESSAGE TABLE
                        
063510  076634  071542  000000  EMT1:   .WORD   EMS300,EMS1,0
063516  076652  076675  076722  EMT2:   .WORD   EMS301,EMS302,EMS303,EMS1,EMS304
063530  101656  101061  101222          .WORD   EMS511,EMS500,EMS501,EMS502,EMS503,0
063544  076652  076742  076675  EMT3:   .WORD   EMS301,EMS306,EMS302
063552  101656  101377  101222          .WORD   EMS511,EMS505,EMS501,EMS502,0
063564  076634  076675  076756  EMT4:   .WORD   EMS300,EMS302,EMS307,EMS2
063574  101656  101247  101222          .WORD   EMS511,EMS502,EMS501,EMS503,0
063606  076652  077017  077034  EMT5:   .WORD   EMS301,EMS310,EMS311
063614  101656  101247  101222          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504
063626  077100  000000                  .WORD   EMS312,0
063632  076652  076742  077034  EMT6:   .WORD   EMS301,EMS306,EMS311
063640  101656  101247  101222          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504,0
063654  076652  077136  076675  EMT7:   .WORD   EMS301,EMS313,EMS302
063662  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,EMS504,EMS503,0
063676  077244  077265  077167  EMT10:  .WORD   EMS316,EMS317,EMS314
063704  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
063714  077244  077265  077216  EMT11:  .WORD   EMS316,EMS317,EMS315
063722  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
063732  077244  077305  077167  EMT12:  .WORD   EMS316,EMS320,EMS314
063740  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
063750  077244  077305  077216  EMT13:  .WORD   EMS316,EMS320,EMS315
063756  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
063766  077244  077325  077167  EMT14:  .WORD   EMS316,EMS321,EMS314
063774  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
064004  077244  077325  077216  EMT15:  .WORD   EMS316,EMS321,EMS315
064012  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
064022  077244  077345  077167  EMT16:  .WORD   EMS316,EMS322,EMS314
064030  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
064040  077244  077345  077216  EMT17:  .WORD   EMS316,EMS322,EMS315
064046  101656  101222  101247          .WORD   EMS511,EMS501,EMS502,0
064056  076652  077404  076122  EMT20:  .WORD   EMS301,EMS324,EMS250
064064  101656  101344                  .WORD   EMS511,EMS504
064070  077100  077435  000000          .WORD   EMS312,EMS326,0
064076  076652  077365  076122  EMT21:  .WORD   EMS301,EMS323,EMS250
064104  101656  101344                  .WORD   EMS511,EMS504
064110  077100  077424  000000          .WORD   EMS312,EMS325,0
064116  076652  077136  076122  EMT22:  .WORD   EMS301,EMS313,EMS250
064124  101656  101344  000000          .WORD   EMS511,EMS504,0
064132  076652  077404  076160  EMT23:  .WORD   EMS301,EMS324,EMS251
064140  101656  101222                  .WORD   EMS511,EMS501
064144  077100  077435  000000          .WORD   EMS312,EMS326,0
064152  076652  077365  076160  EMT24:  .WORD   EMS301,EMS323,EMS251
064160  101656  101222                  .WORD   EMS511,EMS501
064164  077100  077424  000000          .WORD   EMS312,EMS325,0
064172  076652  077136  076160  EMT25:  .WORD   EMS301,EMS313,EMS251
064200  101656  101222  000000          .WORD   EMS511,EMS501,0
064206  076652  076742  076224  EMT26:  .WORD   EMS301,EMS306,EMS252,EMS253,EMS327,EMS254
064222  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,EMS502
064232  077453  077167  000000          .WORD   EMS330,EMS314,0
064240  076634  076224          EMT27:  .WORD   EMS300,EMS252
064244  101656  101222  000000          .WORD   EMS511,EMS501,0
064252  076634  076224          EMT30:  .WORD   EMS300,EMS252
064256  101656  101222  101344          .WORD   EMS511,EMS501,EMS504,0
064266  076634  076224          EMT31:  .WORD   EMS300,EMS252
064272  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
```

```
 58 064302  076652  077404  076224  EMT32:  .WORD   EMS301,EMS324,EMS252
 59 064310  101656  101222  000000          .WORD   EMS511,EMS501,0
 60 064316  076652  077404  076224  EMT33:  .WORD   EMS301,EMS324,EMS252
 61 064324  101656  101222  101344          .WORD   EMS511,EMS501,EMS504,0
 62 064334  076652  077404  076224  EMT34:  .WORD   EMS301,EMS324,EMS252
 63 064342  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
 64 064352  076652  077365  076224  EMT35:  .WORD   EMS301,EMS323,EMS252
 65 064360  101656  101222  000000          .WORD   EMS511,EMS501,0
 66 064366  076652  077136  076224  EMT36:  .WORD   EMS301,EMS313,EMS252
 67 064374  101656  101222  000000          .WORD   EMS511,EMS501,0
 68 064402  076652  077404  076353  EMT37:  .WORD   EMS301,EMS324,EMS255
 69 064410  077100  077435  000000          .WORD   EMS312,EMS326,0
 70 064416  076652  077365  076353  EMT40:  .WORD   EMS301,EMS323,EMS255
 71 064424  101656  101344                  .WORD   EMS511,EMS504
 72 064430  077100  077424  000000          .WORD   EMS312,EMS325,0
 73 064436  076652  077136  076353  EMT41:  .WORD   EMS301,EMS313,EMS255
 74 064444  101656  101344  000000          .WORD   EMS511,EMS504,0
 75 064452  076652  077365  076122  EMT42:  .WORD   EMS301,EMS323,EMS250,EMS327,EMS255
 76 064464  101656  101344  101222          .WORD   EMS511,EMS504,EMS501,EMS503,0
 77 064476  076634  071662          EMT43:  .WORD   EMS300,EMS3
 78 064502  101656  101222  000000          .WORD   EMS511,EMS501,0
 79 064510  077472  071727  076257  EMT44:  .WORD   EMS331,EMS4,EMS253
 80 064516  101656  101222  000000          .WORD   EMS511,EMS501,0
 81 064524  076634  076257          EMT45:  .WORD   EMS300,EMS253
 82 064530  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
 83 064540  076634  076257          EMT46:  .WORD   EMS300,EMS253
 84 064544  101656  101222  101344          .WORD   EMS511,EMS501,EMS504,0
 85 064554  076652  077404  076257  EMT47:  .WORD   EMS301,EMS324,EMS253
 86 064562  101656  101222  101317          .WORD   EMS511,EMS501,EMS503
 87 064570  077100  077435  000000          .WORD   EMS312,EMS326,0
 88 064576  076652  077404  076257  EMT50:  .WORD   EMS301,EMS324,EMS253
 89 064604  101656  101222  101344          .WORD   EMS511,EMS501,EMS504
 90 064612  077100  077435  000000          .WORD   EMS312,EMS326,0
 91 064620  076652  077365  076257  EMT51:  .WORD   EMS301,EMS323,EMS253
 92 064626  101656  101222                  .WORD   EMS511,EMS501
 93 064632  077100  077424  000000          .WORD   EMS312,EMS325,0
 94 064640  076652  077136  076257  EMT52:  .WORD   EMS301,EMS313,EMS253
 95 064646  101656  101222  000000          .WORD   EMS511,EMS501,0
 96 064654  077472  071727  076415  EMT53:  .WORD   EMS331,EMS4,EMS256
 97 064662  101656  101222  000000          .WORD   EMS511,EMS501,0
 98 064670  076652  077404  076415  EMT54:  .WORD   EMS301,EMS324,EMS256
 99 064676  101656  101222                  .WORD   EMS511,EMS501
100 064702  077100  077435  000000          .WORD   EMS312,EMS326,0
101 064710  076652  077365  076415  EMT55:  .WORD   EMS301,EMS323,EMS256
102 064716  101656  101222                  .WORD   EMS511,EMS501
103 064722  077100  077424  000000          .WORD   EMS312,EMS325,0
104 064730  076652  077136  076415  EMT56:  .WORD   EMS301,EMS313,EMS256
105 064736  101656  101222  000000          .WORD   EMS511,EMS501,0
106 064744  076445  077522          EMT57:  .WORD   EMS257,EMS332
107 064750  101656  101452  101222          .WORD   EMS511,EMS506,EMS501,0
108 064760  071760  077540          EMT60:  .WORD   EMS5,EMS333
109 064764  101656  101513  101222          .WORD   EMS511,EMS507,EMS501,0
110 064774  076652  077404  076501  EMT61:  .WORD   EMS301,EMS324,EMS260
111 065002  101656  101222                  .WORD   EMS511,EMS501
112 065006  077100  077435  000000          .WORD   EMS312,EMS326,0
113 065014  076652  077365  076501  EMT62:  .WORD   EMS301,EMS323,EMS260
114 065022  101656  101222                  .WORD   EMS511,EMS501
```

```
    115 065026  077100  077424  000000           .WORD   EMS312,EMS325,0
    116 065034  076652  077136  076501  EMT63:   .WORD   EMS301,EMS313,EMS260
    117 065042  101656  101222  000000           .WORD   EMS511,EMS501,0
    118 065050  076634  072301          EMT64:   .WORD   EMS300,EMS12
    119 065054  101656  101222  000000           .WORD   EMS511,EMS501,0
    120 065062  072301  077564  077662  EMT65:   .WORD   EMS12,EMS335,EMS342
    121 065070  101656  101222  000000           .WORD   EMS511,EMS501,0
    122 065076  072301  077607  077662  EMT66:   .WORD   EMS12,EMS336,EMS342
    123 065104  101656  101222  000000           .WORD   EMS511,EMS501,0
    124 065112  076634  072030          EMT67:   .WORD   EMS300,EMS6
    125 065116  101656  101222                   .WORD   EMS511,EMS501
    126 065122  072074  077540  000000           .WORD   EMS7,EMS333,0
    127 065130  076634  072030  077445  EMT70:   .WORD   EMS300,EMS6,EMS327,EMS7
    128 065140  101656  101222  101344           .WORD   EMS511,EMS501,EMS504,0
    129 065150  072030  077564  077642  EMT71:   .WORD   EMS6,EMS335,EMS340,EMS10,EMS333,EMS342
    130 065164  101656  101222  101247           .WORD   EMS511,EMS501,EMS502,0
    131 065174  072030  077607  077642  EMT72:   .WORD   EMS6,EMS336,EMS340,EMS10,EMS334,EMS342
    132 065210  101656  101222  101247           .WORD   EMS511,EMS501,EMS502,0
    133 065220  076652  076501  076722  EMT73:   .WORD   EMS301,EMS260,EMS303,EMS11
    134 065230  101656  101222  101247           .WORD   EMS511,EMS501,EMS502,0
    135 065240  077714  077730  077662  EMT74:   .WORD   EMS343,EMS344,EMS342,0
    136 065250  076634  072357          EMT75:   .WORD   EMS300,EMS13
    137 065254  101656  101317  000000           .WORD   EMS511,EMS503,0
    138 065262  100005  072357  077757  EMT76:   .WORD   EMS346,EMS13,EMS345
    139 065270  101656  101317  000000           .WORD   EMS511,EMS503,0
    140 065276  077626  072357  077757  EMT77:   .WORD   EMS337,EMS13,EMS345
    141 065304  101656  101317  000000           .WORD   EMS511,EMS503,0
    142 065312  076652  077136  076312  EMT100:  .WORD   EMS301,EMS313,EMS254,EMS347,EMS13
    143 065324  101656  101317  000000           .WORD   EMS511,EMS503,0
    144 065332  100005  072426  077653  EMT101:  .WORD   EMS346,EMS14,EMS341,EMS15
    145 065342  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    146 065352  077626  072426  077653  EMT102:  .WORD   EMS337,EMS14,EMS341,EMS15
    147 065362  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    148 065372  076652  077136  076312  EMT103:  .WORD   EMS301,EMS313,EMS254,EMS347,EMS15
    149 065404  101656  101317                   .WORD   EMS511,EMS503
    150 065410  072426  077522  000000           .WORD   EMS14,EMS332,0
    151 065416  100005  072632  077653  EMT104:  .WORD   EMS346,EMS17,EMS341,EMS16
    152 065426  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    153 065436  077626  072632  077653  EMT105:  .WORD   EMS337,EMS17,EMS341,EMS16
    154 065446  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    155 065456  076652  077136  076312  EMT106:  .WORD   EMS301,EMS313,EMS254,EMS347,EMS16
    156 065470  101656  101317                   .WORD   EMS511,EMS503
    157 065474  072632  077522  000000           .WORD   EMS17,EMS332,0
    158 065502  100005  072673  077653  EMT107:  .WORD   EMS346,EMS20,EMS341,EMS21
    159 065512  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    160 065522  072673  100051  072737  EMT110:  .WORD   EMS20,EMS351,EMS21,EMS350,EMS22,EMS315
    161 065536  101656  101222  000000           .WORD   EMS511,EMS501,0
    162 065544  072673  077554  100044  EMT111:  .WORD   EMS20,EMS334,EMS350,EMS22,EMS333
    163 065556  101656  101222  000000           .WORD   EMS511,EMS501,0
    164 065564  077626  072673  077653  EMT112:  .WORD   EMS337,EMS20,EMS341,EMS21
    165 065574  101656  101317  101222           .WORD   EMS511,EMS503,EMS501,0
    166 065604  077626  072673  077653  EMT113:  .WORD   EMS337,EMS20,EMS341,EMS21,EMS350,EMS22,EMS334
    167 065622  101656  101222  000000           .WORD   EMS511,EMS501,0
    168 065630  072673  100067  072737  EMT114:  .WORD   EMS20,EMS352,EMS21,EMS350,EMS22,EMS333
    169 065644  101656  101222  000000           .WORD   EMS511,EMS501,0
    170 065652  076652  077136  076312  EMT115:  .WORD   EMS301,EMS313,EMS254,EMS347,EMS21
    171 065664  101656  101317                   .WORD   EMS511,EMS503
```

```
    172 065670  072673  077522  000000          .WORD   EMS20,EMS332,0
    173 065676  100005  073063  077653  EMT116: .WORD   EMS346,EMS23,EMS341,EMS24
    174 065706  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
    175 065716  077626  073063  077653  EMT117: .WORD   EMS337,EMS23,EMS341,EMS24
    176 065726  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
    177 065736  076652  077136  076312  EMT120: .WORD   EMS301,EMS313,EMS254,EMS347,EMS24
    178 065750  101656  101317                  .WORD   EMS511,EMS503
    179 065754  073063  077522  000000          .WORD   EMS23,EMS332,0
    180 065762  100005  073220  077653  EMT121: .WORD   EMS346,EMS25,EMS341,EMS26
    181 065772  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
    182 066002  077626  073220  077653  EMT122: .WORD   EMS337,EMS25,EMS341,EMS26
    183 066012  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
    184 066022  076652  077136  076312  EMT123: .WORD   EMS301,EMS313,EMS254,EMS347,EMS26
    185 066034  101656  101317                  .WORD   EMS511,EMS503
    186 066040  073220  077522  000000          .WORD   EMS25,EMS332,0
    187 066046  076634  073355  076756  EMT124: .WORD   EMS300,EMS27,EMS307,EMS2
    188 066056  101656  101317  000000          .WORD   EMS511,EMS503,0
    189 066064  077472  073355  100103  EMT125: .WORD   EMS331,EMS27,EMS353
    190 066072  101656  101317                  .WORD   EMS511,EMS503
    191 066076  073421  077216  000000          .WORD   EMS30,EMS315,0
    192 066104  077626  073355  077653  EMT126: .WORD   EMS337,EMS27,EMS341,EMS30
    193 066114  101656  101317  000000          .WORD   EMS511,EMS503,0
    194 066122  076652  077136  076312  EMT127:  WORD   EMS301,EMS313,EMS254,EMS347,EMS30
    195 066134  101656  101317                  .WORD   EMS511,EMS503
    196 066140  073355  077522  000000          .WORD   EMS27,EMS332,0
    197 066146  073501  100127  076122  EMT130: .WORD   EMS31,EMS354,EMS250
    198 066154  101656  101344  101317          .WORD   EMS511,EMS504,EMS503,0
    199 066164  073552  100127  076122  EMT131: .WORD   EMS32,EMS354,EMS250
    200 066172  101656  101344  101317          .WORD   EMS511,EMS504,EMS503,0
    201 066202  100162  073632  076122  EMT132: .WORD   EMS355,EMS33,EMS250,EMS341,EMS30
    202 066214  101656  101344  000000          .WORD   EMS511,EMS504,0
    203 066222  100162  073662  076122  EMT133: .WORD   EMS355,EMS34,EMS250,EMS341,EMS30
    204 066234  101656  101344  000000          .WORD   EMS511,EMS504,0
    205 066242  077472  071727  076353  EMT134: .WORD   EMS331,EMS4,EMS255
    206 066250  101656  101344  000000          .WORD   EMS511,EMS504,0
    207 066256  073711  100224  100246  EMT135: .WORD   EMS35,EMS357,EMS360,EMS15
    208 066266  101656  101222  000000          .WORD   EMS511,EMS501,0
    209 066274  076532  100322          EMT136: .WORD   EMS261,EMS362
    210 066300  101656  101317  000000          .WORD   EMS511,EMS503,0
    211 066306  076634  073753  076756  EMT137: .WORD   EMS300,EMS36,EMS307,EMS2
    212 066316  101656  101222  000000          .WORD   EMS511,EMS501,0
    213 066324  100162  074003  076353  EMT140: .WORD   EMS355,EMS37,EMS255,EMS341,EMS30
    214 066336  101656  101344  000000          .WORD   EMS511,EMS504,0
    215 066344  100005  074035  077757  EMT141: .WORD   EMS346,EMS40,EMS345
    216 066352  101656  101344  000000          .WORD   EMS511,EMS504,0
    217 066360  077626  074035  077653  EMT142: .WORD   EMS337,EMS40,EMS 1,EMS30
    218 066370  101656  101344  000000          .WORD   EMS511,EMS504,0
    219 066376  100343  077017  074113  EMT143: .WORD   EMS363,EMS310,EMS41
    220 066404  101656  101222  000000          .WORD   EMS511,EMS501,0
    221 066412  077626  074113  077653  EMT144: .WORD   EMS337,EMS41,EMS341,EMS252,EMS327,EMS253
    222 066426  101656  101222  000000          .WORD   EMS511,EMS501,0
    223 066434  074113  100127  100360  EMT145: .WORD   EMS41,EMS354,EMS364,EMS252,EMS365,EMS253
    224 066450  101656  101222  000000          .WORD   EMS511,EMS501,0
    225 066456  076652  076742  073753  EMT146: .WORD   EMS301,EMS306,EMS36
    226 066464  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
    227 066474  100406  074161          EMT147: .WORD   EMS366,EMS42
    228 066500  101656  101317  000000          .WORD   EMS511,EMS503,0
```

```
  229 066506  100431  100103  100401  EMT150: .WORD   EMS367,EMS353,EMS365,EMS42,EMS354,EMS3
  230 066522  101656  101317  000000          .WORD   EMS511,EMS503,0
  231 066530  077626  073753          EMT151: .WORD   EMS337,EMS36
  232 066534  101656  101222  000000          .WORD   EMS511,EMS501,0
  233 066542  074243  100127  073753  EMT152: .WORD   EMS43,EMS354,EMS36
  234 066550  101656  101222  000000          .WORD   EMS511,EMS501,0
  235 066556  100431  100103  100401  EMT153: .WORD   EMS367,EMS353,EMS365,EMS36,EMS370
  236 066570  101656  101317  000000          .WORD   EMS511,EMS503,0
  237 066576  100431  100103  100401  EMT154: .WORD   EMS367,EMS353,EMS365,EMS36,EMS371
  238 066610  101656  101317  000000          .WORD   EMS511,EMS503,0
  239 066616  076634  074314  076756  EMT155: .WORD   EMS300,EMS44,EMS307,EMS2
  240 066626  101656  101317  000000          .WORD   EMS511,EMS503,0
  241 066634  100431  100103  100401  EMT156: .WORD   EMS367,EMS353,EMS365,EMS44,EMS354,EMS3
  242 066650  101656  101317  000000          .WORD   EMS511,EMS503,0
  243 066656  076634  074355  076756  EMT157: .WORD   EMS300,EMS45,EMS307,EMS2
  244 066666  101656  101317  000000          .WORD   EMS511,EMS503,0
  245 066674  100431  100103  100401  EMT160: .WORD   EMS367,EMS353,EMS365,EMS45,EMS354,EMS3
  246 066710  101656  101317  101222          .WORD   EMS511,EMS503,EMS501
  247 066716  073711  077540  000000          .WORD   EMS35,EMS333,0
  248 066724  076634  074432  076756  EMT161: .WORD   EMS300,EMS46,EMS307,EMS2
  249 066734  101656  101317  000000          .WORD   EMS511,EMS503,0
  250 066742  077626  074432  100103  EMT162: .WORD   EMS337,EMS46,EMS353
  251 066750  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
  252 066760  073711  077564  077626  EMT163: .WORD   EMS35,EMS335,EMS337,EMS41,EMS334,EMS372
  253 066774  101656  101222  000000          .WORD   EMS511,EMS501,0
  254 067002  074514  077564  077626  EMT164: .WORD   EMS47,EMS335,EMS337,EMS41,EMS335,EMS372
  255 067016  101656  101222  000000          .WORD   EMS511,EMS501,0
  256 067024  077626  073711  077445  EMT165: .WORD   EMS337,EMS35,EMS327,EMS47
  257 067034  101656  101222                  .WORD   EMS511,EMS501
  258 067040  074113  077540  100545          .WORD   EMS41,EMS333,EMS372,0
  259 067050  076634  074514  076756  EMT166: .WORD   EMS300,EMS47,EMS307,EMS2
  260 067060  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
  261 067070  074554  077564  077642  EMT167: .WORD   EMS50,EMS335,EMS340,EMS36,EMS333
  262 067102  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
  263 067112  077626  073711          EMT170: .WORD   EMS337,EMS35
  264 067116  101656  101222  000000          .WORD   EMS511,EMS501,0
  265 067124  074554  073662  071662  EMT171: .WORD   EMS50,EMS34,EMS3
  266 067132  101656  101222  000000          .WORD   EMS511,EMS501,0
  267 067140  076652  076742  074623  EMT172: .WORD   EMS301,EMS306,EMS51
  268 067146  101656  101222  101344          .WORD   EMS511,EMS501,EMS504,0
  269 067156  100431  100103  100401  EMT173: .WORD   EMS367,EMS353,EMS365,EMS47,EMS354,EMS3
  270 067172  101656  101222  000000          .WORD   EMS511,EMS501,0
  271 067200  076634  076122  077445  EMT174: .WORD   EMS300,EMS250,EMS327,EMS255,EMS327,EMS256
  272 067214  077653  101755                  .WORD   EMS341,EMS600
  273 067220  101656  101222  000000          .WORD   EMS511,EMS501,0
  274 067226  076634  076415  077653  EMT175: .WORD   EMS300,EMS256,EMS341,EMS600
  275 067236  101656  101222  000000          .WORD   EMS511,EMS501,0
  276 067244  076634  076122  077653  EMT176: .WORD   EMS300,EMS250,EMS341,EMS600
  277 067254  101656  101344  000000          .WORD   EMS511,EMS504,0
  278 067262  076634  076353  077653  EMT177: .WORD   EMS300,EMS255,EMS341,EMS600
  279 067272  101656  101344  000000          .WORD   EMS511,EMS504,0
  280 067300  077626  074672  077653  EMT200: .WORD   EMS337,EMS52,EMS341,EMS601
  281 067310  101656  101222  000000          .WORD   EMS511,EMS501,0
  282 067316  100005  074672  077653  EMT201: .WORD   EMS346,EMS52,EMS341,EMS602
  283 067326  101656  101222  000000          .WORD   EMS511,EMS501,0
  284 067334  100005  074623  077653  EMT202: .WORD   EMS346,EMS51,EMS341,EMS602
  285 067344  101656  101222  000000          .WORD   EMS511,EMS501,0
```

```
286 067352  100005  074672  077757  EMT203: .WORD   EMS346,EMS52,EMS345,EMS373,EMS255
287 067364  101656  101344  101222          .WORD   EMS511,EMS504,EMS501,0
288 067374  100005  074672  077653  EMT204: .WORD   EMS346,EMS52,EMS341,EMS27
289 067404  101656  101344  101222          .WORD   EMS511,EMS504,EMS501,0
290 067414  074733  100127  071662  EMT205: .WORD   EMS53,EMS354,EMS3
291 067422  101656  101317  101247          .WORD   EMS511,EMS503,EMS502,EMS510,0
292 067434  077626  074774  100603  EMT206: .WORD   EMS337,EMS54,EMS374,EMS250,EMS327,EMS255
293 067450  077445  071662                  .WORD   EMS327,EMS3
294 067454  101656  101344  101222          .WORD   EMS511,EMS504,EMS501,0
295 067464  074774  100127  071662  EMT207: .WORD   EMS54,EMS354,EMS3
296 067472  101656  101222  000000          .WORD   EMS511,EMS501,0
297 067500  074774  100127  100360  EMT210: .WORD   EMS54,EMS354,EMS364,EMS250
298 067510  101656  101344  000000          .WORD   EMS511,EMS504,0
299 067516  074774  100127  100360  EMT211: .WORD   EMS54,EMS354,EMS364,EMS255
300 067526  101656  101344  000000          .WORD   EMS511,EMS504,0
301 067534  077626  075051          EMT212: .WORD   EMS337,EMS55
302 067540  101656  101344  000000          .WORD   EMS511,EMS504,0
303 067546  075127  077554  077757  EMT213: .WORD   EMS56,EMS334,EMS345,EMS373,EMS262,EMS327,EMS251
304 067564  101656  101222                  .WORD   EMS511,EMS501
305 067570  075127  077564  000000          .WORD   EMS56,EMS335,0
306 067576  077626  075127          EMT214: .WORD   EMS337,EMS56
307 067602  101656  101222  000000          .WORD   EMS511,EMS501,0
308 067610  075222  077540  100044  EMT215: .WORD   EMS57,EMS333,EMS350,EMS60,EMS334
309 067622  101656  101222  000000          .WORD   EMS511,EMS501,0
310 067630  100343  076742  071760  EMT216: .WORD   EMS363,EMS306,EMS5
311 067636  101656  101222  000000          .WORD   EMS511,EMS501,0
312 067644  100343  076742  075360  EMT217: .WORD   EMS363,EMS306,EMS61
313 067652  101656  101222  000000          .WORD   EMS511,EMS501,0
314 067660  075433  077540  100634  EMT220: .WORD   EMS62,EMS333,EMS375,EMS251
315 067670  101656  101222  000000          .WORD   EMS511,EMS501,0
316 067676  075433  077540  100650  EMT221: .WORD   EMS62,EMS333,EMS376,EMS262
317 067706  101656  101222  000000          .WORD   EMS511,EMS501,0
318 067714  075433  077540  100650  EMT222: .WORD   EMS62,EMS333,EMS376,EMS250
319 067724  101656  101222  000000          .WORD   EMS511,EMS501,0
320 067732  100005  075433  077653  EMT223: .WORD   EMS346,EMS62,EMS341,EMS603
321 067742  101656  101222  000000          .WORD   EMS511,EMS501,0
322 067750  100005  075514  100650  EMT224: .WORD   EMS346,EMS63,EMS376,EMS262
323 067760  101656  101222  101317          .WORD   EMS511,EMS501,EMS503,0
324 067770  075514  077540  100044  EMT225: .WORD   EMS63,EMS333,EMS350,EMS363,EMS310,EMS262
325 070004  101656  101222  000000          .WORD   EMS511,EMS501,0
326 070012  075514  100224  073753  EMT226: .WORD   EMS63,EMS357,EMS36,EMS372
327 070022  101656  101222  000000          .WORD   EMS511,EMS501,0
328 070030  075514  100204  100246  EMT227: .WORD   EMS63,EMS356,EMS360,EMS15
329 070040  101656  101222  000000          .WORD   EMS511,EMS501,0
330 070046  075514  100204  100274  EMT230: .WORD   EMS63,EMS356,EMS361,EMS15
331 070056  101656  101222  000000          .WORD   EMS511,EMS501,0
332 070064  075514  100204  074113  EMT231: .WORD   EMS63,EMS356,EMS41
333 070072  101656  101222  000000          .WORD   EMS511,EMS501,0
334 070100  074113  100664  077662  EMT232: .WORD   EMS41,EMS377,EMS342,EMS365,EMS63,EMS332
335 070114  101656  101222  000000          .WORD   EMS511,EMS501,0
336 070122  100431  100103  100401  EMT233: .WORD   EMS367,EMS353,EMS365,EMS63,EMS401
337 070134  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
338 070144  072551  100664  100545  EMT234: .WORD   EMS16,EMS377,EMS372,EMS365,EMS64,EMS354,EMS3
339 070162  101656  101317  101222          .WORD   EMS511,EMS503,EMS501,0
340 070172  076634  075624  076756  EMT235: .WORD   EMS300,EMS65,EMS307,EMS2
341 070202  101656  101247  101317          .WORD   EMS511,EMS502,EMS503,0
342 070212  075127  100664  100650  EMT236: .WORD   EMS56,EMS377,EMS376,EMS252,EMS372,EMS350
```

```
    343 070226  075624  100712                      .WORD   EMS65,EMS401
    344 070232  101656  101247  101317              .WORD   EMS511,EMS502,EMS503,EMS501,0
    345 070244  076634  075665  076756  EMT237:     .WORD   EMS300,EMS66,EMS307,EMS2
    346 070254  101656  101222  101317              .WORD   EMS511,EMS501,EMS503,0
    347 070264  072473  100675  100545  EMT240:     .WORD   EMS15,EMS400,EMS372,EMS350,EMS66,EMS401
    348 070300  101656  101317  101222              .WORD   EMS511,EMS503,EMS501,0
    349 070310  075665  077607  077642  EMT241:     .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS604
    350 070326  101656  101317  000000              .WORD   EMS511,EMS503,0
    351 070334  100736  102107  100727  EMT242:     .WORD   EMS403,EMS604,EMS402,EMS21,EMS377
    352 070346  101656  101317                      .WORD   EMS511,EMS503
    353 070352  073753  077564  000000              .WORD   EMS36,EMS335,0
    354 070360  075665  077607  077642  EMT243:     .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS604
    355 070376  101656  101317  000000              .WORD   EMS511,EMS503,0
    356 070404  075514  100712  101016  EMT244:     .WORD   EMS63,EMS401,EMS405,EMS604
    357 070414  101656  101317  000000              .WORD   EMS511,EMS503,0
    358 070422  073753  100501  101016  EMT245:     .WORD   EMS36,EMS370,EMS405,EMS604
    359 070432  101656  101317  000000              .WORD   EMS511,EMS503,0
    360 070440  100736  102107  100727  EMT246:     .WORD   EMS403,EMS604,EMS402,EMS24,EMS377
    361 070452  101656  101317                      .WORD   EMS511,EMS503
    362 070456  073753  077564  000000              .WORD   EMS36,EMS335,0
    363 070464  075742  077522  101016  EMT247:     .WORD   EMS67,EMS332,EMS405,EMS604
    364 070474  101656  101317  000000              .WORD   EMS511,EMS503,0
    365 070502  075665  077607  077642  EMT250:     .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS605
    366 070520  101656  101317  000000              .WORD   EMS511,EMS503,0
    367 070526  100736  102134  100727  EMT251:     .WORD   EMS403,EMS605,EMS402,EMS21,EMS377
    368 070540  101656  101317                      .WORD   EMS511,EMS503
    369 070544  073753  077564  000000              .WORD   EMS36,EMS335,0
    370 070552  075665  077607  077642  EMT252:     .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS605
    371 070570  101656  101317  000000              .WORD   EMS511,EMS503,0
    372 070576  075514  100712  101016  EMT253:     .WORD   EMS63,EMS401,EMS405,EMS605
    373 070606  101656  101317  000000              .WORD   EMS511,EMS503,0
    374 070614  073753  100501  101016  EMT254:     .WORD   EMS36,EMS370,EMS405,EMS605
    375 070624  101656  101317  000000              .WORD   EMS511,EMS503,0
    376 070632  100736  102134  100727  EMT255:     .WORD   EMS403,EMS605,EMS402,EMS24,EMS377
    377 070644  101656  101317                      .WORD   EMS511,EMS503
    378 070650  073753  077564  000000              .WORD   EMS36,EMS335,0
    379 070656  075742  077522  101016  EMT256:     .WORD   EMS67,EMS332,EMS405,EMS605
    380 070666  101656  101317  000000              .WORD   EMS511,EMS503,0
    381 070674  075665  077607  077642  EMT257:     .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS606
    382 070712  101656  101317  000000              .WORD   EMS511,EMS503,0
    383 070720  100736  102152  100727  EMT260:     .WORD   EMS403,EMS606,EMS402,EMS21,EMS377
    384 070732  101656  101317                      .WORD   EMS511,EMS503
    385 070736  073753  077564  000000              .WORD   EMS36,EMS335,0
    386 070744  075665  077607  077642  EMT261:     .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS606
    387 070762  101656  101317  000000              .WORD   EMS511,EMS503,0
    388 070770  075514  100712  101016  EMT262:     .WORD   EMS63,EMS401,EMS405,EMS606
    389 071000  101656  101317  000000              .WORD   EMS511,EMS503,0
    390 071006  073753  100501  101016  EMT263:     .WORD   EMS36,EMS370,EMS405,EMS606
    391 071016  101656  101317  000000              .WORD   EMS511,EMS503,0
    392 071024  100736  102152  100727  EMT264:     .WORD   EMS403,EMS606,EMS402,EMS24,EMS377
    393 071036  101656  101317                      .WORD   EMS511,EMS503
    394 071042  073753  077564  000000              .WORD   EMS36,EMS335,0
    395 071050  076054  100712  101016  EMT265:     .WORD   EMS70,EMS401,EMS405,EMS605
    396 071060  101656  101317  000000              .WORD   EMS511,EMS503,0
    397 071066  075742  077522  101016  EMT266:     .WORD   EMS67,EMS332,EMS405,EMS606
    398 071076  101656  101317  000000              .WORD   EMS511,EMS503,0
    399 071104  075665  100204  101041  EMT267:     .WORD   EMS66,EMS356,EMS407
```

```
    400 071112  101656  101317  000000          .WORD   EMS511,EMS503,0
    401 071120  075665  077607  077642  EMT270: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS607
    402 071136  101656  101317  000000          .WORD   EMS511,EMS503,0
    403 071144  100736  102172  100727  EMT271: .WORD   EMS403,EMS607,EMS402,EMS21,EMS377
    404 071156  101656  101317                  .WORD   EMS511,EMS503
    405 071162  073355  077607  000000          .WORD   EMS27,EMS336,0
    406 071170  073355  100501  101016  EMT272: .WORD   EMS27,EMS370,EMS405,EMS607
    407 071200  101656  101317  000000          .WORD   EMS511,EMS503,0
    408 071206  073355  100516  101016  EMT273: .WORD   EMS27,EMS371,EMS405,EMS607
    409 071216  101656  101317  000000          .WORD   EMS511,EMS503,0
    410 071224  073753  100777  101016  EMT274: .WORD   EMS36,EMS404,EMS405,EMS607
    411 071234  101656  101317  000000          .WORD   EMS511,EMS503,0
    412 071242  074733  100712  101016  EMT275: .WORD   EMS53,EMS401,EMS405,EMS607
    413 071252  101656  101317  101247          .WORD   EMS511,EMS503,EMS502,0
    414 071262  075742  077522  101016  EMT276: .WORD   EMS67,EMS332,EMS405,EMS607
    415 071272  101656  101317  000000          .WORD   EMS511,EMS503,0
    416 071300  075665  077607  077642  EMT277: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS607
    417 071316  101656  101317  000000          .WORD   EMS511,EMS503,0
    418 071324  100736  102172  100727  EMT300: .WORD   EMS403,EMS607,EMS402,EMS24,EMS377
    419 071336  101656  101317                  .WORD   EMS511,EMS503
    420 071342  073355  077607  000000          .WORD   EMS27,EMS336,0
    421 071350  076054  100712  101016  EMT301: .WORD   EMS70,EMS401,EMS405,EMS607
    422 071360  101656  101317  000000          .WORD   EMS511,EMS503,0
    423
```

```
      1 071366  102210  000000          EHT1:   .WORD   EH1,0
      2 071372  102266  000000          EHT2:   .WORD   EH2,0
      3 071376  102303  000000          EHT5:   .WORD   EH5,0
      4 071402  102341  000000          EHT7:   .WORD   EH7,0
      5 071406  102360  000000          EHT57:  .WORD   EH57,0
      6 071412  102456  000000          EHT65:  .WORD   EH65,0
      7 071416  102532  000000          EHT71:  .WORD   EH71,0
      8 071422  102274  000000          EHT74:  .WORD   EH3,0
      9 071426  102607  000000          EHT115: .WORD   EH115,0
     10 071432  102704  000000          EHT130: .WORD   EH130,0
     11 071436  103022  000000          EHT132: .WORD   EH132,0
     12 071442  103120  000000          EHT142: .WORD   EH142,0
     13 071446  103216  000000          EHT145: .WORD   EH145,0
     14 071452  103333  000000          EHT150: .WORD   EH150,0
     15 071456  103431  000000          EHT213: .WORD   EH213,0
     16 071462  103526  000000          EHT220: .WORD   EH220,0
     17
```

```
      1 071466  103564                EDT1:    .WORD   ED1
      2 071470  103574                EDT2:    .WORD   ED2
      3 071472  103600                EDT5:    .WORD   ED5
      4 071474  103606                EDT57:   .WORD   ED57
      5 071476  103620                EDT65:   .WORD   ED65
      6 071500  103630                EDT71:   .WORD   ED71
      7 071502  103574                EDT74:   .WORD   ED2
      8 071504  103640                EDT115:  .WORD   ED115
      9 071506  103652                EDT130:  .WORD   ED130
     10 071510  103640                EDT132:  .WORD   ED115
     11 071512  103666                EDT220:  .WORD   ED220
     12
```

```
      1 071514  103672                EFT1:   .WORD   EF1
      2 071516  103675                EFT2:   .WORD   EF2
      3 071520  103676                EFT5:   .WORD   EF5
      4 071522  103700                EFT57:  .WORD   EF57
      5 071524  103672                EFT65:  .WORD   EF1
      6 071526  103672                EFT71:  .WORD   EF1
      7 071530  103675                EFT74:  .WORD   EF2
      8 071532  103700                EFT115: .WORD   EF57
      9 071534  103704                EFT130: .WORD   EF130
     10 071536  103700                EFT132: .WORD   EF57
     11 071540  103676                EFT220: .WORD   EF5
     12
```

```
                                .SBTTL  ERROR MESSAGE STRINGS
                        
071542  116  117  116  EMS1:   .ASCIZ  @NONEXISTENT DEVICE 'NED' (RMCS2,BIT 12) @
071613  103  117  116  EMS2:   .ASCIZ  @CONTROLLER CLEAR 'CLR' (RMCS2,BIT 05) @
071662  106  125  116  EMS3:   .ASCIZ  @FUNCTION CODE (RMCS1, BITS 01 - 05) @
071727  125  116  125  EMS4:   .ASCIZ  @UNUSED BIT POSITIONS OF @
071760  104  105  126  EMS5:   .ASCIZ  @DEVICE AVAILABLE 'DVA' (RMCS1, BIT 11) @
072030  120  101  122  EMS6:   .ASCIZ  @PARTIY ERROR 'PAR' (RMER1, BIT 03) @
072074  104  101  124  EMS7:   .ASCIZ  @DATA PARITY ERROR 'DPE' (RMER2, BIT 03) @
072145  120  101  122  EMS10:  .ASCIZ  @PARITY TEST 'PAT' (RMCS2, BIT 04) @
072210  115  101  123  EMS11:  .ASCII  @MASSBUS CONTROL BUS PARITY ERROR 'MCPE' @
072260  050  122  115          .ASCIZ  @(RMCS1, BIT 13) @
072301  111  114  114  EMS12:  .ASCIZ  @ILLEGAL REGISTER ERROR 'ILR' (RMER1, BIT 01) @
072357  104  111  101  EMS13:  .ASCIZ  @DIAGNOSTIC MODE 'DMD' (RMMR1, BIT 00) @
072426  115  105  104  EMS14:  .ASCIZ  @MEDIUM ON LINE 'MOL' (RMDS, BIT 12) @
072473  115  101  111  EMS15:  .ASCIZ  @MAINTENANCE UNIT READY 'MUR' (RMMR1, BIT 09) @
072551  115  101  111  EMS16:  .ASCIZ  @MAINTENANCE WRITE PROTECT 'MWP' (RMMR1, BIT 03) @
072632  127  122  111  EMS17:  .ASCIZ  @WRITE LOCK 'WRL' (RMDS, BIT 11) @
072673  104  105  126  EMS20:  .ASCIZ  @DEVICE CHECK 'DVC' (RMER2, BIT 07) @
072737  115  101  111  EMS21:  .ASCIZ  @MAINTENANCE DRIVE FAULT 'MDF' (RMMR1, BIT 06) @
073016  125  116  123  EMS22:  .ASCIZ  @UNSAFE STATUS 'UNS' (RMER1, BIT 14) @
073063  123  105  105  EMS23:  .ASCIZ  @SEEK INCOMPLETE STATUS 'SKI' (RMER2, BIT 14) @
073141  115  101  111  EMS24:  .ASCIZ  @MAINTENANCE SEEK ERROR 'MSER' (RMMR1, BIT 07) @
073220  120  117  123  EMS25:  .ASCIZ  @POSITIONING IN PROGRESS 'PIP' (RMDS, BIT 13) @
073276  115  101  111  EMS26:  .ASCIZ  @MAINTENANCE ON CYLINDER 'MOC' (RMMR1, BIT 08) @
073355  105  116  104  EMS27:  .ASCIZ  @END OF BLOCK 'EBL' (RMMR1, BIT 13) @
073421  104  111  101  EMS30:  .ASCIZ  @DIAGNOSTIC END OF BLOCK 'DEBL' (RMMR1, BIT 13) @
073501  114  101  123  EMS31:  .ASCIZ  @LAST SECTOR STATUS 'LS' (RMMR1, BIT 02) @
073552  114  101  123  EMS32:  .ASCIZ  @LAST SECTOR/TRACK STATUS 'LST' (RMMR1, BIT 01) @
073632  123  105  103  EMS33:  .ASCIZ  @SECTOR ADDRESS BITS OF @
073662  124  122  101  EMS34:  .ASCIZ  @TRACK ADDRESS BITS OF @
073711  126  117  114  EMS35:  .ASCIZ  @VOLUME VALID 'VV' (RMDS, BIT 06) @
073753  107  117  040  EMS36:  .ASCIZ  @GO BIT (RMCS1, BIT 00) @
074003  103  131  114  EMS37:  .ASCIZ  @CYLINDER ADDRESS BITS OF @
074035  114  101  123  EMS40:  .ASCIZ  @LAST BLOCK TRANSFERRED, 'LBT' (RMDS, BIT 10) @
074113  103  117  115  EMS41:  .ASCIZ  @COMPOSITE ERROR 'ERR' (RMDS, BIT 14) @
074161  103  117  115  EMS42:  .ASCIZ  @COMMAND SEQUENCER TEST BIT 'TST' (RMMR2, BIT 12) @
074243  104  122  111  EMS43:  .ASCIZ  @DRIVE READY STATUS 'DRY' (RMDS, BIT 07) @
074314  103  117  116  EMS44:  .ASCIZ  @CONTINUE 'CONT' (RMMR1, BIT 06) @
074355  111  116  126  EMS45:  .ASCIZ  @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) @
074432  114  117  123  EMS46:  .ASCIZ  @LOSS OF SYSTEM CLOCK ERROR 'LSC' (RMER2, BIT 11) @
074514  117  103  103  EMS47:  .ASCIZ  @OCCUPIED 'OCC' (RMMR1, BIT 15) @
074554  111  114  114  EMS50:  .ASCIZ  @ILLEGAL FUNCTION 'ILF' (RMER1, BIT 0) @
074623  117  106  106  EMS51:  .ASCIZ  @OFFSET DIRECTION 'OFD' (RMOF, BIT 07) @
074672  117  106  106  EMS52:  .ASCIZ  @OFFSET MODE 'OM' (RMDS, BIT 00) @
074733  122  125  116  EMS53:  .ASCIZ  @RUN AND GO 'RG' (RMMR1, BIT 14) @
074774  111  116  126  EMS54:  .ASCIZ  @INVALID ADDRESS ERROR 'IAE' (RMER1, BIT 10) @
075051  101  104  104  EMS55:  .ASCIZ  @ADDRESS OVERFLOW ERROR 'AOE' (RMER1, BIT 09) @
075127  122  105  107  EMS56:  .ASCII  @REGISTER MODIFICATION REFUSED ERROR @
075173  042  122  115          .ASCIZ  @'RMR' (RMER1, BIT 02) @
075222  104  122  111  EMS57:  .ASCIZ  @DRIVE REQUEST REQUIRED STATUS 'DRQ' (RMDT, BIT 11) @
075306  120  122  117  EMS60:  .ASCIZ  @PROGRAMMABLE STATUS 'PGM' (RMDS, BIT 09) @
075360  104  122  111  EMS61:  .ASCIZ  @DRIVE PRESENT STATUS 'DPR' (RMDS, BIT 08) @
075433  120  117  122  EMS62:  .ASCIZ  @PORT REQUEST FLOP 'RQA,RQB' (RMMR2, BITS 15,14) @
075514  101  124  124  EMS63:  .ASCIZ  @ATTENTION 'ATA' (RMDS, BIT 15) @
075554  127  122  111  EMS64:  .ASCIZ  @WRITE LOCK ERROR 'WLE' (RMER1, BIT 11) @
075624  105  130  103  EMS65:  .ASCIZ  @EXCEPTION 'REX' (RMMR1, BIT 12) @
```

```
075665   111   116   126   EMS66:   .ASCIZ   @INVALID COMMAND ERROR ''IVC'' (RMER2, BIT 12) @
075742   124   101   107   EMS67:   .ASCIZ   @TAG BUS (RMMR2, BITS 00-09) OR TAG CONTROL @
076016   114   111   116            .ASCIZ   @LINES (RMMR2, BITS 10,11,13) @
076054   123   105   101   EMS70:   .ASCIZ   @SEARCH ENABLE 'ESRC'' (RMMR1, BIT 11) @

076122   104   111   123   EMS250:  .ASCIZ   @DISK ADDRESS REGISTER (RMDA) @
076160   103   117   116   EMS251:  .ASCIZ   @CONTROL STATUS REGISTER #1 (RMCS1) @
076224   105   122   122   EMS252:  .ASCIZ   @ERROR REGISTER #1 (RMER1) @
076257   105   122   122   EMS253:  .ASCIZ   @ERROR REGISTER #2 (RMER2) @
076312   115   101   111   EMS254:  .ASCIZ   @MAINTENANCE REGISTER #1 (RMMR1) @
076353   104   105   123   EMS255:  .ASCIZ   @DESIRED CYLINDER REGISTER (RMDC) @
076415   117   106   106   EMS256:  .ASCIZ   @OFFSET REGISTER (RMOF) @
076445   104   122   111   EMS257:  .ASCIZ   @DRIVE TYPE REGISTER (RMDT) @
076[illegible]1   110   117   114   EMS260:  .ASCIZ   @HOLDING REGISTER (RMHR) @
076[illegible]2   123   105   122   EMS261:  .ASCIZ   @SERIAL NUMBER REGISTER (RMSN) @
0[illegible]5[illegible]1   101   124   124   EMS262:  .ASCIZ   @ATTENTION SUMMARY REGISTER (RMAS) @

076634   103   101   116   EMS300:  .ASCIZ   @CANNOT CLEAR @
076652   103   101   116   EMS301:  .ASCIZ   @CANNOT WRITE/READ @
076675   101   116   131   EMS302:  .ASCIZ   @ANY DEVICE REGISTER @
076722   127   111   124   EMS303:  .ASCIZ   @WITHOUT @
076733   105   122   122   EMS304:  .ASCIZ   @ERROR @
076742   101   040   117   EMS306:  .ASCIZ   @A ONE FROM @
076756   125   123   111   EMS307:  .ASCIZ   @USING MASSBUS INITIALIZE, I.E., @
077017   101   040   132   EMS310:  .ASCIZ   @A ZERO FROM @
077034   105   126   105   EMS311:  .ASCIZ   @EVERY DEVICE REGISTER BIT POSITION @
077100   124   110   105   EMS312:  .ASCIZ   @THE FOLLOWING BITS ARE STUCK @
077136   101   040   123   EMS313:  .ASCIZ   @A SHIFTING ONE BIT FROM @
077167   101   120   120   EMS314:  .ASCIZ   @APPEARS STUCK AT ZERO @
077216   101   120   120   EMS315:  .ASCIZ   @APPEARS STUCK AT ONE @
077244   122   105   107   EMS316:  .ASCIZ   @REGISTER SELECT @
077265   061   040   050   EMS317:  .ASCIZ   @1 (1,2,4,8,16) @
077305   062   040   050   EMS320:  .ASCIZ   @2 (1,2,4,8,16) @
077325   064   040   050   EMS321:  .ASCIZ   @4 (1,2,4,8,16) @
077345   070   040   050   EMS322:  .ASCIZ   @8 (1,2,4,8,16) @
077365   101   114   114   EMS323:  .ASCIZ   @ALL ONES FROM @
077404   101   114   114   EMS324:  .ASCIZ   @ALL ZEROS FROM @
077424   101   124   040   EMS325:  .ASCIZ   @AT ZERO @
077435   101   124   040   EMS326:  .ASCIZ   @AT ONE @
077445   054   040   117   EMS327:  .ASCIZ   @, OR @
077453   015   012   103   EMS330:  .ASCIZ   <CR><LF>@CS MBA CLRL @
077472   103   101   116   EMS331:  .ASCIZ   @CANNOT READ ZEROS FROM @
077522   111   123   040   EMS332:  .ASCIZ   @IS INCORRECT @
077540   111   123   040   EMS333:  .ASCIZ   @IS NOT SET @
077554   111   123   040   EMS334:  .ASCIZ   @IS SET @
077564   123   110   117   EMS335:  .ASCIZ   @SHOULD NOT BE SET @
077607   123   110   117   EMS336:  .ASCIZ   @SHOULD BE SET @
077626   103   101   116   EMS337:  .ASCIZ   @CANNOT SET @
077642   102   105   103   EMS340:  .ASCIZ   @BECAUSE @
077653   125   123   111   EMS341:  .ASCIZ   @USING @
077662   104   125   122   EMS342:  .ASCIZ   @DURING REGISTER TRANSFER @
077714   125   116   105   EMS343:  .ASCIZ   @UNEXPECTED @
077730   102   125   123   EMS344:  .ASCIZ   @BUS TIMEOUT (04 TRAP) @
077757   102   131   040   EMS345:  .ASCIZ   @BY REGISTER TRANSFER @
100005   103   101   116   EMS346:  .ASCIZ   @CANNOT RESET @
100023   127   111   124   EMS347:  .ASCIZ   @WITHOUT SETTING @
100044   102   125   124   EMS350:  .ASCIZ   @BUT @
```

```
    115 100051      127     101     123  EMS351: .ASCIZ  @WAS RESET BY @
    116 100067      127     101     123  EMS352: .ASCIZ  @WAS SET BY @
    117 100103      111     116     040  EMS353: .ASCIZ  @IN DIAGNOSTIC MODE @
    118 100127      111     123     040  EMS354: .ASCIZ  @IS INCORRECT ACCORDING TO @
    119 100162      103     101     116  EMS355: .ASCIZ  @CANNOT INCREMENT @
    120 100204      127     101     123  EMS356: .ASCIZ  @WAS NOT SET BY @
    121 100224      127     101     123  EMS357: .ASCIZ  @WAS NOT RESET BY @
    122 100246      060     040     124  EMS360: .ASCIZ  @0 TO 1 TRANSITION OF @
    123 100274      061     040     124  EMS361: .ASCIZ  @1 TO 0 TRANSITION OF @
    124 100322      111     123     040  EMS362: .ASCIZ  @IS INCONSISTENT @
    125 100343      103     101     116  EMS363: .ASCIZ  @CANNOT READ @
    126 100360      124     105     123  EMS364: .ASCIZ  @TEST PATTERN IN @
    127 100401      101     116     104  EMS365: .ASCIZ  @AND @
    128 100406      103     101     116  EMS366: .ASCIZ  @CANNOT INITIALIZE @
    129 100431      124     110     105  EMS367: .ASCIZ  @THE COMMAND SEQUENCER HAS BEEN CLOCKED @
    130 100501      122     105     123  EMS370: .ASCIZ  @RESET EARLY @
    131 100516      104     111     104  EMS371: .ASCIZ  @DID NOT RESET ON TIME @
    132 100545      104     125     122  EMS372: .ASCIZ  @DURING COMMAND EXECUTION @
    133 100577      124     117     040  EMS373: .ASCIZ  @TO @
    134 100603      127     111     124  EMS374: .ASCIZ  @WITH ANY COMBINATION OF @
    135 100634      102     131     040  EMS375: .ASCIZ  @BY READING @
    136 100650      102     131     040  EMS376: .ASCIZ  @BY WRITING @
    137 100664      127     101     123  EMS377: .ASCIZ  @WAS SET @
    138 100675      127     101     123  EMS400: .ASCIZ  @WAS NOT SET @
    139 100712      104     111     104  EMS401: .ASCIZ  @DID NOT SET @
    140 100727      127     110     111  EMS402: .ASCIZ  @WHILE @
    141 100736      103     117     115  EMS403: .ASCIZ  @COMMAND SEQUENCER DID NOT ABORT @
    142 100777      127     101     123  EMS404: .ASCIZ  @WAS NOT RESET @
    143 101016      104     125     122  EMS405: .ASCIZ  @DURING @
    144 101026      127     101     123  EMS406: .ASCIZ  @WAS RESET @
    145 101041      123     105     101  EMS407: .ASCIZ  @SEARCH TIMEOUT @
    146
    147 101061      011     104     105  EMS500: .ASCII  @       DEVICE IS NON-EXISTENT,@<CR><LF>
    148 101113      011     104     105          .ASCII  @       DEVICE IS SWITCHED TO OTHER PORT@<CR><LF>
    149 101156      011     124     122          .ASCIZ  @       TRANSCEIVER ENABLE SWITCH IS OFF@<CR><LF>
    150 101222      011     111     106  EMS501: .ASCIZ  @       IF MODULE, M7686,@<CR><LF>
    151 101247      011     115     101  EMS502: .ASCIZ  @       MASSBUS TRANSCEIVER, M5922 OR M5923     @<CR><LF>
    152 101317      011     103     123  EMS503: .ASCIZ  @       CS MODULE, M7684,@<CR><LF>
    153 101344      011     104     123  EMS504: .ASCIZ  @       DS MODULE, M8685/M7685,@<CR><LF>
    154 101377      011     104     105  EMS505: .ASCIZ  @       DEVICE IS SWITCHED TO A/B PORT POSITION@<CR><LF>
    155 101452      011     104     105  EMS506: .ASCIZ  @       DEVICE IS NOT AN RM05/3/2, OR@<CR><LF>
    156 101513      011     104     105  EMS507: .ASCIZ  @       DEVICE IS SWITCHED TO PROGRAMMABLE PORT POSITION, OR@<CR><LF>
    157 101603      011     101     123  EMS510: .ASCIZ  @       ASSUMING THE RH CONTROLLER HAS NO FAULT@<CR><LF>
    158 101656      015     012     011  EMS511: .ASCII          <CR><LF>@       PROBABLE FAULT(S):@<CR><LF>
    159 101705      011     050     116          .ASCIZ  @       (NOT INCLUDING CABLES OR CONNECTORS)@<CR><LF>
    160
    161 101755      122     105     101  EMS600: .ASCIZ  @READ IN PRESET COMMAND @
    162 102005      117     106     106  EMS601: .ASCIZ  @OFFSET COMMAND @
    163 102025      122     105     124  EMS602: .ASCIZ  @RETURN TO CENTER CENTER COMMAND @
    164 102066      122     105     114  EMS603: .ASCIZ  @RELEASE COMMAND @
    165 102107      122     105     103  EMS604: .ASCIZ  @RECALIBRATE COMMAND @
    166 102134      123     105     105  EMS605: .ASCIZ  @SEEK COMMAND @
    167 102152      123     105     101  EMS606: .ASCIZ  @SEARCH COMMAND @
    168 102172      104     101     124  EMS607: .ASCIZ  @DATA COMMAND @
    169
```

```
 1 102210      105      130      120  EH1:    .ASCII  @EXPCTD  RECEVD  REGSTR@<CRLF>
 2 102237      123      124      101           .ASCIZ  @STATUS  STATUS  ADRESS@
 3 102266      040      102      101  EH2:    .ASCII  @ BASE@<CRLF>
 4 102274      101      104      122  EH3:    .ASCIZ  @ADRESS@
 5 102303      105      130      120  EH5:    .ASCII  @EXPCTD   STUCK@<CRLF>
 6 102322      122      105      123           .ASCIZ  @RESULT  BIT(S)@
 7 102341      105      130      120  EH7:    .ASCIZ  @EXPCTD  RECEVD@
 8 102360      123      116      107  EH57:   .ASCII  @SNGPRT  DULPRT  RECEVD  DRVTYP@<CRLF>
 9 102417      104      122      126           .ASCIZ  @DRVTYP  DRVTYP  DRVTYP  REGADR@
10 102456      105      130      120  EH65:   .ASCII  @EXPCTD  RECEVD  TEST@<CRLF>
11 102503      123      124      101           .ASCIZ  @STATUS  STATUS  REGSTR@
12 102532      105      130      120  EH71:   .ASCII  @EXPCTD  RECEVD   TEST@<CRLF>
13 102560      123      124      101           .ASCIZ  @STATUS  STATUS  PATTRN@
14 102607      105      130      120  EH115:  .ASCII  @EXPCTD  RECEVD  REGSTR   TEST@<CRLF>
15 102645      123      124      101           .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN@
16 102704      105      130      120  EH130:  .ASCII  @EXPCTD  RECEVD  REGSTR   TEST   OFFSET@<CRLF>
17 102753      123      124      101           .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN  REGSTR@
18 103022      105      130      120  EH132:  .ASCII  @EXPCTD  ACTUAL  REGSTR  OFFSET@<CRLF>
19 103061      103      117      125           .ASCIZ  @COUNT   COUNT   ADRESS  REGSTR@
20 103120      105      130      120  EH142:  .ASCII  @EXPCTD  RECEVD  REGSTR  OFFSET@<CRLF>
21 103157      123      124      101           .ASCIZ  @STATUS  STATUS  ADRESS  REGSTR@
22 103216      105      130      120  EH145:  .ASCII  @EXPCTD  ACTUAL  REGSTR  RMER1   RMER2@<CRLF>
23 103264      103      115      120           .ASCIZ  @CMPERR  CMPERR  ADRESS  PATTRN  PATTRN@
24
25 103333      105      130      120  EH150:  .ASCII  @EXPCTD  ACTUAL  REGSTR FUNCTION@<CRLF>
26 103373      122      105      123           .ASCIZ  @RESULT  RESULT  ADRESS   CODE@
27 103431      105      130      120  EH213:  .ASCII  @EXPCTD  ACTUAL  STATUS   TEST@<CRLF>
28 103467      122      105      123           .ASCIZ  @RESULT  RESULT  ADRESS  REGSTR@
29 103526      101      103      124  EH220:  .ASCII  @ACTUAL  REGSTR@<CRLF>
30 103545      122      105      123           .ASCIZ  @RESULT  ADRESS@
31                                    .EVEN
32
```

```
     1 103564  001140  001142  001136  ED1:    .WORD   $GDDAT,$BDDAT,$BDADR,0
     2 103574  001136  000000          ED2:    .WORD   $BDADR,0
     3 103600  001140  001142  000000  ED5:    .WORD   $GDDAT,$BDDAT,0
     4 103606  001174  001176  001142  ED57:   .WORD   $TMP0,$TMP1,$BDDAT,$BDADR,0
     5 103620  001140  001142  001174  ED65:   .WORD   $GDDAT,$BDDAT,$TMP0,0
     6 103630  001140  001142  001444  ED71:   .WORD   $GDDAT,$BDDAT,RMHR0,0
     7 103640  001140  001142  001136  ED115:  .WORD   $GDDAT,$BDDAT,$BDADR,$TMP0,0
     8 103652  001140  001142  001136  ED130:  .WORD   $GDDAT,$BDDAT,$BDADR,$TMP0,$TMP1,0
     9 103666  001142  001136          ED220:  .WORD   $BDDAT,$BDADR
    10
    11 103672     000     000     000  EF1:    .BYTE   0,0,0
    12 103675     000                  EF2:    .BYTE   0
    13 103676     000     000          EF5:    .BYTE   0,0
    14 103700     000     000     000  EF57:   .BYTE   0,0,0,0
    15 103704     000     000     000  EF130:  .BYTE   0,0,0,0,0
    16                                  .EVEN
    17
```

```
                                BUFFER:
                                BUFONE: .BLKW   258.
                                BUFTWO: .BLKW   258.

                        103712  .=BUFFER

                                HELP:
103712      200                 .ASCII  <CRLF>
103713      200                 .ASCII  <CRLF>
103714      114     111     123 .ASCII  @LIST OF TESTS@<CRLF>
103732      055     055     055 .ASCII  @-------------@<CRLF>
103750      124     061     011 .ASCII  @T1      TRANSFER TEST@<CRLF>
103771      124     062     011 .ASCII  @T2      CTOD TEST@<CRLF>
104006      124     063     011 .ASCII  @T3      MASSBUS INITIALIZE TEST@<CRLF>
104041      124     064     011 .ASCII  @T4      CLEAR STUCK ACTIVE TEST@<CRLF>
104074      124     065     011 .ASCII  @T5      TRISTATE TRANSFER TEST@<CRLF>
104126      124     066     011 .ASCII  @T6      REGISTER SELECT TEST@<CRLF>
104156      124     067     011 .ASCII  @T7      DRIVE TYPE TEST@<CRLF>
104201      124     061     060 .ASCII  @T10     DEVICE AVAILABLE TEST@<CRLF>
104233      124     061     061 .ASCII  @T11     HOLDING REGISTER TRANSFER TEST@<CRLF>
104276      124     061     062 .ASCII  @T12     CONTROL STATUS #1 TRANSFER TEST@<CRLF>
104342      124     061     063 .ASCII  @T13     ERROR REGISTER #1 TRANSFER TEST@<CRLF>
104406      124     061     064 .ASCII  @T14     CLEAR OFFSET STUCK ACTIVE TEST@<CRLF>
104451      124     061     065 .ASCII  @T15     OFFSET REGISTER TRANSFER TEST@<CRLF>
104513      124     061     066 .ASCII  @T16     ERROR REGISTER #2 TRANSFER TEST@<CRLF>
104557      124     061     067 .ASCII  @T17     SERIAL NUMBER TEST@<CRLF>
104606      124     062     060 .ASCII  @T20     CONTROL BUS PARITY DETECTION TEST@<CRLF>
104654      124     062     061 .ASCII  @T21     CONTROL BUS PARITY GENERATION TEST@<CRLF>
104723      124     062     062 .ASCII  @T22     RMDA,RMDC FAULT TEST@<CRLF>
104754      124     062     063 .ASCII  @T23     DISK ADDRESS TRANSFER TEST@<CRLF>
105013      124     062     064 .ASCII  @T24     DESIRED CYLINDER TRANSFER TEST@<CRLF>
105056      124     062     065 .ASCII  @T25     ILLEGAL REGISTER TEST@<CRLF>
105110      124     062     066 .ASCII  @T26     RESET GO BY INIT TEST@<CRLF>
105142      124     062     067 .ASCII  @T27     DIAGNOSTIC MODE TEST@<CRLF>
105173      124     063     060 .ASCII  @T30     MOL TEST@<CRLF>
105210      124     063     061 .ASCII  @T31     WRITE LOCK TEST@<CRLF>
105234      124     063     062 .ASCII  @T32     DRIVE FAULT TEST@<CRLF>
105261      124     063     063 .ASCII  @T33     SEEK ERROR TEST@<CRLF>
105305      124     063     064 .ASCII  @T34     PIP TEST@<CRLF>
105322      124     063     065 .ASCII  @T35     EBL TEST@<CRLF>
105337      124     063     066 .ASCII  @T36     LAST SECTOR, LAST TRACK TEST@<CRLF>
105400      124     063     067 .ASCII  @T37     RMDA COUNT TEST@<CRLF>
105424      124     064     060 .ASCII  @T40     RMDC COUNT TEST@<CRLF>
105450      124     064     061 .ASCII  @T41     LBT TEST@<CRLF>
105465      124     064     062 .ASCII  @T42     COMPOSITE ERROR TEST@<CRLF>
105516      124     064     063 .ASCII  @T43     WRITE GO TEST@<CRLF>
105540      124     064     064 .ASCII  @T44     BRANCH MULTIPLEXOR TEST@<CRLF>
105574      124     064     065 .ASCII  @T45     SET/RESET GO TEST@<CRLF>
105622      124     064     066 .ASCII  @T46     END 1 RESET GO TEST@<CRLF>
105652      124     064     067 .ASCII  @T47     SET PULSE TEST@<CRLF>
105675      124     065     060 .ASCII  @T50     SET/RESET IVC TEST@<CRLF>
105724      124     065     061 .ASCII  @T51     SET LSC TEST@<CRLF>
105745      124     065     062 .ASCII  @T52     DECODE TEST@<CRLF>
105765      124     065     063 .ASCII  @T53     SET/RESET VOLUME VALID TEST@<CRLF>
106025      124     065     064 .ASCII  @T54     ILLEGAL FUNCTION TEST@<CRLF>
106057      124     065     065 .ASCII  @T55     OCCUPIED TEST@<CRLF>
106101      124     065     066 .ASCII  @T56     READ IN PRESET TEST@<CRLF>
```

```
 58 106131      124     065     067  .ASCII  @T57    RIP/RMOF TEST@<CRLF>
 59 106153      124     066     060  .ASCII  @T60    RMDA/RMDC/RIP TEST@<CRLF>
 60 106202      124     066     061  .ASCII  @T61    OFFSET COMMAND TEST@<CRLF>
 61 106232      124     066     062  .ASCII  @T62    RETURN TO CENTER TEST@<CRLF>
 62 106264      124     066     063  .ASCII  @T63    RMDC CLEAR OFFSET TEST@<CRLF>
 63 106317      124     066     064  .ASCII  @T64    EBL CLEAR OFFSET TEST@<CRLF>
 64 106351      124     066     065  .ASCII  @T65    RUN AND GO TEST@<CRLF>
 65 106375      124     066     066  .ASCII  @T66    SET IAE TEST@<CRLF>
 66 106416      124     066     067  .ASCII  @T67    SEARCH, SEEK, READ, WRITE TEST@<CRLF>
    106461      124     067     060  .ASCII  @T70    INVALID TRACK/SECTOR TEST@<CRLF>
 68 106517      124     067     061  .ASCII  @T71    INVALID CYLINDER TEST@<CRLF>
 69 106551      124     067     062  .ASCII  @T72    SET AOE TEST@<CRLF>
 70 106572      124     067     063  .ASCII  @T73    SET RMR TEST@<CRLF>
 71 106613      124     067     064  .ASCII  @T74    PGM STATUS CHECK@<CRLF>
 72 106640      124     067     065  .ASCII  @T75    DVA/DPR STATUS CHECK@<CRLF>
 73 106671      124     067     066  .ASCII  @T76    PORT REQUEST TEST, PART 1@<CRLF>
 74 106727      124     067     067  .ASCII  @T77    PORT REQUEST TEST, PART 2@<CRLF>
 75 106765      124     061     060  .ASCII  @T100   PORT REQUEST TEST, PART 3@<CRLF>
 76 107024      124     061     060  .ASCII  @T101   RELEASE TEST@<CRLF>
 77 107046      124     061     060  .ASCII  @T102   WRITE ATA TEST@<CRLF>
 78 107072      124     061     060  .ASCII  @T103   RESET ATA BY GO TEST@<CRLF>
 79 107124      124     061     060  .ASCII  @T104   UNIT READY ATA TEST@<CRLF>
 80 107155      124     061     060  .ASCII  @T105   ERROR ATA TEST@<CRLF>
 81 107201      124     061     060  .ASCII  @T106   REGISTER TRANSFER ATA TEST@<CRLF>
 82 107241      124     061     060  .ASCII  @T107   P SET ATA TEST@<CRLF>
 83 107265      124     061     061  .ASCII  @T110   SET WLE TEST@<CRLF>
 84 107307      124     061     061  .ASCII  @T111   EXCEPTION TEST@<CRLF>
 85 107333      124     061     061  .ASCII  @T112   RECALIBRATE TEST@<CRLF>
 86 107361      124     061     061  .ASCII  @T113   SEEK TEST@<CRLF>
 87 107400      124     061     061  .ASCII  @T114   SEARCH TEST@<CRLF>
 88 107421      124     061     061  .ASCII  @T115   SEARCH TIMEOUT TEST@<CRLF>
 89 107452      124     061     061  .ASCII  @T116   DATA COMMAND TESTS (1)@<CRLF>
 90 107506      124     061     061  .ASCII  @T117   DATA COMMAND TESTS (2)@<CRLF>
 91 107542      124     061     062  .ASCII  @T120   DATA COMMAND TESTS (3)@<CRLF>
 92 107576      200                  .ASCII  <CRLF>
 93 107577      117     120     105  .ASCII  @OPERATIONAL SWITCH SETTINGS@<CRLF>
 94 107633      055     055     055  .ASCII  @-------------------------@<CRLF>
 95 107667      123     127     111  .ASCII  @SWITCH                  USE@<CRLF>
 96 107704      055     055     055  .ASCII  @------          --------------------@<CRLF>
 97 107741      040     040     061  .ASCII  @  15            HALT ON ERROR@<CRLF>
 98 107765      040     040     061  .ASCII  @  14            LOOP ON TEST@<CRLF>
 99 110010      040     040     061  .ASCII  @  13            INHIBIT ERROR TYPEOUTS@<CRLF>
100 110045      040     040     061  .ASCII  @  12            @<CRLF>
101 110054      040     040     061  .ASCII  @  11            INHIBIT ITERATIONS@<CRLF>
102 110105      040     040     061  .ASCII  @  10            BELL ON ERROR@<CRLF>
103 110131      040     040     040  .ASCII  @   9            LOOP ON ERROR@<CRLF>
104 110155      040     040     040  .ASCII  @   8            LOOP ON TEST IN SWR<7:0>@<CRLF>
105 110214      040     040     040  .ASCII  @   7            TN128@<CRLF>
106 110230      040     040     040  .ASCII  @   6            TN64@<CRLF>
107 110243      040     040     040  .ASCII  @   5            TN32@<CRLF>
108 110256      040     040     040  .ASCII  @   4            TN16@<CRLF>
109 110271      040     040     040  .ASCII  @   3            TN8@<CRLF>
110 110303      040     040     040  .ASCII  @   2            TN4@<CRLF>
111 110315      040     040     040  .ASCII  @   1            TN2@<CRLF>
112 110327      040     040     040  .ASCIZ  @   0            TN1@<CRLF>
113
114          000200                  .END    200
```

SYMBOL TABLE

```
ABASE = 176700          AVECT1= 120254          CNSL01  062267          ED220   103666          EMS12   072301
ACDW1 = 000000          AVECT2= 000000          CNSL02  062305          ED5     103600          EMS13   072357
ACDW2 = 000000          A16   = 000400          CNSL03  062347          ED57    103606          EMS14   072426
ACPUOP= 000000          A17   = 001000          CNSL04  062367          ED65    103620          EMS15   072473
ADDW0 = 000000          BADTMO  004546          CNSL05  062423          ED71    103630          EMS16   072551
ADDW1 = 000000          BAI   = 000010          CNSL06  062435          EECC  = 000020          EMS17   072632
ADDW10= 000000          BB00  = 000001          CNSL07  062466          EFT1    071514          EMS2    071613
ADDW11= 000000          BB01  = 000002          CNSL08  062631          EFT115  071532          EMS20   072673
ADDW12= 000000          BB02  = 000004          CNSL09  062632          EFT130  071534          EMS21   072737
ADDW13= 000000          BB03  = 000010          CNTCLR  055614          EFT132  071536          EMS22   073016
ADDW14= 000000          BB04  = 000020          COMMA   062145          EFT2    071516          EMS23   073063
ADDW15= 000000          BB05  = 000040          CONT  = 000100          EFT220  071540          EMS24   073141
ADDW2 = 000000          BB06  = 000100          CR    = 000015          EFT5    071520          EMS25   073220
ADDW3 = 000000          BB07  = 000200          CRLF  = 000200          EFT57   071522          EMS250  076122
ADDW4 = 000000          BB08  = 000400          CYLMSK= 001777          EFT65   071524          EMS251  076160
ADDW5 = 000000          BB09  = 001000          DBCK  = 100000          EFT71   071526          EMS252  076224
ADDW6 = 000000          BIT0  = 000001          DBEN  = 040000          EFT74   071530          EMS253  076257
ADDW7 = 000000          BIT00 = 000001          DBL   = 002000          EF1     103672          EMS254  076312
ADDW8 = 000000          BIT01 = 000002          DCK   = 100000          EF130   103704          EMS255  076353
ADDW9 = 000000          BIT02 = 000004          DDISP = 177570          EF2     103675          EMS256  076415
ADEVCT= 000000          BIT03 = 000010          DEBL  = 020000          EF5     103676          EMS257  076445
ADEVM = 000000          BIT04 = 000020          DISPLA  001156          EF57    103700          EMS26   073276
ADR   = 000001          BIT05 = 000040          DISPRE  000174          EHT1    071366          EMS260  076501
AENV  = 000000          BIT06 = 000100          DLT   = 100000          EHT115  071426          EMS261  076532
AENVM = 000000          BIT07 = 000200          DMD   = 000001          EHT130  071432          EMS262  076571
AFATAL= 000000          BIT08 = 000400          DPE   = 000010          EHT132  071436          EMS27   073355
ALL     062134          BIT09 = 001000          DPEHI = 040000          EHT142  071442          EMS3    071662
AMADR1= 000000          BIT1  = 000002          DPELO = 020000          EHT145  071446          EMS30   073421
AMADR2= 000000          BIT10 = 002000          DPR   = 000400          EHT150  071452          EMS300  076634
AMADR3= 000000          BIT11 = 004000          DRQ   = 004000          EHT2    071372          EMS301  076652
AMADR4= 000000          BIT12 = 010000          DRVCLR= 000010          EHT213  071456          EMS302  076675
AMAMS1= 000000          BIT13 = 020000          DRY   = 000200          EHT220  071462          EMS303  076722
AMAMS2= 000000          BIT14 = 040000          DSWR  = 177570          EHT5    071376          EMS304  076733
AMAMS3= 000000          BIT15 = 100000          DTE   = 010000          EHT57   071406          EMS306  076742
AMAMS4= 000000          BIT2  = 000004          DTO   = 010000          EHT65   071412          EMS307  076756
AMSGAD= 000000          BIT3  = 000010          DULPRT= 024024          EHT7    071402          EMS31   073501
AMSGLG= 000000          BIT4  = 000020          DVA   = 004000          EHT71   071416          EMS310  077017
AMSGTY= 000000          BIT5  = 000040          DVC   = 000200          EHT74   071422          EMS311  077034
AMTYP1= 000000          BIT6  = 000100          EBL   = 020000          EH1     102210          EMS312  077100
AMTYP2= 000000          BIT7  = 000200          ECH   = 000100          EH115   102607          EMS313  077136
AMTYP3= 000000          BIT8  = 000400          ECI   = 004000          EH130   102704          EMS314  077167
AMTYP4= 000000          BIT9  = 001000          ECRC  = 001000          EH132   103022          EMS315  077216
AOE   = 001000          BOTADR  054672          EDT1    071466          EH142   103120          EMS316  077244
APASS = 000000          BOTFLG  054674          EDT115  071504          EH145   103216          EMS317  077265
APE   = 100000          PPTVEC= 000014          EDT130  071506          EH150   103333          EMS32   073552
APRIOR= 000000          BSE   = 100000          EDT132  071510          EH2     102266          EMS320  077305
APTCSU= 000040          BUFFER  103712          EDT2    071470          EH213   103431          EMS321  077325
APTENV= 000001          BUFONE  103712          EDT220  071512          EH220   103526          EMS322  077345
APTSIZ= 000200          BUFTWO  104716          EDT5    071472          EH3     102274          EMS323  077365
APTSPD= 000100          CC    = 004000          EDT57   071474          EH5     102303          EMS324  077404
ASWREG= 000000          CH    = 002000          EDT65   071476          EH57    102360          EMS325  077424
ATA   = 100200          CHRCNT  054675          EDT71   071500          EH65    102456          EMS326  077435
ATESTN= 000000          CKSWR = 104410          EDT74   071502          EH7     102341          EMS327  077445
ATNMSK= 000377          CLOCK   001532          ED1     103564          EH71    102532          EMS33   073632
ATNTBL  063052          CLR   = 000040          ED115   103640          EMS1    071542          EMS330  077453
AUNIT = 000000          CMNSTA  007000          ED130   103652          EMS10   072145          EMS331  077472
AUSWR = 000000          CNSL00  062234          ED2     103574          EMS11   072210          EMS332  077522
```

EMS333	077540	EMS47	074514	EMT116	065676	EMT20	064056	EMT262	070770
EMS334	077554	EMS5	071760	EMT117	065716	EMT200	067300	EMT263	071006
EMS335	077564	EMS50	074554	EMT12	063732	EMT201	067316	EMT264	071024
EMS336	077607	EMS500	101061	EMT120	065736	EMT202	067334	EMT265	071050
EMS337	077626	EMS501	101222	EMT121	065762	EMT203	067352	EMT266	071066
EMS34	073662	EMS502	101247	EMT122	066002	EMT204	067374	EMT267	071104
EMS340	077642	EMS503	101317	EMT123	066022	EMT205	067414	EMT27	064240
EMS341	077653	EMS504	101344	EMT124	066046	EMT206	067434	EMT270	071120
EMS342	077662	EMS505	101377	EMT125	066064	EMT207	067464	EMT271	071144
EMS343	077714	EMS506	101452	EMT126	066104	EMT21	064076	EMT272	071170
EMS344	077730	EMS507	101513	EMT127	066122	EMT210	067500	EMT273	071206
EMS345	077757	EMS51	074623	EMT13	063750	EMT211	067516	EMT274	071224
EMS346	100005	EMS510	101603	EMT130	066146	EMT212	067534	EMT275	071242
EMS347	100023	EMS511	101656	EMT131	066164	EMT213	067546	EMT276	071262
EMS35	073711	EMS52	074672	EMT132	066202	EMT214	067576	EMT277	071300
EMS350	100044	EMS53	074733	EMT133	066222	EMT215	067610	EMT3	063544
EMS351	100051	EMS54	074774	EMT134	066242	EMT216	067630	EMT30	064252
EMS352	100067	EMS55	075051	EMT135	066256	EMT217	067644	EMT300	071324
EMS353	100103	EMS56	075127	EMT136	066274	EMT22	064116	EMT301	071350
EMS354	100127	EMS57	075222	EMT137	066306	EMT220	067660	EMT31	064266
EMS355	100162	EMS6	072030	EMT14	063766	EMT221	067676	EMT32	064302
EMS356	100204	EMS60	075306	EMT140	066324	EMT222	067714	EMT33	064316
EMS357	100224	EMS600	101755	EMT141	066344	EMT223	067732	EMT34	064334
EMS36	073753	EMS601	102005	EMT142	066360	EMT224	067750	EMT35	064352
EMS360	100246	EMS602	102025	EMT143	066376	EMT225	067770	EMT36	064366
EMS361	100274	EMS603	102066	EMT144	066412	EMT226	070012	EMT37	064402
EMS362	100322	EMS604	102107	EMT145	066434	EMT227	070030	EMT4	063564
EMS363	100343	EMS605	102134	EMT146	066456	EMT23	064132	EMT40	064416
EMS364	100360	EMS606	102152	EMT147	066474	EMT230	070046	EMT41	064436
EMS365	100401	EMS607	102172	EMT15	064004	EMT231	070064	EMT42	064452
EMS366	100406	EMS61	075360	EMT150	066506	EMT232	070100	EMT43	064476
EMS367	100431	EMS62	075433	EMT151	066530	EMT233	070122	EMT44	064510
EMS37	074003	EMS63	075514	EMT152	066542	EMT234	070144	EMT45	064524
EMS370	100501	EMS64	075554	EMT153	066556	EMT235	070172	EMT46	064540
EMS371	100516	EMS65	075624	EMT154	066576	EMT236	070212	EMT47	064554
EMS372	100545	EMS66	075665	EMT155	066616	EMT237	070244	EMT5	063606
EMS373	100577	EMS67	075742	EMT156	066634	EMT24	064152	EMT50	064576
EMS374	100603	EMS7	072074	EMT157	066656	EMT240	070264	EMT51	064620
EMS375	100634	EMS70	076054	EMT16	064022	EMT241	070310	EMT52	064640
EMS376	100650	EMTVEC=	000030	EMT160	066674	EMT242	070334	EMT53	064654
EMS377	100664	EMT1	063510	EMT161	066724	EMT243	070360	EMT54	064670
EMS4	071727	EMT10	063676	EMT162	066742	EMT244	070404	EMT55	064710
EMS40	074035	EMT100	065312	EMT163	066760	EMT245	070422	EMT56	064730
EMS400	100675	EMT101	065332	EMT164	067002	EMT246	070440	EMT57	064744
EMS401	100712	EMT102	065352	EMT165	067024	EMT247	070464	EMT6	063632
EMS402	100727	EMT103	065372	EMT166	067050	EMT25	064172	EMT60	064760
EMS403	100736	EMT104	065416	EMT167	067070	EMT250	070502	EMT61	064774
EMS404	100777	EMT105	065436	EMT17	064040	EMT251	070526	EMT62	065014
EMS405	101016	EMT106	065456	EMT170	067112	EMT252	070552	EMT63	065034
EMS406	101026	EMT107	065502	EMT171	067124	EMT253	070576	EMT64	065050
EMS407	101041	EMT11	063714	EMT172	067140	EMT254	070614	EMT65	065062
EMS41	074113	EMT110	065522	EMT173	067156	EMT255	070632	EMT66	065076
EMS42	074161	EMT111	065544	EMT174	067200	EMT256	070656	EMT67	065112
EMS43	074243	EMT112	065564	EMT175	067226	EMT257	070674	EMT7	063654
EMS44	074314	EMT113	065604	EMT176	067244	EMT26	064206	EMT70	065130
EMS45	074355	EMT114	065630	EMT177	067262	EMT260	070720	EMT71	065150
EMS46	074432	EMT115	065652	EMT2	063516	EMT261	070744	EMT72	065174

SYMBOL TABLE

```
EMT73   065220    ILR    = 000002   NOTAVL  062732   RMAS   = 000016   RMOF   = 000032
EMT74   065240    ILRG50= 000050    NOTPRS  062715   RMASI    001350   RMOFI    001364
EMT75   065250    ILRG52= 000052    NSA    = 100000   RMASO    001424   RMOFO    001440
EMT76   065262    ILRG54= 000054    OCC    = 100000   RMBA   = 000004   RMR    = 000004
EMT77   065276    ILRG56= 000056    OFD    = 000200   RMBAE  = 000050   RMSN   = 000030
ENRGDT  063510    ILRG60= 000060    OFFSET= 000014    RMBAEI   001402   RMSNI    001362
EQUALS  062132    ILRG62= 000062    OM     = 000001   RMBAEO   001456   RMSNO    001436
ERR    = 040000   ILRG64= 000064    ONES     063122   RMBAI    001336   RMWC   = 000002
ERRNMB  054670    ILRG66= 000066    OPE    = 020000   RMBAO    001412   RMWCI    001334
ERROR  = 104000   ILRG70= 000070    OPI    = 020000   RMCS1  = 000000   RMWCO    001410
ERRTYP  054046    ILRG72= 000072    OR     = 000200   RMCS1I   001332   RQA    = 100000
ERRVEC= 000004    ILRG74= 000074    PACACK= 000022    RMCS10   001406   RQB    = 040000
ERTY00  054676    ILRG76= 000076    PAKACK= 000022    RMCS2  = 000010   RTC    = 000016
ERTY01  054704    IOTVEC= 000020    PAR    = 000010   RMCS2I   001342   R6     =%000006
ERTY02  054714    IPCK0  = 000001   PAT    = 000020   RMCS20   001416   R7     =%000007
ERTY03  054723    IPCK1  = 000002   PCLOCK  055220    RMCS3  = 000052   SADMSK= 000377
ERTY04  054731    IPCK2  = 000004   PCOUNT  055306    RMCS3I   001404   SAVREG= 104414
ERTY05  054734    IPCK3  = 000010   PDA    = 000400   RMCS30   001460   SA1    = 000001
ESRC   = 004000   IR     = 000100   PGE    = 002000   RMDA   = 000006   SA16   = 000020
FER    = 000020   IVC    = 010000   PGM    = 001000   RMDAI    001340   SA2    = 000002
FMT16  = 010000   LBC    = 002000   PHA    = 000200   RMDAO    001414   SA4    = 000004
FNCDTB  062752    LBT    = 002000   PIP    = 020000   RMDB   = 000022   SA8    = 000010
FNCMSK= 000077    LCLOCK  055236    PIRQ   = 177772   RMDBI    001354   SC     = 100000
F0     = 000002   LCOUNT  055306    PIRQVE= 000240    RMDB0    001430   SCOPE  = 000004
F1     = 000004   LF     = 000012   PLCLK   055244    RMDC   = 000034   SC1MSK= 003700
F2     = 000010   LODEV   062675    PLFS   = 002000   RMDCI    001366   SC0    = 000100
F3     = 000020   LS     = 000004   PLSTP   055354    RMDCO    001442   SC1    = 000200
F4     = 000040   LSC    = 004000   PR0    = 000000   RMDS   = 000012   SC2    = 000400
GETBUF  001332    LST    = 000002   PR1    = 000040   RMDSI    001344   SC3    = 001000
GO     = 000001   LSTOP   055350    PR2    = 000100   RMDS0    001420   SC4    = 002000
GTSWR  = 104407   LSTRK   001330    PR3    = 000140   RMDT   = 000026   SEARCH= 000030
HCE    = 000200   MCLK   = 004000   PR4    = 000200   RMDTI    001360   SEEK   = 000004
HCI    = 002000   MCPE   = 020000   PR5    = 000240   RMDTO    001434   SETOM    055512
HCRC   = 000400   MDF    = 000100   PR6    = 000300   RMEC1  = 000044   SETVV    055370
HELP    103712    MDPE   = 000400   PR7    = 000340   RMEC1I   001376   SHUT     055642
HT     = 000011   MI     = 000004   PS     = 177776   RMEC10   001452   SIZCLK   054760
IAE    = 002000   MIXED    063062   PSEL   = 002000   RMEC2  = 000046   SKI    = 040000
IDXMSK= 000077    MOC    = 000400   PSTOP   055342    RMEC2I   001400   SNGPRT= 020024
IE     = 000100   MOH    = 020000   PSW    = 177776   RMEC20   001454   STACK  = 001100
ILF    = 000001   MUL    = 010000   PUTBUF  001406    RMER1  = 000014   STANDA   006074
ILF02  = 000002   MRD    = 002000   PWRVEC= 000024    RMER1I   001346   START    004626
ILF24  = 000024   MR1AAA= 051401    QUES     062141   RMER10   001422   STKLMT= 177774
ILF26  = 000026   MS     = 000040   RD     = 000070   RMER2  = 000042   STOP     001534
ILF30  = 000030   MSC    = 000002   RDCHR  = 104411   RMER2I   001374   SWR      001154
ILF32  = 000032   MSDRVS  062653    RDLIN  = 104412   RMER20   001450   SWREG    000176
ILF34  = 000034   MSE    = 100000   RDOCT  = 104413   RMHR   = 000036   SW0    = 000001
ILF36  = 000036   MSER   = 000200   RDY    = 000200   RMHRI    001370   SW00   = 000001
ILF40  = 000040   MSGDRV  062667    READY    007152   RMHR0    001444   SW01   = 000002
ILF42  = 000042   MSHELP  062150    RECAL  = 000006   RMLA   = 000020   SW02   = 000004
ILF44  = 000044   MUR    = 001000   RESREG= 104415    RMLAI    001352   SW03   = 000010
ILF46  = 000046   MWD    = 000010   RESVEC= 000010    RMLA0    001426   SW04   = 000020
ILF54  = 000054   MWP    = 000010   REX    = 010000   RMMR1  = 000024   SW05   = 000040
ILF56  = 000056   MXF    = 001000   RG     = 040000   RMMR1I   001356   SW06   = 000100
ILF64  = 000064   NDTMSK= 115760    RGDTPT  063062    RMMR10   001432   SW07   = 000200
ILF66  = 000066   NED    = 010000   RH     = 000072   RMMR2  = 000040   SW08   = 000400
ILF74  = 000074   NEM    = 004000   RIP    = 000020   RMMR2I   001372   SW09   = 001000
ILF76  = 000076   NOP    = 000000   RLEASE= 000012    RMMR20   001446   SW1    = 000002
```

```
SW10  = 002000          TST16  014672          TYPDS = 104405          $DDW1  001310          $MAMS1  001252
SW11  = 004000          TST17  015404          TYPE  = 104401          $DDW2  001312          $MAMS2  001256
SW12  = 010000          TST2   007616          TYPOC = 104402          $DDW3  001314          $MAMS3  001262
SW13  = 020000          TST20  015506          TYPON = 104404          $DDW4  001316          $MAMS4  001266
SW14  = 040000          TST21  015772          TYPOS = 104403          $DDW5  001320          $MBADR  001102
SW15  = 100000          TST22  016110          UBUSQS  062201          $DDW6  001322          $MFLG   062126
SW2   = 000004          TST23  016236          UNS   = 040000          $DDW7  001324          $MNEW   061264
SW3   = 000010          TST24  016510          UNTMSK= 000007          $DEVCT 001232          $MSGAD  001236
SW4   = 000020          TST25  017010          UPE   = 020000          $DEVM  001300          $MSGLG  001240
SW5   = 000040          TST26  017432          USE   = 040000          $DOAGN 054036          $MSGTY  001222
SW6   = 000100          TST27  017526          U0    = 000001          $DTBL  056264          $MSWR   061253
SW7   = 000200          TST3   010000          U1    = 000002          $ENDAD 054026          $MTYP1  001253
SW8   = 000400          TST30  020014          U2    = 000004          $ENDCT 053672          $MTYP2  001257
SW9   = 001000          TST31  020330          VV    = 000100          $ENULL 054042          $MTYP3  001263
TADMSK= 177400          TST32  020620          WATCH   001530          $ENV   001242          $MTYP4  001267
TAG   = 020000          TST33  021336          WC    = 000040          $ENVM  001243          $MXCNT  057376
TAGADR= 001114          TST34  021650          WCD   = 000050          $EOP   053636          $NULL   001170
TAP   = 040000          TST35  022142          WCE   = 040000          $EOPCT 053664          $NWTST= 000001
TA1   = 000400          TST36  022464          WCEHI = 010000          $EOSP  053600          $OCNT   056526
TA16  = 010000          TST37  023024          WCELO = 004000          $ERFLG 001117          $OMODE  056530
TA2   = 001000          TST4   010150          WCF   = 000040          $ERMAX 001131          $OVER   057362
TA4   = 002000          TST40  023500          WCH   = 000052          $ERROR 057640          $PASS   001230
TA8   = 004000          TST41  023764          WD    = 000060          $ERRPC 001132          $PASTM  001106
TBITVE= 000014          TST42  024246          WH    = 000062          $ERRTB 001536          $POWER  061654
TIME    001526          TST43  024650          WLE   = 004000          $ERTTL 001126          $PSW    001524
TKVEC = 000060          TST44  025020          WRL   = 004000          $ESCAP 001210          $PWRDN  061506
TPVEC = 000064          TST45  025400          XNUDC = 176000          $ETABL 001242          $PWRMG  061642
TRAPVE= 000034          TST46  026110          XNUER2= 001567          $ETEND 001326          $PWRUP  061560
TRE   = 040000          TST47  026404          XNUOF = 161577          $FATAL 001224          $QUES   001216
TRTVEC= 000014          TST5   010300          XSIZ    005666          $FFLG  062130          $RDCHR  060700
TST   = 010000          TST50  027006          XXDP    001326          $FILLC 001172          $RDLIN  060770
TSTNMB  054666          TST51  027242          ZEROS   063164          $FILLS 001171          $RDOCT  061276
TSTQUE  001462          TST52  027450          $APTHD  001100          $GDADR 001134          $RDSZ = 000010
TST1    007306          TST53  030302          $ATYC   061710          $GDDAT 001140          $RESRE  055746
TST10   012572          TST54  030554          $ATY1   061664          $GET42 054016          $RM02   054740
TST100  035462          TST55  030756          $ATY3   061672          $GTSWR 060426          $RM03   054745
TST101  035614          TST56  031170          $ATY4   061702          $HD   = 000000          $RM05   054752
TST102  036004          TST57  031364          $AUTOB  001150          $HIBTS 001100          $RTNAD  054040
TST103  036206          TST6   011364          $BASE   001276          $HIOCT 061376          $SAVRE  055710
TST104  036334          TST60  031544          $BDADR  001136          $ICNT  001120          $SAVR6  061652
TST105  036540          TST61  031762          $BDDAT  001142          $ILLUP 061646          $SCOPE  057066
TST106  036670          TST62  032130          $BELL   001212          $INTAG 001151          $SETUP= 000137
TST107  037064          TST63  032354          $BIN    056056          $ITEMB 001130          $STUP = 177777
TST11   012656          TST64  032470          $CDW1   001302          $LF    001220          $SVLAD  057326
TST110  037262          TST65  032656          $CDW2   001304          $LFLG  062127          $SVPC = 000204
TST111  037540          TST66  033106          $CHARC  057062          $LLCSR 001516          $SWR  = 167400
TST112  040044          TST67  033324          $CKSWR  060336          $LLVEC 001520          $SWREG  001244
TST113  041436          TST7   012434          $CMTAG  001114          $LPADR 001122          $SWRMK= 000000
TST114  043164          TST70  033552          $CM3  = 000000          $LPCSB 001510          $SW08T  057400
TST115  045344          TST71  034022          $CM4  = 000005          $LPCSR 001506          $TESTN  001226
TST116  045724          TST72  034230          $CNTLC  061234          $LPERR 001124          $TIMES  001206
TST117  050104          TST73  034520          $CNTLG  061246          $LPVEC 001512          $TKB    001162
TST12   013116          TST74  034746          $CNTLU  061241          $MADR1 001254          $TKCNT  060040
TST120  052372          TST75  035054          $CPUOP  001250          $MADR2 001260          $TKINT  060050
TST13   013464          TST76  035200          $CRLF   001217          $MADR3 001264          $TKQEN= 060047
TST14   014246          TST77  035330          $DBLK   056274          $MADR4 001270          $TKQIN  060042
TST15   014346          TYPBN - 104406         $DDW0   001306          $MAIL  001222          $TKQOU  060044
```

```
$TKQSR  060046          $TN   = 000121          $TSTM   001104          $TYPOC  056330          $XOFF = 000023
$TKS    001160          $TPB    001166          $TSTNM  001116          $TYPON  056344          $XON  = 000021
$TKSRV  060120          $TPFLG  001173          $TTYIN  061224          $TYPOS  056304          $XTSTR  057100
$TMP0   001174          $TPS    001164          $TYPBN  056004          $UNIT   001234          $$GET4= 000000
$TMP1   001176          $TRAP   061400          $TYPDS  056060          $UNITM  001110          $$SW08= 000121
$TMP2   001200          $TRAP2  061440          $TYPE   056532          $USWR   001246          $OFILL  056527
$TMP3   001202          $TRP  = 000016          $TYPEC  056744          $VECT1  001272          .$X   = 001100
$TMP4   001204          $TRPAD  061452          $TYPEX  057064          $VECT2  001274

. ABS.  110342      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  55280 WORDS  ( 216 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  69 PAGES
CZRMPA.BIC,CZRMPA/C-CZRMPA.DOC,CZRMPA,SYSMAC/M
```

```
$$GET4   13-20     13-20#
$$SW08   19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6      19-6
         19-6      19-6      19-6      19-6      19-6      19-6      19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#     19-6#
         19-6#     19-6#     19-6#
$0FILL   19-4      19-4#     19-4*     19-4*
$40CAT   19-6      19-7
$APTHD    4-991     4-991#
$ASTAT   19-12     19-12
$ATY1    19-12#
$ATY3    19-5      19-12#
$ATY4    19-7      19-12#
$ATYC    19-12     19-12#
$AUTOB    5-0#      8-25*     8-62     10-27*    19-8      19-8      19-8
$BASE     5-0#      8-72      9-56      9-65*    10-46     12-1      12-67     12-73     12-106    12-137    12-159    12-329    12-521    12-548
         12-559    12-612    12-679    12-783    12-796    12-866    12-975    12-998    12-:51    12-:79    12-:99    12-;43    12-<05    12-=00
         12-=17    12-=75    12->40    12-?01    12-@03    12-@64    12-A24    12-B06    12-B74    12-C76    12-D29    12-D82    12-E62    12-E97
         12-G16    12-H68    12-I43    12-J82    12-K28    12-K62    12-L94    12-M54    12-M90    12-N38    12-N77    12-007    12-047    12-073
         12-P17    12-P38    12-P65    12-Q09    12-Q86    12-R34    12-R84    12-S21    12-S71    12-T28    12-T46    12-T69    12-T98    12-U27
         12-U57    12-U94    12-V32    12-V52    12-V85    12-W06    12-W55    12-X00    12-X70    12-Y47    12-\09    12-_00    12-c06    12-c74
         12-g76    12-k50    18-9
$BDADR    5-0#     12-14*    12-15*    12-35*    12-36*    12-67*    12-542*   12-543*   12-555*   12-563*   12-564*   12-625*   12-626*   12-638*
         12-639*   12-652*   12-653*   12-672*   12-673*   12-692*   12-693*   12-700*   12-701*   12-708*   12-709*   12-721*   12-722*   12-729*
         12-730*   12-738*   12-739*   12-751*   12-752*   12-775*   12-776*   12-791*   12-792*   12-799*   12-800*   12-869*   12-870*   12-978*
         12-?79*   12-:15*   12-:16*   12-;11*   12-;12*   12-;22*   12-;23*   12-;36*   12-;37*   12-;48*   12-;49*   12-<17*   12-<18*   12-<44*
         12-=03*   12-=19*   12-=20*   12-=78*   12-=79*   12->43*   12->44*   12-?05*   12-?06*   12-?27*   12-?28*   12-?44*   12-?45*   12-?60*
         12-?61*   12-?86*   12-?87*   12-@07*   12-@08*   12-@68*   12-@69*   12-A27*   12-A28*   12-B09*   12-B10*   12-B76*   12-B77*   12-C79*
         12-C80*   12-D31*   12-D32*   12-D85*   12-D86*   12-E64*   12-E99*   12-F00*   12-G41*   12-G55*   12-G56*   12-G68*   12-G93*   12-G94*
         12-H71*   12-I45*   12-I46*   12-J84*   12-J85*   12-K30*   12-K31*   12-K82*   12-K83*   12-L03*   12-L04*   12-L24*   12-L25*   12-L58*
         12-L59*   12-L73*   12-L74*   12-L85*   12-L86*   12-L96*   12-L97*   12-M77*   12-M78*   12-N24*   12-N25*   12-N79*   12-N80*   12-032*
         12-033*   12-041*   12-042*   12-049*   12-050*   12-075*   12-076*   12-P12*   12-P13*   12-P19*   12-P20*   12-P42*   12-P43*   12-P69*
         12-P70*   12-R20*   12-R21*   12-R64*   12-R65*   12-S08*   12-S09*   12-S58*   12-S59*   12-S73*   12-S74*   12-T41*   12-T54*   12-T63*
         12-T64*   12-T93*   12-T94*   12-U22*   12-U23*   12-U51*   12-U52*   12-U89*   12-U90*   12-V14*   12-V15*   12-V27*   12-V28*   12-V46*
         12-V47*   12-V66*   12-V67*   12-V78*   12-V79*   12-W00*   12-W01*   12-W27*   12-W28*   12-W79*   12-W80*   12-X31*   12-X32*   12-X92*
         12-X93*   12-Y22*   12-Y23*   12-Y79*   12-Y80*   12-Z17*   12-Z47*   12-Z48*   12-Z90*   12-Z91*   12-[32*   12-[47*   12-[72*   12-[73*
         12-\46*   12-\47*   12-\87*   12-]20*   12-]21*   12-]66*   12-]67*   12-^11*   12-^26*   12-^56*   12-^57*   12-_37*   12-_38*   12-_78*
         12-`14*   12-`15*   12-`77*   12-`78*   12-a22*   12-a37*   12-a92*   12-a93*   12-b09*   12-b53*   12-b54*   12-c50*   12-c51*   12-c66*
         12-c07*   12-d10*   12-d11*   12-d59*   12-d60*   12-d87*   12-d88*   12-e03*   12-e34*   12-e35*   12-e61*   12-e62*   12-e91*   12-e92*
         12-f29*   12-f30*   12-f63*   12-f64*   12-g29*   12-g30*   12-h01*   12-h02*   12-h17*   12-h18*   12-h45*   12-h46*   12-h97*   12-h98*
         12-i32*   12-i61*   12-i62*   12-i93*   12-i94*   12-j50*   12-j51*   12-j79*   12-j80*   12-k15*   12-k16*   12-k44*   12-k45*   12-l12*
         12-l13*   12-l43*   12-l44*   12-m00*   12-m01*   12-m32*   12-m33*   16-23*    16-24*    17-22*    17-23*    30-1      30-2      30-4
         30-7      30-8      30-9
```

```
$DDW7    5-0#
$DEVCT   5-0#
$DEVM    5-0#     8-64*    8-74     8-93*    9-115*   9-122*   9-149*   9-150    10-3     10-30
$DOAGN   13-20    13-20    13-20#
$DTBL    19-3     19-3#
$ENDAD   4-988    8-25     13-20#   19-7
$ENDCT   8-19     13-20#
$ENULL   13-20#
$ENV     5-0#     8-25     10-27    19-5     19-7     19-12    19-12
$ENVM    5-0#     8-19     8-68     19-5     19-5     19-12
$EOP     10-35    13-20#   18-20
$EOPCT   8-19*    13-20    13-20#
$EOSP    12-69    12-545   13-9#
$ERFLG   5-0#     19-6     19-6     19-6     19-6     19-6     19-6*    19-7     19-7     19-7*
$ERMAX   5-0#     8-19*    19-6     19-6     19-6     19-6*
$ERROR   8-19     19-7#
$ERRPC   5-0#     14-53    19-7     19-7     19-7     19-7*    19-7*
$ERRTB   7-0#     14-64
$ERTTL   5-0#     13-20    13-20*   19-7     19-7     19-7*
$ESCAP   5-0#     8-19*    19-6*    19-7     19-7     19-7     19-7
$ETABL   5-0#
$ETEND   4-991    5-0#
$FATAL   5-0#     19-12*
$FFLG    19-12    19-12#   19-12*   19-12*   19-12*
$FILLC   5-0#     19-5     19-5     19-5
$FILLS   5-0#     19-5     19-5
$GDADR   5-0#     12-T42*
$GDDAT   5-0#     12-11*   12-12*   12-13*   12-32*   12-33*   12-34*   12-209*  12-253*  12-312*  12-554*  12-575*  12-585*  12-586
         12-603*  12-624*  12-637*  12-649*  12-650   12-671*  12-691*  12-699*  12-707*  12-720*  12-728*  12-737*  12-748*  12-749
         12-774*  12-790*  12-801*  12-840*  12-841   12-859*  12-871*  12-942*  12-943   12-964*  12-965*  12-966   12-983*  12-988
         12-:01*  12-:30   12-:45*  12-:55*  12-;10*  12-;21*  12-;35*  12-;47*  12-;75*  12-;76   12-;96*  12-<07*  12-<16*  12-<50
         12-<53   12-<81*  12-=12*  12-=28*  12-=38*  12-=49*  12-=66*  12-=87*  12-=98*  12->10*  12->25*  12->45*  12->63*  12->85*
         12-?07*  12-?19*  12-?25   12-?29   12-?38*  12-?48*  12-?52   12-?54*  12-?58   12-?62   12-?80*  12-?83*  12-?84   12-@16*
         12-@28*  12-@43*  12-@46*  12-@47   12-@77*  12-@87*  12-A03*  12-A06*  12-A07   12-A29*  12-A57*  12-A77*  12-B21*  12-B30*
         12-B31   12-B40*  12-B49*  12-B50   12-B79*  12-C05   12-C13*  12-C16*  12-C17*  12-C26*  12-C47   12-C54*  12-C57*  12-C69*
         12-C86*  12-D07   12-D14*  12-D15   12-D17*  12-D53*  12-D71*  12-D96*  12-D98*  12-E46*  12-E49*  12-E50   12-E85*  12-F17*
         12-F41*  12-F42   12-G40*  12-G47*  12-G50*  12-G53   12-G69*  12-G83*  12-G87*  12-G90*  12-G95   12-H85*  12-I16*  12-I61*
         12-I86*  12-I89*  12-I90   12-J98*  12-K12*  12-K13*  12-K14   12-K42*  12-K57*  12-K81*  12-L02*  12-L23*  12-L57*  12-M10*
         12-M29*  12-M73*  12-M74*  12-M75   12-N17*  12-N20*  12-N21   12-N81*  12-O09*  12-O51*  12-P03*  12-P11*  12-P33*  12-P61*
         12-P67*  12-P87   12-Q04*  12-R16*  12-R17*  12-R18   12-R63*  12-S07*  12-S60*  12-S94*  12-S95   12-T40*  12-T53*  12-T62*
         12-V16*  12-V26*  12-V45*  12-V68*  12-V80*  12-V99*  12-W23*  12-W24   12-W81*  12-X30*  12-X94*  12-Y21*  12-Y78*  12-Z18*
         12-Z49*  12-Z89*  12-[31*  12-[48*  12-[71*  12-\45*  12-\88*  12-]22*  12-]65*  12-^10*  12-^27*  12-^52*  12-^53*  12-^54
         12- 36*  12- 79*  12-`16*  12-`76*  12-a21*  12-a38*  12-a91*  12-b08*  12-b49*  12-b50*  12-b51   12-c49*  12-c65*  12-d09*
         12-d58*  12-d86*  12-e02*  12-e33*  12-e58*  12-e59   12-e90*  12-f28*  12-f62*  12-g00*  12-g03*  12-g04   12-h00*  12-h16*
         12-h44*  12-h96*  12-i30*  12-i31*  12-i60*  12-i92*  12-j49*  12-j78*  12-k14*  12-k43*  12-{11*  12-{33*  12-{34*  12-{35
         12-{71*  12-{78*  12-{81*  12-{82*  12-{88*  12-{92   12-m22*  12-m25   16-25*   17-21*   30-1     30-3     30-5     30-6
         30-7     30-8
$GET42   13-20#
$GTSWR   19-8#    19-10    19-10
$HD      4-684    4-684    4-684
$HIBTS   4-991#
$HIOCT   19-9#    19-9*
$ICNT    5-0#     19-6     19-6     19-6     19-6*    19-6*
$ILLUP   19-11    19-11    19-11#
$INTAG   5-0#     19-8     19-8     19-8     19-8     19-8*
$ITEMB   5-0#     14-48    19-7     19-7     19-7     19-7*
```

SLF 5-0# 19-5 19-5 19-7 19-7 19-8 19-8 H 6 19-8

```
$LFLG   19-12#  19-12*
$LLCSR   6-0#   15-26   15-55*  15-83*
$LLVEC   6-0#   15-17   15-17*  15-18*  15-29*  15-30*  15-42   15-42*  15-43*
$LPADR   5-0#    8-19*  19-6    19-6    19-6    19-6*   19-6*   19-6*
$LPCSB   6-0#   15-51*
$LPCSR   6-0#   15-12   15-52*  15-80*
$LPERR   5-0#    8-19*  12-Y83* 12-Z21* 12-Z52* 12-Z94* 12-[51* 12-\50* 12-\91* 12-]25* 12-]70* 12-^30* 12-_41* 12-`85*
        12-`19*  12-`81* 12-a41* 12-b12* 12-d14* 12-e06* 12-e65* 12-f37* 12-h20* 12-i00* 12-i64* 12-j54* 19-6    19-6
        19-6    19-6*   19-7
$LPVEC   6-0#   15-15*  15-16*  15-31   15-31*  15-32*  15-40   15-40*  15-41*
$MADR1   5-0#
$MADR2   5-0#
$MADR3   5-0#
$MADR4   5-0#
$MAIL    4-991   4-991   5-0#    8-19    8-25    10-27   12-1    12-73   12-106  12-137  12-159  12-329  12-521  12-548
        12-559  12-612  12-679  12-783  12-796  12-866  12-975  12-998  12-:51  12-:79  12-:99  12-;43  12-<05  12-=00
        12-=17  12-=75  12->40  12-?01  12-@03  12-@64  12-A24  12-B06  12-B74  12-C76  12-D29  12-D82  12-E62  12-E97
        12-G18  12-H68  12-I43  12-J82  12-K28  12-K62  12-L94  12-M54  12-M90  12-N38  12-N77  12-007  12-047  12-073
        12-P17  12-P38  12-P65  12-Q09  12-Q86  12-R34  12-R84  12-S21  12-S71  12-T28  12-T46  12-T69  12-T98  12-U27
        12-U57  12-U94  12-V32  12-V52  12-V85  12-W06  12-W55  12-X00  12-X70  12-Y47  12-\09  12-_00  12-c06  12-c74
        12-g76  12-k50  19-5    19-6    19-7
$MAMS1   5-0#
$MAMS2   5-0#
$MAMS3   5-0#
$MAMS4   5-0#
$MBADR   4-991#
$MFLG   19-12   19-12#  19-12*  19-12*
$MNEW   19-8    19-8#
$MSGAD   5-0#   19-12   19-12*
$MSGLG   5-0#   19-12*
$MSGTY   5-0#   19-12   19-12   19-12*  19-12*
$MSWR   19-8    19-8#
$MTYP1   5-0#
$MTYP2   5-0#
$MTYP3   5-0#
$MTYP4   5-0#
$MXCNT  19-6    19-6    19-6    19-6#
$NULL    5-0#   19-5    19-5    19-5
$NWTST  12-1    12-1    12-1#   12-1#   12-73   12-73   12-73#  12-73#  12-106  12-106  12-106#  12-106#  12-137  12-137
        12-137# 12-137# 12-159  12-159  12-159# 12-159# 12-329  12-329  12-329# 12-329# 12-521  12-521  12-521# 12-521#
        12-548  12-548  12-548# 12-548# 12-559  12-559  12-559# 12-559# 12-612  12-612  12-612# 12-612# 12-679  12-679
        12-679# 12-679# 12-783  12-783  12-783# 12-783# 12-796  12-796  12-796# 12-796# 12-866  12-866  12-866# 12-866#
        12-975  12-975  12-975# 12-975# 12-998  12-998  12-998# 12-998# 12-:51  12-:51  12-:51# 12-:51# 12-:79  12-:79
        12-:79# 12-:79# 12-:99  12-:99  12-:99# 12-:99# 12-;43  12-;43  12-;43# 12-;43# 12-<05  12-<05  12-<05# 12-<05#
        12-=00  12-=00  12-=00# 12-=00# 12-=17  12-=17  12-=17# 12-=17# 12-=75  12-=75  12-=75# 12-=75# 12->40  12->40
        12->40# 12->40# 12-?01  12-?01  12-?01# 12-?01# 12-@03  12-@03  12-@03# 12-@03# 12-@64  12-@64  12-@64# 12-@64#
        12-A24  12-A24  12-A24# 12-A24# 12-B06  12-B06  12-B06# 12-B06# 12-B74  12-B74  12-B74# 12-B74# 12-C76  12-C76
        12-C76# 12-C76# 12-D29  12-D29  12-D29# 12-D29# 12-D82  12-D82  12-D82# 12-D82# 12-E62  12-E62  12-E62# 12-E62#
        12-E97  12-E97  12-E97# 12-E97# 12-G18  12-G18  12-G18# 12-G18# 12-H68  12-H68  12-H68# 12-H68# 12-I43  12-I43
        12-I43# 12-I43# 12-J82  12-J82  12-J82# 12-J82# 12-K28  12-K28  12-K28# 12-K28# 12-K62  12-K62  12-K62# 12-K62#
        12-L94  12-L94  12-L94# 12-L94# 12-M54  12-M54  12-M54# 12-M54# 12-M90  12-M90  12-M90# 12-M90# 12-N38  12-N38
        12-N38# 12-N38# 12-N77  12-N77  12-N77# 12-N77# 12-007  12-007  12-007# 12-007# 12-047  12-047  12-047# 12-047#
        12-073  12-073  12-073# 12-073# 12-P17  12-P17  12-P17# 12-P17# 12-P38  12-P38  12-P38# 12-P38# 12-P65  12-P65
        12-P65# 12-P65# 12-Q09  12-Q09  12-Q09# 12-Q09# 12-Q86  12-Q86  12-Q86# 12-Q86# 12-R34  12-R34  12-R34# 12-R34#
        12-R84  12-R84  12-R84# 12-R84# 12-S21  12-S21  12-S21# 12-S21# 12-S71  12-S71  12-S71# 12-S71# 12-T28  12-T28
        12-T28# 12-T28# 12-T46  12-T46  12-T46# 12-T46# 12-T69  12-T69  12-T69# 12-T69# 12-T98  12-T98  12-T98# 12-T98#
```

J 6

12-U27 12-U27 12-U27# 12-U27# 12-U57 12-U57 12-U57# 12-U57# 12-U94 12-U94 12-U94# 12-U94# 12-V32 12-V32

SEQ 0280

$TIMES	5-0#	8-19*	10-38*	13-20*	19-6	19-6	19-6 L 6	19-6*	19-6*

STSTNM	5-0#	10-37*	13-20*	19-6	19-6	19-6	19-6	N 6 19-6	19-6*	19-6*	19-7	19-7	19-7

SEQ 0284

```
$TTYIN  19-8     19-8     19-8     19-8     19-8     19-8#
$TYPBN  19-2#    19-10    19-10
$TYPDS  19-3#    19-10    19-10
$TYPE   19-5#    19-10    19-10    19-12
$TYPEC  19-5     19-5     19-5     19-5#    19-8
$TYPEX  19-5     19-5     19-5#
$TYPOC  19-4#    19-10    19-10
$TYPON  19-4     19-4#    19-10
$TYPOS  19-4#    19-10
$UNIT    5-0#    10-41*   14-23
$UNITM   4-991#
$USWR    5-0#
$VECT1   5-0#     9-69     9-81*    9-85     9-107*
$VECT2   5-0#
$XOFF   19-5     19-5
$XON    19-5     19-5
$XTSTR  19-6#
.$ASTA  19-12    19-12
.$X      4-991    4-991#
A16      4-930#  12-<83
A17      4-929#  12-<83
ABASE    4-981#   5-0      5-0
ACDW1    5-0      5-0
ACDW2    5-0      5-0
ACPUOP   5-0      5-0
ADDW0    5-0      5-0
ADDW1    5-0      5-0
ADDW10   5-0
ADDW11   5-0
ADDW12   5-0
ADDW13   5-0
ADDW14   5-0
ADDW15   5-0
ADDW2    5-0      5-0
ADDW3    5-0      5-0
ADDW4    5-0      5-0
ADDW5    5-0      5-0
ADDW6    5-0      5-0
ADDW7    5-0      5-0
ADDW8    5-0
ADDW9    5-0
ADEVCT   5-0      5-0
ADEVM    5-0      5-0
ADR     12-1     12-1#    12-1#    12-73    12-73#   12-73#   12-106   12-106#  12-106#  12-137   12-137#  12-137#  12-159   12-159#
        12-159#  12-329    12-329#  12-329#  12-521   12-521#  12-521#  12-548   12-548#  12-548#  12-559   12-559#  12-559#  12-612
        12-612#  12-612#   12-679   12-679#  12-679#  12-783   12-783#  12-783#  12-796   12-796#  12-796#  12-866   12-866#  12-866#
        12-975   12-975#   12-975#  12-998   12-998#  12-998#  12-:51   12-:51#  12-:51#  12-:79   12-:79#  12-:79#  12-:99   12-:99#
        12-:99#  12-;43    12-;43#  12-;43#  12-<05   12-<05#  12-<05#  12-=00   12-=00#  12-=00#  12-=17   12-=17#  12-=17#  12-=75
        12-=75#  12-=75#   12->40   12->40#  12->40#  12-?01   12-?01#  12-?01#  12-@03   12-@03#  12-@03#  12-@64   12-@64#  12-@64#
        12-A24   12-A24#   12-A24#  12-B06   12-B06#  12-B06#  12-B74   12-B74#  12-B74#  12-C76   12-C76#  12-C76#  12-D29   12-D29#
        12-D29#  12-D82    12-D82#  12-D82#  12-E62   12-E62#  12-E62#  12-E97   12-E97#  12-E97#  12-G18   12-G18#  12-G18#  12-H68
        12-H68#  12-H68#   12-I43   12-I43#  12-I43#  12-J82   12-J82#  12-J82#  12-K28   12-K28#  12-K28#  12-K62   12-K62#  12-K62#
        12-L94   12-L94#   12-L94#  12-M54   12-M54#  12-M54#  12-M90   12-M90#  12-M90#  12-N38   12-N38#  12-N38#  12-N77   12-N77#
        12-N77#  12-007    12-007#  12-007#  12-047   12-047#  12-047#  12-073   12-073#  12-073#  12-P17   12-P17#  12-P17#  12-P38
        12-P38#  12-P38#   12-P65   12-P65#  12-P65#  12-Q09   12-Q09#  12-Q09#  12-Q86   12-Q86#  12-Q86#  12-R34   12-R34#  12-R34#
        12-R84   12-R84#   12-R84#  12-S21   12-S21#  12-S21#  12-S71   12-S71#  12-S71#  12-T28   12-T28#  12-T28#  12-T46   12-T46#
```

12-T46# 12-T69 12-T69# 12-T69# 12-T98 12-T98# 12-T98# 12-U27 12-U27# 12-U27# 12-U57 12-U57# 12-U57# 12-U94

SEQ 0286

12-A37 12-A48 12-187

SFQ 0288

CNSLOQ 20-21#

SEQ 0290

DDISP 4-688# 5-0 8-19

SEQ 0292

ED2 26-2 26-7 30-2#

K 7

SEQ 0294

EH132 25-11 29-18#

M 7

SEQ 0296

CROSS REFERENCE TABLE (CREF V01-05)

SEQ 0297

EH142	25-12	29-20#												
EH145	25-13	29-22#												
EH150	25-14	29-25#												
EH2	25-2	29-3#												
EH213	25-15	29-27#												
EH220	25-16	29-29#												
EH3	25-8	29-4#												
EH5	25-3	29-5#												
EH57	25-5	29-8#												
EH65	25-6	29-10#												
EH7	25-4	29-7#												
EH71	25-7	29-12#												
EHT1	7-3	7-49	7-52	7-55	7-58	7-61	7-64	7-70	7-73	7-76	7-79	7-82	7-85	7-88
	7-91	7-94	7-97	7-100	7-106	7-109	7-112	7-115	7-118	7-121	7-124	7-127	7-130	7-133
	7-136	7-139	7-145	7-148	7-151	7-154	7-157	7-166	7-169	7-184	7-187	7-190	7-196	7-199
	7-205	7-208	7-214	7-217	7-220	7-223	7-226	7-229	7-235	7-238	7-244	7-247	7-253	7-256
	7-259	7-277	7-280	7-283	7-286	7-298	7-301	7-307	7-310	7-316	7-319	7-322	7-325	7-328
	7-334	7-340	7-343	7-346	7-349	7-355	7-358	7-367	7-376	7-379	7-382	7-385	7-388	7-391
	7-394	7-397	7-424	7-427	7-430	7-445	7-448	7-451	7-454	7-457	7-460	7-473	7-479	7-485
	7-488	7-492	7-495	7-498	7-501	7-504	7-508	7-511	7-515	7-518	7-521	7-524	7-527	7-530
	7-533	7-537	7-540	7-543	7-546	7-549	7-552	7-555	7-558	7-561	7-564	7-567	7-570	7-573
	7-576	7-580	7-583	7-586	25-1#									
EHT115	7-232	7-241	7-250	7-262	7-409	7-412	7-476	25-9#						
EHT130	7-265	7-268	25-10#											
EHT132	7-271	7-274	7-289	25-11#										
EHT142	7-292	7-295	7-415	25-12#										
EHT145	7-304	25-13#												
EHT150	7-313	7-331	7-337	7-361	7-364	7-370	7-400	7-406	7-467	7-470	7-482	25-14#		
EHT2	7-6	25-2#												
EHT213	7-418	7-421	7-464	25-15#										
EHT220	7-433	7-436	7-439	7-442	25-16#									
EHT5	7-15	7-18	25-3#											
EHT57	7-142	25-5#												
EHT65	7-160	7-163	25-6#											
EHT7	7-22	25-4#												
EHT71	7-172	7-175	7-178	7-193	7-202	7-211	25-7#							
EHT74	7-181	25-8#												
EMS1	24-3	24-4	28-3#											
EMS10	24-129	24-131	28-10#											
EMS11	24-133	28-11#												
EMS12	24-118	24-120	24-122	28-13#										
EMS13	24-136	24-138	24-140	24-142	28-14#									
EMS14	24-144	24-146	24-150	28-15#										
EMS15	24-144	24-146	24-148	24-207	24-328	24-330	24-347	24-349	24-365	24-381	24-401	28-16#		
EMS16	24-151	24-153	24-155	24-338	28-17#									
EMS17	24-151	24-153	24-157	28-18#										
EMS2	24-8	24-187	24-211	24-239	24-243	24-248	24-259	24-340	24-345	28-4#				
EMS20	24-158	24-160	24-162	24-164	24-166	24-168	24-172	28-19#						
EMS21	24-158	24-160	24-164	24-166	24-168	24-170	24-351	24-367	24-383	24-403	28-20#			
EMS22	24-160	24-162	24-166	24-168	28-21#									
EMS23	24-173	24-175	24-179	28-22#										
EMS24	24-173	24-175	24-177	24-360	24-376	24-392	24-418	28-23#						
EMS25	24-180	24-182	24-186	28-24#										
EMS250	24-33	24-36	24-39	24-75	24-197	24-199	24-201	24-203	24-271	24-276	24-292	24-297	24-318	28-63#
EMS251	24-41	24-44	24-47	24-303	24-314	28-64#								
EMS252	24-49	24-52	24-54	24-56	24-58	24-60	24-62	24-64	24-66	24-221	24-223	24-342	28-65#	

EMS253 24-49 24-79 24-81 24-83 24-85 24-88 24-91 B 8 24-94 24-221 24-223 28-66#

SEQ 0298

F 8

24-114 24-117 24-119 24-121 24-123 24-125 24-128 24-130 24-132 24-134 24-145 24-147 24-152 24-154

SEQ 0302

	24-159	24-161	24-163	24-165	24-167	24-169	24-174	24-176	24-181	24-183	24-208	24-212	24-220	24-222
	24-224	24-226	24-232	24-234	24-246	24-251	24-253	24-255	24-257	24-260	24-262	24-264	24-266	24-268
	24-270	24-273	24-275	24-281	24-283	24-285	24-287	24-289	24-294	24-296	24-304	24-307	24-309	24-311
	24-313	24-315	24-317	24-319	24-321	24-323	24-325	24-327	24-329	24-331	24-333	24-335	24-337	24-339
	24-344	24-346	24-348	28-150#										
EMS502	24-5	24-7	24-9	24-11	24-14	24-16	24-18	24-20	24-22	24-24	24-26	24-28	24-30	24-32
	24-50	24-130	24-132	24-134	24-291	24-341	24-344	24-413	28-151#					
EMS503	24-5	24-9	24-11	24-14	24-16	24-50	24-57	24-63	24-76	24-82	24-86	24-137	24-139	24-141
	24-143	24-145	24-147	24-149	24-152	24-154	24-156	24-159	24-165	24-171	24-174	24-176	24-178	24-181
	24-183	24-185	24-188	24-190	24-193	24-195	24-198	24-200	24-210	24-226	24-228	24-230	24-236	24-238
	24-240	24-242	24-244	24-246	24-249	24-251	24-260	24-262	24-291	24-323	24-337	24-339	24-341	24-344
	24-346	24-348	24-350	24-352	24-355	24-357	24-359	24-361	24-364	24-366	24-368	24-371	24-373	24-375
	24-377	24-380	24-382	24-384	24-387	24-389	24-391	24-393	24-396	24-398	24-400	24-402	24-404	24-407
	24-409	24-411	24-413	24-415	24-417	24-419	24-422	28-152#						
EMS504	24-11	24-14	24-16	24-34	24-37	24-40	24-55	24-61	24-71	24-74	24-76	24-84	24-89	24-128
	24-198	24-200	24-202	24-204	24-206	24-214	24-216	24-218	24-268	24-277	24-279	24-287	24-289	24-294
	24-298	24-300	24-302	28-153#										
EMS505	24-7	28-154#												
EMS506	24-107	28-155#												
EMS507	24-109	28-156#												
EMS51	24-267	24-284	28-44#											
EMS510	24-291	28-157#												
EMS511	24-5	24-7	24-9	24-11	24-14	24-16	24-18	24-20	24-22	24-24	24-26	24-28	24-30	24-32
	24-34	24-37	24-40	24-42	24-45	24-48	24-50	24-53	24-55	24-57	24-59	24-61	24-63	24-65
	24-67	24-71	24-74	24-76	24-78	24-80	24-82	24-84	24-86	24-89	24-92	24-95	24-97	24-99
	24-102	24-105	24-107	24-109	24-111	24-114	24-117	24-119	24-121	24-123	24-125	24-128	24-130	24-132
	24-134	24-137	24-139	24-141	24-143	24-145	24-147	24-149	24-152	24-154	24-156	24-159	24-161	24-163
	24-165	24-167	24-169	24-171	24-174	24-176	24-178	24-181	24-183	24-185	24-188	24-190	24-193	24-195
	24-198	24-200	24-202	24-204	24-206	24-208	24-210	24-212	24-214	24-216	24-218	24-220	24-222	24-224
	24-226	24-228	24-230	24-232	24-234	24-236	24-238	24-240	24-242	24-244	24-246	24-249	24-251	24-253
	24-255	24-257	24-260	24-262	24-264	24-266	24-268	24-270	24-273	24-275	24-277	24-279	24-281	24-283
	24-285	24-287	24-289	24-291	24-294	24-296	24-298	24-300	24-302	24-304	24-307	24-309	24-311	24-313
	24-315	24-317	24-319	24-321	24-323	24-325	24-327	24-329	24-331	24-333	24-335	24-337	24-339	24-341
	24-344	24-346	24-348	24-350	24-352	24-355	24-357	24-359	24-361	24-364	24-366	24-368	24-371	24-373
	24-375	24-377	24-380	24-382	24-384	24-387	24-389	24-391	24-393	24-396	24-398	24-400	24-402	24-404
	24-407	24-409	24-411	24-413	24-415	24-417	24-419	24-422	28-158#					
EMS52	24-280	24-282	24-286	24-288	28-45#									
EMS53	24-290	24-412	28-46#											
EMS54	24-292	24-295	24-297	24-299	28-47#									
EMS55	24-301	28-48#												
EMS56	24-303	24-305	24-306	24-342	28-49#									
EMS57	24-308	28-51#												
EMS6	24-124	24-127	24-129	24-131	28-8#									
EMS60	24-308	28-52#												
EMS600	24-272	24-274	24-276	24-278	28-161#									
EMS601	24-280	28-162#												
EMS602	24-282	24-284	28-163#											
EMS603	24-320	28-164#												
EMS604	24-349	24-351	24-354	24-356	24-358	24-360	24-363	28-165#						
EMS605	24-365	24-367	24-370	24-372	24-374	24-376	24-379	28-166#						
EMS606	24-381	24-383	24-386	24-388	24-390	24-392	24-395	24-397	28-167#					
EMS607	24-401	24-403	24-406	24-408	24-410	24-412	24-414	24-416	24-418	24-421	28-168#			
EMS61	24-312	28-53#												
EMS62	24-314	24-316	24-318	24-320	28-54#									
EMS63	24-322	24-324	24-326	24-328	24-330	24-332	24-334	24-336	24-356	24-372	24-388	28-55#		
EMS64	24-338	28-56#												

EMS66	24-345	24-347	24-349	24-354	24-365	24-370	24-381	24-386	24-399	24-401	24-416	28-58#
EMS67	24-363	24-379	24-397	24-414	28-59#							
EMS7	24-126	24-127	28-9#									
EMS70	24-395	24-421	28-61#									
EMT1	7-3	24-3#										
EMT10	7-25	24-17#										
EMT100	7-193	24-142#										
EMT101	7-196	24-144#										
EMT102	7-199	24-146#										
EMT103	7-202	24-148#										
EMT104	7-205	24-151#										
EMT105	7-208	24-153#										
EMT106	7-211	24-155#										
EMT107	7-214	24-158#										
EMT11	7-28	24-19#										
EMT110	7-217	24-160#										
EMT111	7-220	24-162#										
EMT112	7-223	24-164#										
EMT113	7-226	24-166#										
EMT114	7-229	24-168#										
EMT115	7-232	24-170#										
EMT116	7-235	24-173#										
EMT117	7-238	24-175#										
EMT12	7-31	24-21#										
EMT120	7-241	24-177#										
EMT121	7-244	24-180#										
EMT122	7-247	24-182#										
EMT123	7-250	24-184#										
EMT124	7-253	24-187#										
EMT125	7-256	24-189#										
EMT126	7-259	24-192#										
EMT127	7-262	24-194#										
EMT13	7-34	24-23#										
EMT130	7-265	24-197#										
EMT131	7-268	24-199#										
EMT132	7-271	24-201#										
EMT133	7-274	24-203#										
EMT134	7-277	24-205#										
EMT135	7-280	24-207#										
EMT136	7-283	24-209#										
EMT137	7-286	24-211#										
EMT14	7-37	24-25#										
EMT140	7-289	24-213#										
EMT141	7-292	24-215#										
EMT142	7-295	24-217#										
EMT143	7-298	24-219#										
EMT144	7-301	24-221#										
EMT145	7-304	24-223#										
EMT146	7-307	24-225#										
EMT147	7-310	24-227#										
EMT15	7-40	24-27#										
EMT150	7-313	24-229#										
EMT151	7-316	24-231#										
EMT152	7-319	24-233#										
EMT153	7-322	24-235#										
EMT154	7-325	24-237#										

EMT15C 7-328 24-239#

J 8

SEQ 0306

EMT156	7-331	24-241#
EMT157	7-334	24-243#
EMT16	7-43	24-29#
EMT160	7-337	24-245#
EMT161	7-340	24-248#
EMT162	7-343	24-250#
EMT163	7-346	24-252#
EMT164	7-349	24-254#
EMT165	7-352	24-256#
EMT166	7-355	24-259#
EMT167	7-358	24-261#
EMT17	7-46	24-31#
EMT170	7-361	24-263#
EMT171	7-364	24-265#
EMT172	7-367	24-267#
EMT173	7-370	24-269#
EMT174	7-373	24-271#
EMT175	7-376	24-274#
EMT176	7-379	24-276#
EMT177	7-382	24-278#
EMT2	7-6	24-4#
EMT20	7-49	24-33#
EMT200	7-385	24-280#
EMT201	7-388	24-282#
EMT202	7-391	24-284#
EMT203	7-394	24-286#
EMT204	7-397	24-288#
EMT205	7-400	24-290#
EMT206	7-403	24-292#
EMT207	7-406	24-295#
EMT21	7-52	24-36#
EMT210	7-409	24-297#
EMT211	7-412	24-299#
EMT212	7-415	24-301#
EMT213	7-418	24-303#
EMT214	7-421	24-306#
EMT215	7-424	24-308#
EMT216	7-427	24-310#
EMT217	7-430	24-312#
EMT22	7-55	24-39#
EMT220	7-433	24-314#
EMT221	7-436	24-316#
EMT222	7-439	24-318#
EMT223	7-442	24-320#
EMT224	7-445	24-322#
EMT225	7-448	24-324#
EMT226	7-451	24-326#
EMT227	7-454	24-328#
EMT23	7-58	24-41#
EMT230	7-457	24-330#
EMT231	7-460	24-332#
EMT232	7-464	24-334#
EMT233	7-467	24-336#
EMT234	7-470	24-338#
EMT235	7-473	24-340#
EMT236	7-476	24-342#

EMT237 7-479 7-482 24-345#

L 8

SEQ 0308

EMT5 7-15 24-10#

N 8

SEQ 0310

19-10 19-10 19-10 19-10 19-10 19-10 19-10 19-10 19-10 19-10

SEQ 0312

IPCK3 4-966#

SEQ 0314

G 9

12-[40 12-[41 12-[60 12-[84 12-[85 12-\19 12-\28 12-\29 12-\60 12-\70 12-\71 12-\76 12-\80 12-\81

SEQ 0316

PRS 4-688# 15-63

SEQ 0318

12-412 12-467* 12-471 12-484* 12-488 12-503* 12-570* 12-571* 12-581* 12-582* 12-597* 12-598* 12-:84* 12-:86*

K 9

SEQ 0320

M 9

12-928* 12-929* 12-930 12-956* 12-957 12-:12 12-?06 12-?12* 12-?14 12-?41 12-?45 12-?76* 12-?78 12-?87

SEQ 0322

```
        12-a08   12-a12*  12-a13   12-a23*  12-a25   12-a39*  12-a41   12-B93*  12-D64*  12-D92*  12-E02*  12-E17*  12-E41*  12-E42*
        12-E73*  12-F09*  12-G29*  12-H79*  12-I55*  12-J85   12-J94*  12-J95   12-K10   12-K31   12-K38*  12-K39   12-K54   12-L56*
        12-M06*  12-N45*  12-N88*  12-P78*  12-S83*  12-V00*  12-V37*  12-V58*  12-V91*  12-Y56*  12-Y94*  12-Z32*  12-Z63*  12-[05*
        12-[62*  12-\23*  12-\64*  12-]05*  12-]39*  12-]84*  12-^45*  12- _14*  12- _55*  12- _99*  12-`34*  12-`95*  12-a55*  12-b27*
        12-c15*  12-c85*  12-d25*  12-e17*  12-e77*  12-f49*  12-g87*  12-h31*  12-i19*  12-i81*  12-j65*  12-l00*  16-18*   17-16*
RMER2I   6-0#    12-121*  12-125*  12-130   12-148*  12-152*  12-185*  12-192*  12-198*  12-204   12-234*  12-240*  12-247   12-274*
        12-299   12-392*  12-398*  12-399   12-414*  12-420*  12-421   12-472*  12-475*  12-476   12-506*  12-512*  12-513   12-880*
        12-883   12-891   12-899   12-909*  12-912   12-920   12-930*  12-932   12-940   12-:12*  12-:17
RMER20   6-0#
RMHR     4-909#  12-371*  12-410*  12-564   12-572   12-583   12-599   12-:27*  12-:63*  12-:66
RMHRI    6-0#    12-:66*
RMHRO    6-0#    12-:04*  12-:06*  12-:27   12-:39   12-:46*  12-:54*  12-:60*  12-:63   12-:75   30-6
RMLA     4-903#  12-409*
RMLAI    6-0#
RMLAO    6-0#
RMMR1    4-904#  12-144*  12-149   12-370*  12-=20   12-=25   12-=33*  12-=34*  12-=35   12-=44*  12-=45*  12-=46   12-=59*  12-=60*
        12-=61   12-=83*  12-=93*  12-=94*  12->14*  12->15*  12->49*  12->58*  12->59*  12->75*  12->76*  12-?11*  12-?39*  12-?40*
        12-?75*  12-?77*  12-a11*  12-a22*  12-a24*  12-a38*  12-a40*  12-a72*  12-a73*  12-a83*  12-a99*  12-A00*  12-A28   12-A33
        12-A41*  12-A42*  12-A43*  12-A44   12-A52*  12-A53*  12-A54   12-A69*  12-A70*  12-A71   12-B10   12-B18   12-B91*  12-B97*
        12-C02*  12-C03*  12-C37*  12-C38*  12-C42*  12-C43*  12-C99*  12-D00*  12-D04*  12-D05*  12-D62*  12-D66*  12-D67*  12-D90*
        12-E15*  12-E39*  12-E70*  12-E71*  12-F06*  12-F07*  12-F33*  12-F34*  12-G26*  12-G27*  12-G64*  12-G65*  12-H76*  12-H77*
        12-I0?*  12-I02*  12-I46   12-I52*  12-I53*  12-I58   12-I82*  12-I83*  12-I84   12-J91*  12-J92*  12-K08*  12-K09*  12-K35*
        12-K36*  12-K53*  12-K72*  12-K73*  12-K93*  12-K94*  12-L14*  12-L15*  12-L18   12-L25   12-L53*  12-L54*  12-L70   12-L74
        12-M0?*  12-M04*  12-M21*  12-M22*  12-M63*  12-M67*  12-M68*  12-N01*  12-N09*  12-N10*  12-N15   12-N25   12-N42*  12-N43*
        12-N56*  12-N57*  12-N85*  12-N86*  12-N94*  12-N95*  12-O13*  12-O14*  12-O23*  12-O24*  12-O58*  12-O61*  12-O62*  12-O91*
        12-O94*  12-O95*  12-P54*  12-P56*  12-P57*  12-P70   12-P75*  12-P76*  12-P85   12-Q23*  12-Q33*  12-Q34*  12-R01*  12-R08*
        12-R09*  12-R50*  12-R54*  12-R55*  12-R95*  12-R98*  12-R99*  12-S33*  12-S45*  12-S46*  12-S51*  12-S52*  12-S80*  12-S81*
        12-T17   12-U66*  12-U69*  12-U70*  12-U98*  12-V01*  12-V35*  12-V38*  12-V56*  12-V62*  12-V74*  12-V89*  12-W13*  12-W64*
        12-W70*  12-W71*  12-X11*  12-X19*  12-X21*  12-X89   12-X93   12-Y04*  12-Y07*  12-Y08*  12-Y15   12-Y23   12-Y54*  12-Y61*
        12-Y62*  12-Y67*  12-Y69*  12-Y70*  12-Y92*  12-Z00*  12-Z01*  12-Z06*  12-Z10*  12-Z11*  12-Z30*  12-Z38*  12-Z39*  12-Z61*
        12-Z68*  12-Z69*  12-Z74*  12-Z75*  12-Z80*  12-Z81*  12-[03*  12-[11*  12-[12*  12-[17*  12-[22*  12-[23*  12-[36*  12-[40*
        12-[41*  12-[60*  12-[84*  12-[85*  12-\19*  12-\28*  12-\29*  12-\34*  12-\36*  12-\37*  12-\60*  12-\70*  12-\71*  12-\76*
        12-\80*  12-\81*  12-]01*  12-]11*  12-]12*  12-]35*  12-]44*  12-]45*  12-]50*  12-]51*  12-]56*  12-]57*  12-]80*  12-]90*
        12-]91*  12-]96*  12-^01*  12-^02*  12-^15*  12-^19*  12-^20*  12-^41*  12-^68*  12-^69*  12- _10*  12- _19*  12- _20*  12- _25*
        12- _27*  12- _28*  12- _51*  12- _61*  12- _62*  12- _67*  12- _71*  12- _72*  12- _95*  12-`05*  12-`06*  12-`30*  12-`39*  12-`40*
        12-`46*  12-`49*  12-`54*  12-`55*  12-`61*  12-`62*  12-`63*  12-`67*  12-`68*  12-`91*  12-a01*  12-a02*  12-a07*  12-a12*
        12-a13*  12-a26*  12-a30*  12-a31*  12-a51*  12-a60*  12-a61*  12-a67*  12-a70*  12-a75*  12-a76*  12-a83   12-a86*  12-a87*
        12-a93   12-a97*  12-b01*  12-b02*  12-b23*  12-b39*  12-b40*  12-b65*  12-b66*  12-c13*  12-c22*  12-c23*  12-c28*  12-c29*
        12-c33*  12-c34*  12-c41*  12-c42*  12-c43   12-c51   12-c57*  12-c83*  12-c92*  12-c93*  12-c98*  12-d02*  12-d03*  12-d23*
        12-d32*  12-d33*  12-d38*  12-d42*  12-d43*  12-d50*  12-d51*  12-d52   12-d60   12-d68*  12-d69*  12-d76*  12-d77*  12-d83
        12-d88   12-d95*  12-d96*  12-e15*  12-e24*  12-e25*  12-e30   12-e35   12-e42*  12-e43*  12-e75*  12-e84   12-e92   12-f02*
        12-f03*  12-f08*  12-f12*  12-f13*  12-f20*  12-f21*  12-f22   12-f30   12-f47*  12-f56   12-f64   12-f71*  12-f72*  12-f77*
        12-f78*  12-f89*  12-f90*  12-g06*  12-g07*  12-g85*  12-g94   12-h02   12-h09*  12-h10*  12-h29*  12-h38   12-h46   12-h53*
        12-h54*  12-h59*  12-h63*  12-h64*  12-h70*  12-h71*  12-h76*  12-h80*  12-h81*  12-h88*  12-h89*  12-h90   12-h98   12-i17*
        12-i24   12-i32   12-i39*  12-i40*  12-i46*  12-i47*  12-i51*  12-i52*  12-i79*  12-i86   12-i94   12-j01*  12-j02*  12-j08*
        12-j09*  12-j13*  12-j14*  12-j19*  12-j23*  12-j24*  12-j29*  12-j33*  12-j34*  12-j41*  12-j42*  12-j43   12-j51   12-j63*
        12-j72   12-j80   12-j87*  12-j88*  12-j94*  12-j98*  12-j99*  12-k06*  12-k07*  12-k08   12-k16   12-k21*  12-k25*  12-k26*
        12-k33*  12-k34*  12-k37   12-k45   12-k98*  12-l05   12-l13   12-l38*  12-l39*  12-l51*  12-l52*  12-l94*  12-l95*  12-m08*
        12-m09*  12-m27*  12-m28*  16-15*   16-16*   17-14*
RMMR1I   6-0#    12-149*  12-154   12-B18*  12-B19   12-B38
RMMR10   6-0#
RMMR2    4-910#  12-389*  12-F00   12-F13   12-F39   12-T79   12-T90   12-T94   12-U08   12-U19   12-U23   12-U37   12-U48   12-U52
        12-U76   12-U90   12-[67   12-[73   12-^50   12-^57   12-b47   12-b54   12-e46   12-e62   12-f98   12-g30   12-l21   12-l44
        12-l56   12-m01   12-m13   12-m33
RMMR2I   6-0#
RMMR20   6-0#
```

RMOF	4-907*	12-81*	12-88	12-168*	12-175*	12-183	12-217*	12-223*	12-231	12-264*	12-271	12-385*	12-391	12-407*

D 10

12-K28 12-K62 12-L94 12-M54 12-M90 12-N38 12-N77 12-007 12-047 12-073 12-P17 12-P38 12-P65 12-Q09

SEQ 0326

12-f99 12-L32 12-L70

SEQ 0328

TST52 12-K62# 19-6

H 10

SEQ 0330

WCE 4-937# 21-77 21-78

J 10

SEQ 0332

```
WCEHI    4-962#
WCELO    4-963#
WCF      4-780#   4-787   12-697   12-726
WCH      4-722#  12-F77   12-Q66
WD       4-725#  12-F85   12-Q72   12-S40   12-X49   12-X53   12-Y05
WH       4-726#  12-F93   12-Q63
WLE      4-774#   4-787   12-X27   12-X51   21-81    21-82
WRL      4-760#  12->51   12->61   12->63   12->78   12->82
XNUDC    4-864#  12-97    12-191   12-224   12-239   12-290   12-293   12-376   12-437   12-459   12-509   12-:92   12-;54   12-;64
        12-;74   12-;92   12-N66   12-039
XNUER2   4-889#  12-125   12-192   12-226   12-240   12-300   12-303   12-398   12-420   12-475   12-512   12-884   12-933   12-941
        12-942   12-965   12-E47
XNUOF    4-859#  12-81    12-99    12-190   12-223   12-238   12-284   12-287   12-395   12-417   12-456   12-507   12-810   12-818
        12-831   12-839   12-840   12-856   12-N69   12-001
XSIZ     8-67#    9-123    9-153
XXDP     6-0#     8-30*    8-33*    8-34     8-36*    8-41     8-52     8-77     8-79
ZEROS   23-38#
```

```
$$CMRE   4-993#
$$CMTM   4-993#  5-0      5-0      5-0      5-0      5-0
$$ESCA   4-688#
$$NEWT   4-688#  12-1     12-73    12-106   12-137   12-159   12-329   12-521   12-548   12-559   12-612   12-679   12-783   12-796
        12-866   12-975   12-998   12-:51   12-:79   12-:99   12-;43   12-<05   12-=00   12-=17   12-=75   12->40   12-?01   12-@03
        12-@64   12-A24   12-B06   12-B74   12-C76   12-D29   12-D82   12-E62   12-E97   12-G18   12-H68   12-I43   12-J82   12-K28
        12-K62   12-L94   12-M54   12-M90   12-N38   12-N77   12-O07   12-O47   12-O73   12-P17   12-P38   12-P65   12-Q09   12-Q86
        12-R34   12-R84   12-S21   12-S71   12-T28   12-T46   12-T69   12-T98   12-U27   12-U57   12-U94   12-V32   12-V52   12-V85
        12-W06   12-W55   12-X00   12-X70   12-Y47   12-\09   12-_00   12-c06   12-c74   12-g76   12-k50
$$SET    19-10    19-10    19-10    19-10    19-10    19-10    19-T0    19-10    19-10    19-T0    19-10    19-10    19-10    19-10#
$$SETM   8-19     8-19#
$$SKIP   4-688#
.$ACT1   4-680#   4-988
.$APTB   4-680#   5-0      5-0#
.$APTH   4-680#   4-991
.$APTY   4-680#   19-12
.$CATC   4-67?#   4-986
.$CMTA   4-677#   4-993
.$EOP    4-677#   13-20
.$ERRO   4-677#   19-7
.$ERRT   4-677#
.$POWE   4-679#   19-11
.$RDDE   4-678#
.$RDOC   4-678#   19-9
.$READ   4-678#   19-8
.$SAVE   4-679#   19-1
.$SCOP   4-677#   19-6
.$SIZE   4-679#
.$TRAP   4-679#   19-10
.$TYPB   4-678#   19-2
.$TYPD   4-678#   19-3
.$TYPE   4-677#   19-5
.$TYPO   4-678#   19-4
.EQUAT   4-676#   4-688
.HEADE   4-676#   4-684
.SETUP   4-676#   4-984
.SWRHI   4-676#   4-685
.SWRLO   4-676#   4-685#   4-686
CLEAR    4-491#   12-7     12-28    12-75    12-108   12-116   12-139   12-162   12-212   12-260   12-367   12-383   12-405   12-427
        12-444   12-466   12-482   12-497   12-550   12-561   12-615   12-620   12-686   12-763   12-785   12-878   12-954   12-977
        12-:10   12-:62   12-:81   12-;02   12-;46   12-<12   12-=02   12-=08   12-=24   12-=77   12->46   12-?03   12-@05   12-@66
        12-A32   12-B15   12-B90   12-C34   12-C85   12-C95   12-D45   12-D84   12-E14   12-E38   12-E69   12-F05   12-G25   12-H75
        12-I51   12-J90   12-K34   12-L52   12-M02   12-N41   12-N84   12-O12   12-P74   12-S79   12-T31   12-T49   12-U97   12-V34
        12-V55   12-V88   12-W12   12-X88   16-14
CLKOFF   4-600#   12-K50   12-P91   12-P96   12-Y20   12-Y30   12-c64   12-c70   12-e89   12-e95   12-f61   12-f67   12-g99   12-h05
        12-h43   12-h49   12-i29   12-i35   12-i91   12-i97   12-j77   12-j83   12-l10   12-l16
CLKON    4-591#   12-K47   12-P83   12-Y14   12-c58   12-e83   12-f55   12-g93   12-h37   12-i23   12-i85   12-j71   12-l04
CLKSNC   4-660#
COMMEN   4-688#
ENBSCH   4-626#
ENDCOM   4-688#
ERR      4-559#   7-3      7-6      7-9      7-12     7-15     7-18     7-22     7-25     7-28     7-31     7-34     7-37     7-40
        7-43     7-46     7-49     7-52     7-55     7-58     7-61     7-64     7-67     7-70     7-73     7-76     7-79     7-82
        7-85     7-88     7-91     7-94     7-97     7-100    7-103    7-106    7-109    7-112    7-115    7-118    7-121    7-124
        7-127    7-130    7-133    7-136    7-139    7-142    7-145    7-148    7-151    7-154    7-157    7-160    7-163    7-166
        7-169    7-172    7-175    7-178    7-181    7-184    7-187    7-190    7-193    7-196    7-199    7-202    7-205    7-208
```

GETMR¹	4-254#	12-149	12-=25	12-=35	12-=46	12-=61	12-A33	N 10 12-A44	12-A54	12-A71	12-B18	12-158	12-184	12-L18 SEQ 0336

```
PUTDS    4-354#  12-469  12-501
PUTDT    4-402#  12-448
PUTEC1   4-418#  12-450
PUTEC2   4-426#  12-411
PUTER1   4-362#  12-112  12-142  12-167  12-174  12-225  12-266  12-368  12-428  12-445  12-483  12-502  12-684  12-714
        12-715   12-745  12-746  12-764  12-?13  12-B92  12-D63  12-D91  12-E01  12-E16  12-E18  12-E40  12-E72  12-F08
        12-G28   12-H78  12-I54  12-J93  12-K37  12-K69  12-K90  12-L11  12-L55  12-M05  12-N44  12-N87  12-015  12-016
        12-P77   12-S82  12-U99  12-V36  12-V57  12-V90  12-V95  12-W14  12-Y13  12-Y55  12-Y93  12-Z31  12-Z62  12-[04
        12-[61   12-\22  12-\63  12-]04  12-]38  12-]83  12-^44  12-_13  12-_54  12-_98  12-`33  12-`94  12-a54  12-b26
        12-c14   12-c84  12-d24  12-e16  12-e76  12-f48  12-g86  12-h30  12-i18  12-i80  12-j64  12-k99  16-17   17-15
PUTER2   4-394#  12-113  12-143  12-169  12-177  12-226  12-267  12-386  12-408  12-449  12-468  12-486  12-500  12-876
        12-907   12-908  12-928  12-929  12-956  12-?12  12-?76  12-a12  12-a23  12-a39  12-B93  12-D64  12-D92  12-E02
        12-E17   12-E41  12-E42  12-E73  12-F09  12-G29  12-H79  12-I55  12-J94  12-K38  12-L56  12-M06  12-N45  12-N88
        12-P78   12-S83  12-V00  12-V37  12-V58  12-V91  12-Y56  12-Y94  12-Z32  12-Z63  12-[05  12-[62  12-\23  12-\64
        12-]05   12-]39  12-]84  12-^45  12-_14  12-_55  12-_99  12-`34  12-`95  12-a55  12-b27  12-c15  12-c85  12-d25
        12-e17   12-e77  12-f49  12-g87  12-h31  12-i19  12-i81  12-j65  12-l00  16-18   17-16
PUTHR    4-434#  12-371  12-410  12-:27  12-:63
PUTLA    4-386#  12-409
PUTMR1   4-442#  12-144  12-370  12-=33  12-=34  12-=44  12-=45  12-=59  12-=60  12-=83  12-=93  12-=94  12->14  12->15
        12->49   12->58  12->59  12->75  12->76  12-?11  12-?39  12-?40  12-?75  12-?77  12-a11  12-a22  12-a24  12-a38
        12-a40   12-a72  12-a73  12-a83  12-a99  12-A00  12-A41  12-A42  12-A43  12-A52  12-A53  12-A69  12-A70  12-B91
        12-B97   12-C02  12-C03  12-C37  12-C38  12-C42  12-C43  12-C99  12-D00  12-D04  12-D05  12-D62  12-D66  12-D67
        12-D90   12-E15  12-E39  12-E70  12-E71  12-F06  12-F07  12-F33  12-F34  12-G26  12-G27  12-G64  12-G65  12-H76
        12-H77   12-I01  12-I02  12-I52  12-I53  12-I82  12-I83  12-J91  12-J92  12-K08  12-K09  12-K35  12-K36  12-K53
        12-K72   12-K73  12-K93  12-K94  12-L14  12-L15  12-L53  12-L54  12-M03  12-M04  12-M21  12-M22  12-M63  12-M67
        12-M68   12-N01  12-N09  12-N10  12-N42  12-N43  12-N56  12-N57  12-N85  12-N86  12-N94  12-N95  12-013  12-014
        12-023   12-024  12-058  12-061  12-062  12-091  12-094  12-095  12-P54  12-P56  12-P57  12-P75  12-P76  12-Q23
        12-Q33   12-Q34  12-R01  12-R08  12-R09  12-R50  12-R54  12-R55  12-R95  12-R98  12-R99  12-S33  12-S45  12-S46
        12-S51   12-S52  12-S80  12-S81  12-U66  12-U69  12-U70  12-U98  12-V01  12-V35  12-V38  12-V56  12-V62  12-V74
        12-V89   12-W13  12-W64  12-W70  12-W71  12-X11  12-X19  12-X21  12-Y04  12-Y07  12-Y08  12-Y54  12-Y61  12-Y62
        12-Y67   12-Y69  12-Y70  12-Y92  12-Z00  12-Z01  12-Z06  12-Z10  12-Z11  12-Z30  12-Z38  12-Z39  12-Z61  12-Z68
        12-Z69   12-Z74  12-Z75  12-Z80  12-Z81  12-[03  12-[11  12-[12  12-[17  12-[22  12-[23  12-[36  12-[40  12-[41
        12-[60   12-[84  12-[85  12-\19  12-\28  12-\29  12-\34  12-\36  12-\37  12-\60  12-\70  12-\71  12-\76  12-\80
        12-\81   12-]01  12-]11  12-]12  12-]35  12-]44  12-]45  12-]50  12-]51  12-]56  12-]57  12-]80  12-]90  12-]91
        12-]96   12-^01  12-^02  12-^15  12-^19  12-^20  12-^41  12-^68  12-^69  12-_10  12-_19  12-_20  12-_25  12-_27
        12-_28   12-_51  12-_61  12-_62  12-_67  12-_71  12-_72  12-_95  12-`05  12-`06  12-`30  12-`39  12-`40  12-`46
        12-`49   12-`54  12-`55  12-`61  12-`62  12-`63  12-`67  12-`68  12-`91  12-a01  12-a02  12-a07  12-a12  12-a13
        12-a26   12-a30  12-a31  12-a51  12-a60  12-a61  12-a67  12-a70  12-a75  12-a76  12-a86  12-a87  12-a97  12-b01
        12-b02   12-b23  12-b39  12-b40  12-b65  12-b66  12-c13  12-c22  12-c23  12-c28  12-c29  12-c33  12-c34  12-c41
        12-c42   12-c57  12-c83  12-c92  12-c93  12-c98  12-d02  12-d03  12-d23  12-d32  12-d33  12-d38  12-d42  12-d43
        12-d50   12-d51  12-d68  12-d69  12-d76  12-d77  12-d95  12-d96  12-e15  12-e24  12-e25  12-e42  12-e43  12-e75
        12-f02   12-f03  12-f08  12-f12  12-f13  12-f20  12-f21  12-f47  12-f71  12-f72  12-f77  12-f78  12-f89  12-f90
        12-g06   12-g07  12-g85  12-h09  12-h10  12-h29  12-h53  12-h54  12-h59  12-h63  12-h64  12-h70  12-h71  12-h76
        12-h80   12-h81  12-h88  12-h89  12-i17  12-i39  12-i40  12-i46  12-i47  12-i51  12-i52  12-i79  12-j01  12-j02
        12-j08   12-j09  12-j13  12-j14  12-j19  12-j23  12-j24  12-j29  12-j33  12-j34  12-j41  12-j42  12-j63  12-j87
        12-j88   12-j94  12-j98  12-j99  12-k06  12-k07  12-k21  12-k25  12-k26  12-k33  12-k34  12-k98  12-l38  12-l39
        12-l51   12-l52  12-l94  12-l95  12-m08  12-m09  12-m27  12-m28  16-15   16-16   17-14
PUTMR2   4-450#  12-389
PUTOF    4-458#  12-81   12-168  12-175  12-217  12-223  12-264  12-385  12-407  12-446  12-470  12-498  12-786  12-804
        12-805   12-325  12-826  12-852  12-853  12-879  12-955  12-B16  12-B95  12-C35  12-C96  12-D48  12-E43  12-N50
        12-N89   12-083  12-P50  12-Q20  12-Q98  12-R47  12-S34  12-k88
PUTSN    4-410#  12-388  12-431
PUTWC    4-323#  12-S38  12-k94
PUTX     4-307#
REPORT   4-688#
RGBFMC   4-20#   6-0     6-0
```