ECCAA-1.8

RH750 MB
ADAPTER TEST

EP-ECCAA-DL-1.8
1 OF 2 JUL 1985
COPYRIGHT© 1979-85

digital
MADE IN USA

ECCAA-1.8

RH750 MB
ADAPTER TEST

EP-ECCAA-DL-1.8
2 OF 2 JUL 1985
COPYRIGHT© 1979-85

digital
MADE IN USA

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

PRODUCT ID: ZZ-ECCAA-1.8
PRODUCT TITLE: ECCAA RH750 MB ADAPTER TEST
DECO/DEPO: 1.8
DATE: Feb. 8, 1985
DEPARTMENT: BASE SYSTEMS DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1979, 1983, 1984
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754

This software is furnished under a license for use only on a single computer system and may be copied only with the inclusion of the above copyright notice. This software, or any other copies thereof, may not be provided or otherwise made available to any other person except for use on such system and to one who agrees to these license terms. Title to and ownership of the software shall at all times remain in DEC.

The information in this software is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation.

DEC assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DEC.

Table of Contents

CONTENTS

1	ABSTRACT	3
2	REQUIREMENTS	3
2.1	HARDWARE	3
2.2	SOFTWARE REQUIREMENTS	3
2.3	PREREQUISITES	3
3	OPERATING INSTRUCTIONS	3
4	FUNCTIONAL DESCRIPTION	4
4.1	PROGRAM OVERVIEW	4
4.2	PROGRAM SIZE	4
4.3	PROGRAM RUN TIMES	5
4.4	RUN-TIME DYNAMICS	5
4.5	EVENT FLAGS	5
4.6	FAULT DETECTION	5
4.7	PERFORMANCE DURING HARDWARE FAILURES	5
4.8	SEQUENCE OF TESTING MULTIPLE UNITS	5
4.9	PROGRAM APPLICATIONS	5
4.10	PROGRAM SET UP AND RUN WITH APT AND APT-RD	6
4.11	SPECIAL NOTES SECTION	6
4.12	TEST AND SUBTEST DESCRIPTIONS	6
5	MAINTENANCE HISTORY	16

1 ABSTRACT

This program provides functional verification and repair level testing of an RH-750. The program is macro level three, which operates with the Diagnostic Supervisor. Where possible, failures will be isolated to the gate array.

2 REQUIREMENTS

2.1 HARDWARE

VAX 11/750 CPU 256 KB of memory Console Terminal Suitable load device such as TU58 or Automated Product Test (APT). One to three RH-750's, each one connected to a Massbus Controller Tester (RH11-TB). RH11-TB is necessary for Test 20. If an RH11-TB is not on the system, the program will skip Test 20, however this will prohibit checking the integrity of all Massbus signal lines (to provide cable testing).

2.2 SOFTWARE REQUIREMENTS

None

2.3 PREREQUISITES

The EVKAA VAX Hardcore Instruction Test, ECKAL Cache/TB Test, and ECKAM Memory test must have been run successfully prior to the execution of the program.

3 OPERATING INSTRUCTIONS

1. Boot in the Diagnostic Supervisor
2. Load ECCAA.EXE
3. ATTACH RH750 CMI RH0 5
4. ATTACH MBE RH0 MB7
5. SELECT RH0,MB7 and SET TRACE
6. START

4 FUNCTIONAL DESCRIPTION

4.1 PROGRAM OVERVIEW

The program consists of 20 tests. They are as follows:

TEST 1: REGISTER AND INTERNAL BUS VALIDATION TEST

TEST 2: INITIALIZATION TEST

TEST 3: CONTROL REGISTER SA0 TEST

TEST 4: DIAGNOSTIC REGISTER TEST

TEST 5: VIRTUAL ADDRESS REGISTER SA0 AND LOADING TEST

TEST 6: MAP REGISTER SA0 TEST

TEST 7: EXTERNAL REGISTER DATA PATH TEST

TEST 8: CAR BUS FUNCTION TEST

TEST 9: MBDIB DIAGNOSTIC REGISTER TEST

TEST 10: STATUS REGISTER STATIC SA0 TEST

TEST 11: MAPPING FUNCTION TEST

TEST 12: STATUS REGISTER XFER BITS TEST

TEST 13: BLOCK TRANSFER TEST

TEST 14: BYTE COUNTER TEST

TEST 15: FUNCTIONAL READ/WRITE TEST

TEST 16: SILO TEST

TEST 17: INTERRUPT TEST

TEST 18: ATTENTION SUMMARY REGISTER TEST

TEST 19: IGNORE BYTE COUNTER TEST

TEST 20: RH11-TB TEST

4.2 PROGRAM SIZE

52 Kilobytes

4.3 PROGRAM RUN TIMES

One complete pass with Trace set runs less than one-minute.

4.4 RUN-TIME DYNAMICS

None

4.5 EVENT FLAGS

Not Applicable

4.6 FAULT DETECTION

The hardware error detecting mechanism is compare and branch. The reporting mechanism is the Diagnostic Supervisor. Due to the inability to force parity errors at the Macro level, one-hundred percent error detection cannot be guaranteed. However, ninety-seven percent of the stuck-at-zero(S-A-0) and stuck-at-one(S-A-1) faults should be detected.

4.7 PERFORMANCE DURING HARDWARE FAILURES

On the occurence of an unexpected exception or interrupt, the program will print a message.

On a power failure the program should be restarted.

4.8 SEQUENCE OF TESTING MULTIPLE UNITS

The program tests every RH750 selected (in the order that they were selected) for each pass of the program.

4.9 PROGRAM APPLICATIONS

MANUFACTURING

Manufacturing can use the program for gate array and module quick-verification. It may also prove useful for environmental (thermal) testing.

FIELD SERVICE

Field Service can use this program for verification of customer installation, fault isolation, repair, repair verification, preventive maintenance, and verification of Engineering Change Order (ECO) installation.

CUSTOMER

The customer can use the program to provide verification of self-installation, preventive maintenance, and fault detection and isolation.

4.10 PROGRAM SET UP AND RUN WITH APT AND APT-RD

Standard Diagnostic Supervisor Interface

4.11 SPECIAL NOTES SECTION

Massbus Excerciser RH11TB

Whenever the RH11TB is connected to an RH750 and CR<03> Maintenance Mode is asserted, SR<16> Attention will always be asserted. To handle this situation, a subroutine called RH11TBPRES is called at the beginning of Tests 7-19 to ascertain whether or not an RH11TB is present. If present a mask bit is set in a location labeled RH11TBMSK and every time the Status Register results are to be checked, the mask bit location is set into the expected results register R4. If an RH11TB is not present, the subroutine will clear the mask location, and the Attention bit will not be expected.

4.12 TEST AND SUBTEST DESCRIPTIONS

TEST 1: REGISTER AND INTERNAL BUS VALIDATION TEST

This test verifies that the Internal Registers, Map Registers, and MBA Register Space exist, and that the Internal Bus (IBUS) is capable of transferring zeroes and ones in all bit

positions.

Subtest one performs a read of each Internal Register to verify its existence. The starting address for the Internal Registers is F28000 (RH0). Unlike the VAX11/780, the Map Selected Register (F28018) is not implemented ti the RH750 and is therefore bypassed.

Subtest two performs a read of each of the 256 Map Registers starting at address F28800.

Subtest three performs a read of all External Registers in the MBA Address space to verify their existence. The first External Address for Drive 0 is F28400.

Subtest four verifies that the Internal Bus is capable of transferring zeroes and ones in all bit positions. This is accomplished by reading the Configuration Register after performing an INIT. This checks for any SA1 conditions. All ones are then written into the VAR and DR to check that the IBUS has no SA0 faults.

TEST 2: INITIALIZATION TEST

This test initializes the RH750 and checks that all applicable Internal and Map Registers have no SA1 fault. Internal Registers that are not cleared by INIT are first written inot and then cleard and checked for SA0 faults.

TEST 3: CONTROL REGISTER SA0 TEST

This test checks that each bit in the Control Register is not SA0.

Subtest one checks all bits for a SA1 condition.

Subtest two checks that each individual bit in the CR is not SA0.

TEST 4: DIAGNOSTIC REGISTER TEST

This test first checks the Diagnostic Register for any SA1 faults with no RH750 activity and in the Maintenance Mode. It then writes each bit individually and checks for SA0 faults.

TEST 5: VIRTUAL ADDRESS REGISTER SA0 AND LOADING TEST

This test is comprised of two subtests.

Subtest one checks for SA0 faults in the VAR by floating a one from bit zero through bit twenty.

Subtest two writes and checks unique patterns into the VAR.

TEST 6: MAP REGISTER SA0 TEST

This test is comprised of three subtests.

Subtest one checks each Map Register for SA0 faults by writing and reading an all ones pattern.

Subtest two floats a one across each bit position in each Map Register.

Subtest three checks for dual addressing faults by writing a unique pattern in a single cell and checking that the pattern is not written into any other cell.

TEST 7: EXTERNAL REGISTER DATA PATH TEST

This test checks the Massbus Control path for SA0 and SA1 conditions. It also checks for Control path Parity Errors. This is accomplished by writting zeroes onto the Control path and verifying no SA1 faults, by floating a one in each bit position, and by writting an incrementing pattern onto the Control path and checking for Parity Errors.

TEST 8: CAR BUS FUNCTION TEST

This test is comprised of four subtests, checking the ability to perform a Read Command and then verifying the CMI Bus Fucntion bits, CAR<27:25>.

Subtest one first verifies that a Read Command can successfully executed while in the Maintenance Mode and that MASSRUN DR<19>, can be set and cleared.

Subtest two checks for a CMI Read Command by performing a WRITE-TO-DRIVE function and then reading the CAR and checking for the correct bus function code (0).

Subtest three performs a Write Check function (CMI read), and checks the CAR for the correct code (0).

Subtest four performs Read-From-Drive function (CMI write) and then checks the CAR for the correct code (4).

TEST 9: MBDIB DIAGNOSTIC REGISTER TEST

This test checks the integrity of the RH750's data paths by writing and reading various patters. The test is comprised of eight subtests, each one using the following patterns, with a 512 byte transfer except subtest 2.

Subtest 1: all zeroes Subtest 2: floating zero (eight byte transfer) Subtest 3: all ones Subtest 4: floating one Subtest 5: floating zero Subtest 6: F00F Subtest 7: A55A

TEST 10: STATUS REGISTER STATIC SA0 TEST

This test checks the following bits for SA0 conditions:

SR<31> DT_BUSY
SR<12> DT_ABORT
SR<16> ATTENTION
SR<18> NON-EXISTENT DRIVE
SR<08> MISSED TRANSFER
SR<11> DATA TRANSFER LATE
SR<19> PROGRAM ERROR

Each bit is tested by forcing the RH750 into a state in which an expected status condition should be present. If the state condition is not as expected, then an error is reported with the expected state and the received state, and the logical difference. ,b Note that R4 will indicate the expected state and R3 will contain the received state.

TEST 11: MAPPING FUNCTION TEST

This test verifies that the Map Register selection function is correctly selecting the Map Register pointed at by the Virtual Address Register. Parity associated with the Map Registers is also tested. Each Map Register is set equal to its ones complement modulo 256. For example, Map Register 0 equals 255, Map register 1 equals 254, ...Map Register 255 equals 0.

The following register conventions are used in this test:

R2 = RH750 Address
R3 = Map Register actually selected
R4 = Map Register that should have been selected
R6 = Value that should have been in the Map Register not including the V-bit
R7 = Received value of Map Register
R8 = Physical Address expected
R9 = Actual Physical Address selected

TEST 12: STATUS REGISTER XFER BITS TEST

This test verifies that the following Status Register bits are not SA0:

SR<06> MDPE Massbus Data Parity Error SR<17> MCPE Massbus Control Parity Error SR<05> MAPPE Page Frame Map Parity Error SR<04> INVMAP Invalid Map SR<09> WCK LWR ERR Write Check Lower Error SR<10> WCK UP ERR Write Check Upper Error SR<01> NRSTAT No Response Status SR<29> CRD Corrected Read Data SR<03> ERR STAT Error Status SR<14> SPE Silo Parity Error SR<07> MBEXC Massbus Exception

In addition, this test checks bit<12> of the RH750 address is zero, that MCPE will not set when reading the Attention Summary Register, and that SR<31:16> is supplied when reading any external register.

Subtest 1 checks that the RH750 will not respond with address bit<12> set.

Subtest 2 Checks that MDPE sets by reading data from the Massbus to memory with the Invert Massbus Data Parity bit set in the Diagnostic Register.

Subtest 3 checks that the Massbus Control Parity Error is forced by setting Invert Massbus Control Parity and reading a Drive Register while in the Maintenance Mode.

Subtest 4 checks that a Map Parity Error is forced by setting the Invert Map Parity bit and executing a transfer.

Subtest 5 checks that an Invalid Map Error is detected by transferring data to the Massbus using an Invalid Map.

Subtest 6 checks that Write Check Error Low byte is forced by writing data onto the Massbus while in the Maintenance Mode and then doing a Write Check with a different data pattern in memory.

Subtest 7 checks that Write Check Error High byte is forced by writing data onto the Massubs while in the Maintenance Mode and then doing a Write Check with a different data pattern in memory.

Subtest 8 checks that NR Status is forced by attempting to transfer data from non-existent memory.

Subtest 9 checks that the Corrected Read Data Bit, SR<29>, is not SA0. This is accomplished by setting up the memory controller to force a single bit error in a longword address.

A W-T-D command is then issued from the RH750 to that memory address containing the CRD.

Subtest 10 checks that the Error Status Bit, SR<03>, is not SA0. This is accomplished by setting up the memory controller to force a double-bit error in a longword address. A W-T-D command is then issued from the RH750 to read that memory address containing the double-bit error.

Subtest 11 checks that the RH750 will not set MCPE when reading the Attention Summary Register. This is accomplished by Inverting Massbus Control Parity Generation, reading the ASR, and then checking

Subtest 12 checks that the RH750 will supply Status Register<31:16>, when reading any External Register. This is accomplished by forcing the Attention bit, SR<16>, and the Massbus Control Parity bit,SR<17>, to be set while the remainder of the SR is zero, then a Drive Register is read and CMI <31:16> is compared to SR<31:16>.

Subtest 13 performs a dynamic test of SR<07>, MB Exception. This is accomplished by issuing a setting Block Sending Command and then executing a CMI transaction which should result in a Massbus Exception. that an MCPE did not occur.

Subtest 14 checks the Silo Parity Error bit, SR<14>, by setting the Invert Silo Parity Generation bit, which is DR<22>, and then executing a W-T-D function.

TEST 13: BLOCK TRANSFER TEST

This test verifies that the RH750 transfers data from the Massbus to memory without errors; it also verifies that the RH750 transfers data across page boundaries correctly. This is accomplished by writing an all 1's pattern onto the Massbus in Maintenance Mode, reading that pattern into the first and last long words of a 1 page buffer and checking for data compare errors.

Map Registers for this test are set to point to the Page Frame of the buffer. The Virtual address is then set to point to the last Long Word boundary in the page. The transfer is executed causing the Virtual Address Register to select the next Map Register and transfer four bytes into the first Long word of the buffer.

TEST 14: BYTE COUNTER TEST

This test verifies the Byte Counter functionality in the

following manner: Checks for SA1 faults by floating a one through each BCR bit position, Verifying that the lower half decrements for each byte clocked into the Silo by issuing W-T-D commands with varying byte counts, Verifying that the upper half decrements for each byte clocked into the Silo from the Massbus by issuing R-F-D commands.

TEST 15: FUNCTIONAL READ/WRITE TESTS

This test verifies that the RH750 can execute pages transfers along with the proper execution of read, read reverse, write-check, and write-check reverse. A one-page buffer with an all ones pattern is set up and written onto the Massbus. Pages transfers are then executed using all Map Registers for each of the following drive commands: read, write-check, read reverse, and write-check reverse.

TEST 16: SILO TEST

This test is comprised of two subtests to check for proper Silo operation for various error conditions.

Subtest one verifies that data in the Silo is not transferred to memory when an error occurs during a transfer. This is accomplished by first performing a W-T-D function with the MB Data Parity Bit Inverted. Next, a R-F-D is attempted referencing the bad data. As a result, no data should be transferred to memory.

Subtest two verifies that data in the Silo is not transferred to memory as a result of an Invalid Map Error. This is accomplished by setting up a transfer to occur across two pages in memory, with the second page being invalid. The Silo data from the first page should be successfully transferred to memory while the Silo data from the second page should not be transferred.

TEST 17: INTERRUPT TEST

This test checks for the correct RH750 Interrupt Vector Address and verifies that Interrupts occur for every status condition capable of generating an Interrupt. Interrupts are checked by setting a flag indicating that Interrupts are to occur, clearing the INTOCCURED flag, setting up a condition which will cause the Interrupt, checking the Interrupt flag and reporting errors if the Interrupt failed to occur or if the status is not as expected.

TEST 18: ATTENTION SUMMARY REGISTER TEST

This test checks that SR<18> NED, does not get set when reading the Attention Summary Register, and then looks for a Non-Existent Drive. If an NED is found the NED bit is cleared in the SR, the Attention Summary Register is read, and the SR is checked to make sure that NED is not set.

TEST 19: IGNORE BYTE COUNTER MODE TEST

This test checks the Ignore Byte Counter Mode which allows for transfers of more than 65 Kilobytes. This mode is controlled by CR<04>, IBC bit. The test is comprised of four subtests.

Subtest one checks that the BCR and VAR continue incrementing after BCR overflow while in the IBC mode.

Subtest two checks that a PGE occurs when DTBUSY is set and an attempt is make to set IBC.

Subtest three checks that the BCR stops at the next BCR overflow when IBC is cleared while a transfer is underway.

Subtest four checks that writing to a Map Register in IBC mode clears the Interrupt condition, and that a PGE is not caused as a result of writing to a Map Register while a transfer is in progress.

TEST 20: RH11-TB TEST

In this test the program will look for a Massbus Exerciser present on the Massbus of the RH750 under test. If present, the program will check to see if the MBE is initialized properly, and then will check the integrity of all Massbus signal lines (to provide cable testing), and finally will perform a series of high speed data transfers.

Subtest 1 asserts Massbus Initialize, and if there is an MBE on the Massbus, it will become Drive 7 on the trailing edge of Mass Init. Drive 7's Drive Type Register is checked to see if it is HEX 20, identifying it as an MBE. At this point, if there is no MBE on the Massbus of the RH750 under test, the program exits to the CEP Supervisor. If the MBE is present, the program proceeds, checking for the presence of Drive Available, Medium-On-Line, Drive Present, and Drive Ready. It further checks that Attention, Composite Error, and Positioning-In-Progress are not active.

Subtest 2 tests the Massbus Drive Select Lines by selecting Drive N (where N = 0-7), and examining the RH750 Status

Register for a Non-Existent Drive. If the Drive does not exist, select the MBE as that Drive, clear NED, write Register 0 of that Drive, and checks the MBE Attention Summary Register to see if the correct value appears in the Drive Select Field.

Subtest 3 tests the Massbus Register Select Lines by selecting all Drive Registers starting with Drive Register 0, and incrementing through to the last. Each time the Register Select Field of the Attention Summary Register will be checked to see is the RS Lines were properly written.

Subtest 4 checks the Massbus Attention Line by issuing a search command to the MBE which will result in the MBE asserting Attention in approximately 100 microseconds. Attention is checked in its asserted state. The negation of Attention has already been checked in the MBE Initialization.

Subtest 5 checks the Massbus Control Lines by writing all ones, all zeroes, floating one, and floating zero patterns on the Massbus Control Lines by use of the MBE Data Buffer Register.

Subtest 6 tests the Massbus Occupied Line by checking for its occurence during any normal data transfer operation. The negation of Occupied will be tested here by disabling Occupied, performing a Data Transfer, and testing for Missed Transfer in the RH750.

Subtest 7 checks Massbus Fail by placing the RH750 in the Maintenance Mode which asserts Massbus Fail, and making sure that there is no change in an MBE Register(R FAIL bit should not be set).

Subtest 8 performs a data transfer of an all one's pattern.

Subtest 9 performs a data transfer of an all zero's pattern.

Subtest 10 performs a data transfer of an F00F pattern.

Subtest 11 performs a data transfer of an A55A pattern.

Subtest 12 performs a data transfer of an AA55 pattern.

Subtest 13 performs a data transfer using a floating one pattern.

Subtest 14 performs a data transfer using a floating zero pattern.

Subtest 15 performs a data transfer using an all one's pattern with Write Check Reverse and an Odd Byte Count.

Subtest 16 performs a data transfer using an all one's pattern with Write Check Forward and an Odd Byte Count.

Subtest 17 checks that the Massbus data is loaded into the correct position in the SCDB for the four possible byte boundaries in the forward direction. This is accomplished by use of the Enable Pattern Shift capabilities of the MBE.

Subtest 18 checks that the Massbus data is loaded into the correct position in the SCDB for the four possible byte boundaries in the reverse direction.

Subtest 19 checks that the Massbus data is loaded into the correct position in the SCDB for the four possible byte boundaries in the forward direction. This is accomplished by use of the Enable Pattern Shift capabilities of the MBE and a WRITE CHECK FOREWARD command.

Subtest 20 checks that the Massbus data is loaded into the correct position in the SCDB for the four possible byte boundaries in the reverse direction. This is accomplished by use of the Enable Pattern Shift capabilities of the MBE and a WRITE CHECK REVERSE command.

5 MAINTENANCE HISTORY

1.0 Initial Release September 1980

1.1 Don Monroe January 1981
Modified initialization code to distinguish between an RH750 and an MBE.

1.2 Dom Andella May 1981
Added component callout messages for DT Busy One-Shot and timing components used in Test 10.

1.3 Dan Milleville June 1981
Changed ASCIC messages for MDP numbers from 0 - 7 to 1 - 8, and changed message printed when MBE selected but not present to a hard error.

1.4 Dan Milleville June 1981
Changed timing in Test 10 Subtest 12 to 1 MS., and Test 10 Subtest 13 to 300 US.

1.5 Dan Milleville Jan 1982
Added Subtest 5 to Test 1 - checks that the Unit Under Test (UUT) responds to only 1 address between 40F28800(X) and 40F2F800(X), in steps of 1000(X)
Added Subtest 1 to Test 6 - checks that there is no "bit cross talk" between the 100(X) Map Registers, i.e. writing to 1 register does not result in any other bits in any other registers becoming set.

1.6 Dan Milleville Nov 1982
Because of an ECO, a TSTL had to be added to T1S5 in the routine that tried to clear all registers.

1.7 Ricardo De Andrade May 1983
Fixed Test 1, used to fail with two MBA's because of a bad call to the P_TABLE when the UUT was RH1 or RH2.

1.8 Susan E. Hutcheson January 1985
Test 12 subtest 10 forces a double bit error but does not clean up error status flags in CRS0. These bits must be cleared manually. Search on 1.8 .

Object Module Synopsis

Module Name	Ident	Bytes	File	Creation Date	Creator
ECCAA_1	1.8	11517	ECCAA1.OBJ;1	20-FEB-1985 07:48	VAX/VMS Macro V04-00
ECCAA_2	1.8	21075	ECCAA2.OBJ;1	20-FEB-1985 07:48	VAX/VMS Macro V04-00
ECCAA_3	1.8	15982	ECCAA3.OBJ;1	20-FEB-1985 07:50	VAX/VMS Macro V04-00

Program Section Synopsis

Psect Name	Module Name	Base	End	Length	Align	Attributes
$HEADER		00000200	0000027A	0000007B (123.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD,NOWRT,NOVEC
	ECCAA_1	00000200	0000027A	0000007B (123.)	PAGE 9	
$TSTCNT		0000027C	0000027F	00000004 (4.)	LONG 2	NOPIC,USR,OVR,REL,LCL,NOSHR,NOEXE, RD,NOWRT,NOVEC
	ECCAA_2	0000027C	0000027F	00000004 (4.)	LONG 2	
	ECCAA_3	0000027C	0000027F	00000004 (4.)	LONG 2	
. BLANK .		00000280	000002B6	00000037 (55.)	BYTE 0	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00000280	000002B6	00000037 (55.)	BYTE 0	
ARGLIST		000002B8	000007EB	00000534 (1332.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	000002B8	000005E7	00000330 (816.)	LONG 2	
	ECCAA_3	000005E8	000007EB	00000204 (516.)	LONG 2	
BUFFERS		00000800	00000DFF	00000600 (1536.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD, WRT,NOVEC
	ECCAA_1	00000800	00000DFF	00000600 (1536.)	PAGE 9	
CLEANUP		00000E00	00000E56	00000057 (87.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00000E00	00000E56	00000057 (87.)	LONG 2	
DISPATCH		00000E58	00001037	000001E0 (480.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD,NOWRT,NOVEC
	ECCAA_2	00000E58	00000FBF	00000168 (360.)	LONG 2	
	ECCAA_3	00000FC0	00001037	00000078 (120.)	LONG 2	
DISPATCH_X		00001038	0000104F	00000018 (24.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD,NOWRT,NOVEC
	ECCAA_1	00001038	0000104F	00000018 (24.)	LONG 2	
DRIVE_PRES		00001050	00001089	0000003A (58.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00001050	00001089	0000003A (58.)	WORD 1	
INITIALIZE		0000108C	00001278	000001ED (493.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000108C	00001278	000001ED (493.)	LONG 2	
ISR0		0000127C	00001298	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000127C	00001298	0000001D (29.)	LONG 2	
ISR1		0000129C	000012B8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000129C	000012B8	0000001D (29.)	LONG 2	
ISR10		000012BC	000012D8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000012BC	000012D8	0000001D (29.)	LONG 2	
ISR11		000012DC	000012F8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000012DC	000012F8	0000001D (29.)	LONG 2	
ISR12		000012FC	00001318	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000012FC	00001318	0000001D (29.)	LONG 2	
ISR13		0000131C	00001338	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000131C	00001338	0000001D (29.)	LONG 2	
ISR14		0000133C	00001358	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000133C	00001358	0000001D (29.)	LONG 2	
ISR15		0000135C	00001378	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000135C	00001378	0000001D (29.)	LONG 2	

Psect Name	Module Name	Base	End	Length	Align	Attributes
ISR2		0000137C	00001398	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000137C	00001398	0000001D (29.)	LONG 2	
ISR3		0000139C	000013B8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000139C	000013B8	0000001D (29.)	LONG 2	
ISR4		000013BC	000013D8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000013BC	000013D8	0000001D (29.)	LONG 2	
ISR5		000013DC	000013F8	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000013DC	000013F8	0000001D (29.)	LONG 2	
ISR6		000013FC	00001418	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000013FC	00001418	0000001D (29.)	LONG 2	
ISR7		0000141C	00001438	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000141C	00001438	0000001D (29.)	LONG 2	
ISR8		0000143C	00001458	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000143C	00001458	0000001D (29.)	LONG 2	
ISR9		0000145C	00001478	0000001D (29.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000145C	00001478	0000001D (29.)	LONG 2	
ISSUE_MMREAD		0000147A	00001499	00000020 (32.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000147A	00001499	00000020 (32.)	WORD 1	
MACH_CHK_SRV		0000149C	000014AC	00000011 (17.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000149C	000014AC	00000011 (17.)	LONG 2	
MM_XFER		000014AE	00001528	0000007B (123.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000014AE	00001528	0000007B (123.)	WORD 1	
OWN		0000152A	000031A7	00001C7E (7294.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000152A	000031A7	00001C7E (7294.)	WORD 1	
PRINT_CMDER		000031A8	000031D6	0000002F (47.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000031A8	000031D6	0000002F (47.)	WORD 1	
PRINT_DTERR		000031D8	000031F7	00000020 (32.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000031D8	000031F7	00000020 (32.)	WORD 1	
PRINT_IBCERR		000031F8	0000331D	00000126 (294.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000031F8	0000331D	00000126 (294.)	WORD 1	
PRINT_INTERR		0000331E	0000335B	0000003E (62.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000331E	0000335B	0000003E (62.)	WORD 1	
PRINT_INTERRX		0000335C	0000335C	00000001 (1.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000335C	0000335C	00000001 (1.)	WORD 1	
PRINT_MAPERR		0000335E	00003382	00000025 (37.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000335E	00003382	00000025 (37.)	WORD 1	
PRINT_NULL		00003384	00003386	00000003 (3.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00003384	00003386	00000003 (3.)	WORD 1	
PRINT_NXADR		00003388	0000339D	00000016 (22.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00003388	0000339D	00000016 (22.)	WORD 1	
PRINT_SBE		0000339E	0000356B	000001CE (462.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000339E	0000356B	000001CE (462.)	WORD 1	
PRINT_SILOER		0000356C	0000359A	0000002F (47.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000356C	0000359A	0000002F (47.)	WORD 1	
PRINT_VECTERR		0000359C	000035CA	0000002F (47.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	0000359C	000035CA	0000002F (47.)	WORD 1	
RH11TB_PRES		000035CC	0000360E	00000043 (67.)	WORD 1	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	000035CC	0000360E	00000043 (67.)	WORD 1	
RH_ISR		00003610	0000365F	00000050 (80.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00003610	0000365F	00000050 (80.)	LONG 2	
SUMMARY		00003660	00003669	0000000A (10.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ECCAA_1	00003660	00003669	0000000A (10.)	LONG 2	

Psect Name	Module Name	Base	End	Length	Align	Attributes
TEST_001		00003800	00003C38	00000439 (1081.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00003800	00003C38	00000439 (1081.)	PAGE 9	
TEST_002		00003E00	00004017	00000218 (536.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00003E00	00004017	00000218 (536.)	PAGE 9	
TEST_003		00004200	000043CF	000001D0 (464.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00004200	000043CF	000001D0 (464.)	PAGE 9	
TEST_004		00004400	00004ABD	000006BE (1726.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00004400	00004ABD	000006BE (1726.)	PAGE 9	
TEST_005		00004C00	0000527D	0000067E (1662.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00004C00	0000527D	0000067E (1662.)	PAGE 9	
TEST_006		00005400	0000566B	0000026C (620.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00005400	0000566B	0000026C (620.)	PAGE 9	
TEST_007		00005800	00005988	00000189 (393.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00005800	00005988	00000189 (393.)	PAGE 9	
TEST_008		00005A00	00005DAD	000003AE (942.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00005A00	00005DAD	000003AE (942.)	PAGE 9	
TEST_009		00005E00	0000699D	00000B9E (2974.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00005E00	0000699D	00000B9E (2974.)	PAGE 9	
TEST_010		00006A00	0000748A	00000A8B (2699.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00006A00	0000748A	00000A8B (2699.)	PAGE 9	
TEST_011		00007600	000077D7	000001D8 (472.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00007600	000077D7	000001D8 (472.)	PAGE 9	
TEST_012		00007800	0000847D	00000C7E (3198.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00007800	0000847D	00000C7E (3198.)	PAGE 9	
TEST_013		00008600	0000888A	0000028B (651.)	PAGE 9	NOPIC,USP,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00008600	0000888A	0000028B (651.)	PAGE 9	
TEST_014		00008A00	00008D54	00000355 (853.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00008A00	00008D54	00000355 (853.)	PAGE 9	
TEST_015		00008E00	00009457	00000658 (1624.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_2	00008E00	00009457	00000658 (1624.)	PAGE 9	
TEST_016		00009600	00009A16	00000417 (1047.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_3	00009600	00009A16	00000417 (1047.)	PAGE 9	
TEST_017		00009C00	0000AE6A	0000126B (4715.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_3	00009C00	0000AE6A	0000126B (4715.)	PAGE 9	
TEST_018		0000B000	0000B136	00000137 (311.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_3	0000B000	0000B136	00000137 (311.)	PAGE 9	
TEST_019		0000B200	0000B695	00000496 (1174.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_3	0000B200	0000B695	00000496 (1174.)	PAGE 9	
TEST_020		0000B800	0000D79E	00001F9F (8095.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	ECCAA_3	0000B800	0000D79E	00001F9F (8095.)	PAGE 9	

Symbols By Name

Symbol	Value	Symbol	Value	Symbol	Value
$ENV	00000001	ARG_LIST	0000158C-R	ARRAY_MSG13	00001953-R
$MO	00000001	ARRAY_MSG1	00001814-R	ARRAY_MSG14	00001980-R
ABORT_IF_FAIL	00001583-R	ARRAY_MSG10	0000190B-R	ARRAY_MSG15	000019AD-R
ALL	00000001	ARRAY_MSG11	00001923-R	ARRAY_MSG16	000019DA-R
ALL_ONES	0000FFFF	ARRAY_MSG12	0000193B-R	ARRAY_MSG17	00001A01-R

Symbol	Value	Symbol	Value	Symbol	Value
ARRAY_MSG18	00001A1E-R	ASCII_MBE	00002EA9-R	DRIVE7	0000156C-R
ARRAY_MSG19	00001A3D-R	ASR_OFFSET	00000410	DRIVES_PRESENT	00001050-R
ARRAY_MSG2	00001848-R	ATA	00008000	DRIVE_ADR_TBL	00001550-R
ARRAY_MSG20	00001A5E-R	ATTENTION	00010000	DRIVE_INUSE	00001570-R
ARRAY_MSG21	00001A7D-R	ATTN	00000018	DRIVE_OFFSET	00000080
ARRAY_MSG22	00001A98-R	BCR	00000010	DRIVE_SEL_MSK	0000E000
ARRAY_MSG23	00001ACB-R	BLANK	00002EB1-R	DRV_ERRMASK	0000E000
ARRAY_MSG24	00001AFE-R	BLKSCSBI	0000001C	DRV_INITMASK	00001180
ARRAY_MSG26	00001B31-R	BLKSND_COMD	10000000	DRY	00000080
ARRAY_MSG27	00001B58-R	BR4	00000014	DS$ABORT	00010020
ARRAY_MSG28	00001B7D-R	BR5	00000015	DS$ASKADR	00010090
ARRAY_MSG29	00001BB0-R	BR6	00000016	DS$ASKDATA	00010080
ARRAY_MSG3	0000186F-R	BR7	00000017	DS$ASKLGCL	00010098
ARRAY_MSG30	00001BE3-R	BYTE0	000000FF	DS$ASKSTR	000100A0
ARRAY_MSG31	00001C08-R	BYTE1	0000FF00	DS$ASKVLD	00010088
ARRAY_MSG32	00001C27-R	BYTE2	00FF0000	DS$ATTACH	000101A8
ARRAY_MSG33	00001C46-R	BYTE3	FF000000	DS$BGNSUB	00010030
ARRAY_MSG34	00001C65-R	BYTE_COUNT_MSK	0000FFFF	DS$BRANCH	000100A8
ARRAY_MSG35	00001C84-R	CAR	0000001C	DS$BREAK	00010058
ARRAY_MSG36	00001CA3-R	CAR_SNAP	00001588-R	DS$CANWAIT	00010070
ARRAY_MSG37	00001CC2-R	CB_HUNG	00800000	DS$CHANNEL	00010180
ARRAY_MSG38	00001CE1-R	COMP_MSG1	0000201F-R	DS$CKLOOP	00010040
ARRAY_MSG39	00001D00-R	COMP_MSG2	0000206C-R	DS$CLRVEC	00010168
ARRAY_MSG4	00001884-R	CONT_PC	0000157F-R	DS$CNTRLC	00010078
ARRAY_MSG40	00001D1F-R	CPE	00000008	DS$CVTREG	000100B0
ARRAY_MSG41	00001D3E-R	CR	00000004	DS$ENDPASS	00010010
ARRAY_MSG42	00001D5D-R	CRTED_READ_DATA	20000000	DS$ENDSUB	00010038
ARRAY_MSG43	00001D7C-R	CSR	00000000	DS$ERRDEV	000100C8
ARRAY_MSG44	00001D9B-R	DATA_IBUFFER	00000800-R	DS$ERRHARD	000100D0
ARRAY_MSG45	00001DBA-R	DATA_OBUFFER	00000A00-R	DS$ERRPREP	00010108
ARRAY_MSG46	00001DD9-R	DATA_XFER_ABRT	00001000	DS$ERRSOFT	000100D8
ARRAY_MSG47	00001DFA-R	DATA_XFER_DONE	00002000	DS$ERRSYS	000100C0
ARRAY_MSG48	00001E1B-R	DATA_XFER_LATE	00000800	DS$ESCAPE	00010050
ARRAY_MSG49	00001E40-R	DCM	04000000	DS$FREEDBGSYM	000101B8
ARRAY_MSG5	00001899-R	DEFAULT	00000000	DS$GETBUF	00010120
ARRAY_MSG50	00001E65-R	DEVICES	00001536-R	DS$GETMEM	00010130
ARRAY_MSG51	00001E8A-R	DEVICE_PTR	0000180C-R	DS$GETTERM	000101C8
ARRAY_MSG52	00001EB5-R	DEV_REG	000017EC-R	DS$GPHARD	00010018
ARRAY_MSG53	00001EE0-R	DIF_MSG	00002F9E-R	DS$HELP	000101B0
ARRAY_MSG54	00001F0B-R	DISABLE_LOG	30000000	DS$INITSCB	00010170
ARRAY_MSG55	00001F2D-R	DMD	00000001	DS$INLOOP	00010048
ARRAY_MSG56	00001F4F-R	DOCC	00000100	DS$K_ERROR	00000002
ARRAY_MSG57	00001F71-R	DPE	00000010	DS$K_NORMAL	00000001
ARRAY_MSG58	00001F8D-R	DPR	00000100	DS$K_SEVERE	00000004
ARRAY_MSG59	00001FA9-R	DR	00000014	DS$K_SUBSYS	00000066
ARRAY_MSG6	000018AE-R	DRIVE0	00001550-R	DS$K_WARNING	00000000
ARRAY_MSG60	00001FC5-R	DRIVE1	00001554-R	DS$LOAD	00010198
ARRAY_MSG61	00001FE1-R	DRIVE2	00001558-R	DS$LOADPCS	000101C0
ARRAY_MSG62	00001FFD-R	DRIVE3	0000155C-R	DS$MAPDBGBLOCK	00010118
ARRAY_MSG7	000018C3-R	DRIVE4	00001560-R	DS$MMOFF	00010158
ARRAY_MSG8	000018DB-R	DRIVE5	00001564-R	DS$MMON	00010150
ARRAY_MSG9	000018F3-R	DRIVE6	00001568-R	DS$MOVPHY	00010148

Symbol	Value	Symbol	Value	Symbol	Value
DS$MOVVRT	00010140	DS$_OVERFLOW	00660008	FMT_IBCER2	00002D0A-R
DS$PARSE	000100B8	DS$_POWER	00660098	FMT_IBCER3	00002D1A-R
DS$PRINTB	000100E0	DS$_PROGERR	00660020	FMT_IBCER4	00002D33-R
DS$PRINTF	000100F0	DS$_SEVERE	00660004	FMT_IBCER5	00002D43-R
DS$PRINTS	000100F8	DS$_TRANSL	006600A0	FMT_IBCER6	00002D74-R
DS$PRINTSIG	00010100	DS$_TRUNCATE	00660028	FMT_IBCER7	00002D93-R
DS$PRINTX	000100E8	DS$_UNEXPINT	006600D8	FMT_IBCER8	00002DED-R
DS$PROBE	000101A0	DS$_VASFULL	00660048	FMT_IBCER9	00002E0C-R
DS$RELBUF	00010128	DS$_WARNING	00660000	FMT_IBUS0	000027A6-R
DS$RELMEM	00010138	DS_MSK	0FFFF8FF	FMT_IBUS1	0000277D-R
DS$SERVICE_DISPATCHER	000101D0	DTE	00001000	FMT_INIT1	00002740-R
DS$SETIPL	00010178	DT_ABORT	00000002	FMT_INTFAILED	000030A9-R
DS$SETMAP	00010188	DT_BUSY	80000000	FMT_LOCATION	00002721-R
DS$SETPRIEXV	00010110	DVA	00000800	FMT_MAPFUN_ERR	00002B01-R
DS$SETVEC	00010160	EBL	0000001A	FMT_MAP_EXERR	00002B18-R
DS$SHOCHAN	00010190	EDM	02000000	FMT_MAP_HEAD	000029D4-R
DS$SUMMARY	00010028	ENABLE_PS	00000800	FMT_MAP_REG	000023A5-R
DS$WAITMS	00010060	ERCE	10000000	FMT_MBEREG_ERR	00002BA8-R
DS$WAITUS	00010068	ERR	00004000	FMT_MBE_ASR	00002661-R
DS$_ARITH	006600D0	ERR_STAT	00000008	FMT_MBE_CR1	0000245A-R
DS$_ASBE	00660118	EXCEPTION	00000015	FMT_MBE_ER	000025B0-R
DS$_BADLINK	006600F0	EXISTING_DRIVE	00001574-R	FMT_MBE_SR	00002502-R
DS$_BADTYPE	006600E8	EXP_INTERRUPT	0000157C-R	FMT_MISSEDXFER	00002A8E-R
DS$_BIIC	00660120	EXP_MSG	00002F92-R	FMT_NOINTEST	000030E3-R
DS$_CHME	006600A8	EXT_REG_MSG	00002EAF-R	FMT_NOTHING	000029FA-R
DS$_CHMK	006600E0	EXT_REG_OFFSET	00000400	FMT_NO_MBE	00002E64-R
DS$_DEVNAME	00660108	FAIL	00000013	FMT_NO_MBE_SEL	00002E86-R
DS$_ERROR	00660002	FERR	00000004	FMT_NO_RH750	0000305B-R
DS$_FHWE	00660068	FMT_ALLDRVPRES	000027FC-R	FMT_NX_ER	00002FE0-R
DS$_FRAGBUF	00660080	FMT_ANY_REG	000022EF-R	FMT_PTR	000017E4-R
DS$_ICBUSY	006600C8	FMT_BADNEXUS	0000282B-R	FMT_RHS	00002709-R
DS$_ICERR	006600C0	FMT_BODY	000029EF-R	FMT_SERR	00003036-R
DS$_IHWE	00660060	FMT_BYTMSK_ER	00002CE2-R	FMT_SILOERROR	00002A21-R
DS$_ILLCHAR	00660018	FMT_CMICMD_ERR	00002CCE-R	FMT_STATUS_REG	0000216C-R
DS$_ILLPAGCNT	00660078	FMT_COMP_ERR	00002A03-R	FMT_UNEXP_INT	0000307A-R
DS$_ILLUNIT	00660100	FMT_CONTRL_REG	000020B9-R	FMT_VAL	00002E39-R
DS$_INSFMEM	00660050	FMT_DCMPERR	00002C33-R	FMT_VARER	000028C3-R
DS$_IPL2HI	006600B8	FMT_DIAG_REG	00002233-R	FMT_VARINCERR	0000290D-R
DS$_IVADDR	00660040	FMT_DRIVENORES	000027E6-R	FMT_VECERR	0000300D-R
DS$_IVVECT	00660038	FMT_DRIVERES	000027CF-R	FMT_XFERERR	00002872-R
DS$_KRNLSTK	00660090	FMT_DTERR	00002BD3-R	IBC_MSG	00001804-R
DS$_LOGIC	00660070	FMT_DUADR_ERR	00002976-R	ILF	00000001
DS$_MCHK	00660088	FMT_DUALRS_ERR	00002951-R	IMAPP	0000001D
DS$_MMOFF	00660058	FMT_DUALRS_MSG	00002F26-R	IMBCP	0000001E
DS$_NEEDUNIT	006600F8	FMT_DUALWT_ERR	00002931-R	IMBDP	0000001F
DS$_NODE	00660128	FMT_DUALWT_MSG	00002ECF-R	INDEX_MSG	00002FD8-R
DS$_NOPCS	00660110	FMT_ERRSUM	00002CAC-R	INH_BYTE_CNT	00000010
DS$_NORMAL	00660001	FMT_EXCP_ERR	00002AD0-R	INTERRUPT_EN	00000004
DS$_NOSUPPORT	006600B1	FMT_EXTREG_ERR	00002B7E-R	INT_OCCURRED	0000157D-R
DS$_NOTDON	00660030	FMT_FATAL_MMM	0000275B-R	INVRT_MAP_PAR	20000000
DS$_NOTIMP	006600B0	FMT_HEADER	000029BE-R	INVRT_MB_CPAR	40000000
DS$_NULLSTR	00660010	FMT_IBCER1	00002CFA-R	INVRT_MB_DPAR	80000000

Symbol	Value	Symbol	Value	Symbol	Value
INVRT_SILO_PAR	00400000	MEM_CSR0	40F20000	READ_REV	0000003F
IPLR	00000012	MEM_CSR1	40F20004	REC_MSG	00002F86-R
ISP	00000016	MIOBUFFER	00000C00-R	REG_NAME	000017D8-R
ISR0	0000127C-R	MISSED_XFER	00000100	REG_NO	000017DC-R
ISR1	0000129C-R	MM_XFER	000014AE-R	REG_SEL_MSK	00001F00
ISR10	000012BC-R	MOL	00001000	REG_STRING	000017E0-R
ISR11	000012DC-R	NIBBLE0	0000000F	REPORT_BUFFER	000015D4-R
ISR12	000012FC-R	NIBBLE1	000000F0	RH11TB_MSK	00001808-R
ISR13	0000131C-R	NIBBLE2	00000F00	RH11TB_PRES	000035CC-R
ISR14	0000133C-R	NIBBLE3	0000F000	RH750_LINK	0000153A-R
ISR15	0000135C-R	NIBBLE4	000F0000	RHBCR_MSG	00002FC0-R
ISR2	0000137C-R	NIBBLE5	00F00000	RHCR_MSG	00002FAA-R
ISR3	0000139C-R	NIBBLE6	0F000000	RHDR_MSG	00002FC8-R
ISR4	000013BC-R	NIBBLE7	F0000000	RHMAPR_MSG	00002FCF-R
ISR5	000013DC-R	NODRIVE	00001578-R	RHSR_CHECK	000F5FFA
ISR6	000013FC-R	NON_XIST_DRIVE	00040000	RHSR_MSG	00002FB1-R
ISR7	0000141C-R	NOOP	00000000	RHVAR_MSG	00002FB8-R
ISR8	0000143C-R	NO_MBE	0000154B-R	RH_CUR_ADR	0000152A-R
ISR9	0000145C-R	NO_MBE_MSG	00002EB3-R	RH_ISR	00003610-R
ISR_TBL	00003168-R	NO_RESPONSE	00000002	RH_VECS	00003124-R
ISSUE_MMREAD	0000147A-R	NO_RH750	0000154A-R	RH_VECTOR	00001542-R
LOWBITS_MSK	0000003F	NUMBER_BUFFER	000017D4-R	RMR	00000004
LUN	00001546-R	NXM_ADR	00001800-R	RRUN	00000002
MACH_CHK_SRV	0000149C-R	NXM_RD_FLAG	0000157E-R	RS_MSK	0FFF07FF
MAINT_MODE	00000008	OCC	00000019	R_FAIL	00000040
MAP_1_PATTRN	80007FFF	OPI	00002000	R_MB_EXC	00000020
MAP_INVALID	00000010	PAGE_BYTE_MSK	000001FF	SA0_MSG	00002F63-R
MAP_OFFSET	00000800	PF_FIELD	00000009	SA1_MSG	00002F6E-R
MAP_PE	00000020	PF_WIDTH	00000015	SCLK	0000001B
MAP_PTR_MSK	0001FE00	PGM_INIT	00000001	SEARCH	00000019
MASS_CNTRL_PE	00020000	PIP	00002000	SEEK	00000002
MASS_CTOD	00010000	PM	08000000	SILO_PE	00004000
MASS_DATA_PE	00000040	PPFN_MASK	00007FFF	SIM_ATTN	01000000
MASS_ECP	00000080	PRINT_CMDER	000031A8-R	SIM_EBL	04000000
MASS_EXCP	00020000	PRINT_DTERR	000031D8-R	SIM_EXCEPT	00200000
MASS_FAIL	00100000	PRINT_IBCERR	000031F8-R	SIM_OCC	02000000
MASS_RUN	00080000	PRINT_INTERR	0000331E-R	SIM_SCLK	08000000
MASS_WCLK	00040000	PRINT_INTERRX	0000335C-R	SR	00000008
MBDIB_SEL	00800000	PRINT_MAPERR	0000335E-R	SUFFIX_PTR	000017E8-R
MBE	00000002	PRINT_NULL	00003384-R	SYS$ALLOC	00010238
MBE_ASR	00000010	PRINT_NULL_X	00003386-R	SYS$ASCTIM	00010248
MBE_CR1	00000000	PRINT_NXADR	00003388-R	SYS$ASSIGN	00010250
MBE_CR2	00000014	PRINT_SBE	0000339E-R	SYS$BINTIM	00010258
MBE_CUR_ADR	0000152E-R	PRINT_SILOER	0000356C-R	SYS$CANCEL	00010260
MBE_DBR	0000001C	PRINT_VECTERR	0000359C-R	SYS$CANTIM	00010268
MBE_DTR	00000018	PROGRAM_ERROR	00080000	SYS$CLOSE	000105B8
MBE_ER	00000008	PTBASE	0000154C-R	SYS$CLREF	00010298
MBE_MR	0000000C	RANDOM	00008259	SYS$CONNECT	000105C0
MBE_PARAM	00000087	RCPA	00000008	SYS$DALLOC	000102D8
MBE_REG_MSG	00002EAD-R	RDPA	00000010	SYS$DASSGN	000102E0
MBE_SR	00000004	READ	00000039	SYS$DISCONNECT	000105D0
MCLK	00000002	READ_HEADER	0000003B	SYS$FAO	00010350

Symbol	Value	Symbol	Value	Symbol	Value
SYS$FAOL	00010358	T12_S9	00007F2B-R	T20_S20	0000D683-R
SYS$GET	00010580	T13_S1	0000861E-R	T20_S3	0000BA98-R
SYS$GETCHN	000104C8	T13_S2	0000872D-R	T20_S4	0000BB38-R
SYS$GETTIM	00010378	T14_S1	00008A1A-R	T20_S5	0000BBDC-R
SYS$LKWSET	000103A0	T14_S2	00008AF9-R	T20_S6	0000BDAC-R
SYS$NUMTIM	000103B8	T14_S3	00008B86-R	T20_S7	0000BEB0-R
SYS$OPEN	00010608	T14_S4	00008C97-R	T20_S8	0000BF79-R
SYS$QIO	000103C8	T15_S1	00008E26-R	T20_S9	0000C1CD-R
SYS$QIOW	00010200	T15_S2	00008F30-R	T3_S1	00004246-R
SYS$READ	00010590	T15_S3	0000908B-R	T3_S2	0000429A-R
SYS$READEF	000103D0	T15_S4	000091F0-R	T4_S1	00004446-R
SYS$SETAST	000103F8	T15_S5	00009344-R	T4_S2	0000449B-R
SYS$SETEF	00010400	T16_S1	00009612-R	T4_S3	0000450F-R
SYS$SETIMR	00010420	T16_S2	0000981A-R	T4_S4	00004579-R
SYS$SETPRI	00010428	T17_S1	00009C5F-R	T5_S1	00004C36-R
SYS$SETPRT	00010430	T17_S10	0000A75A-R	T5_S2	00004CBF-R
SYS$SETRWM	00010438	T17_S11	0000A851-R	T6_S1	00005452-R
SYS$SETSWM	00010448	T17_S12	0000A97B-R	T6_S2	000054DC-R
SYS$ULKPAG	00010460	T17_S13	0000AAAC-R	T6_S3	00005534-R
SYS$ULWSET	00010468	T17_S14	0000ABFC-R	T6_S4	000055E4-R
SYS$UNWIND	00010470	T17_S15	0000AD31-R	T8_S1	00005A1E-R
SYS$WAITFR	00010478	T17_S2	00009E3D-R	T8_S2	00005B00-R
SYS$WFLAND	00010488	T17_S3	00009F3C-R	T8_S3	00005BE1-R
SYS$WFLOR	00010490	T17_S4	00009FE1-R	T8_S4	00005CC2-R
T10_S1	00006A6C-R	T17_S5	0000A0FB-R	T9_S1	00005E2A-R
T10_S10	00007204-R	T17_S6	0000A262-R	T9_S2	00005FFD-R
T10_S11	000072BB-R	T17_S7	0000A395-R	T9_S3	000061B7-R
T10_S12	0000734D-R	T17_S8	0000A4BF-R	T9_S4	0000635B-R
T10_S13	000073F0-R	T17_S9	0000A5FD-R	T9_S5	000064E9-R
T10_S2	00006B7B-R	T18_S1	0000B02A-R	T9_S6	00006677-R
T10_S3	00006C24-R	T18_S2	0000B09D-R	T9_S7	00006805-R
T10_S4	00006D21-R	T19_S1	0000B226-R	TEMP	000017F0-R
T10_S5	00006E01-R	T19_S2	0000B335-R	TEMP1	000017F4-R
T10_S6	00006F11-R	T19_S3	0000B3F9-R	TEMP2	000017F8-R
T10_S7	00006FD0-R	T19_S4	0000B4D4-R	TEMP3	000017FC-R
T10_S8	0000708C-R	T1_S1	00003836-R	TEST_001	00003834-R
T10_S9	00007148-R	T1_S2	000038CC-R	TEST_001_X	00003C31-R
T11_S1	00007631-R	T1_S3	00003965-R	TEST_002	00003E1C-R
T11_S2	000076F9-R	T1_S4	000039FF-R	TEST_002_X	00004010-R
T12_S1	0000782A-R	T1_S5	00003B13-R	TEST_003	00004220-R
T12_S10	00008038-R	T20_S1	0000B816-R	TEST_003_X	000043C8-R
T12_S11	00008156-R	T20_S10	0000C422-R	TEST_004	00004420-R
T12_S12	000081BF-R	T20_S11	0000C69C-R	TEST_004_X	00004AB6-R
T12_S13	0000823C-R	T20_S12	0000C916-R	TEST_005	00004C34-R
T12_S14	0000837A-R	T20_S13	0000CB90-R	TEST_005_X	00005276-R
T12_S2	000078D4-R	T20_S14	0000CDFE-R	TEST_006	0000541C-R
T12_S3	000079E8-R	T20_S15	0000D070-R	TEST_006_X	00005664-R
T12_S4	00007AA1-R	T20_S16	0000D261-R	TEST_007	00005828-R
T12_S5	00007B84-R	T20_S17	0000D337-R	TEST_007_X	00005981-R
T12_S6	00007C6A-R	T20_S18	0000D452-R	TEST_008	00005A1C-R
T12_S7	00007D60-R	T20_S19	0000D57A-R	TEST_008_X	00005DA6-R
T12_S8	00007E4A-R	T20_S2	0000B9BC-R	TEST_009	00005E28-R

Symbol	Value	Symbol	Value	Symbol	Value
TEST_009_X	00006996-R	WAIT_TIME	00001584-R		
TEST_010	00006A28-R	WCEL	00000009		
TEST_010_X	00007483-R	WCEU	0000000A		
TEST_011	0000761C-R	WRITE	00000031		
TEST_011_X	000077D0-R	WRITE_CHECK	00000029		
TEST_012	00007828-R	WRITE_CHECKHDR	0000002B		
TEST_012_X	00008476-R	WRITE_CHECKREV	0000002F		
TEST_013	0000861C-R	WRITE_CHK_HIGH	00000400		
TEST_013_X	00008883-R	WRITE_CHK_LOW	00000200		
TEST_014	00008A18-R	WRITE_HEADER	00000033		
TEST_014_X	00008D4D-R				
TEST_015	00008E24-R				
TEST_015_X	00009450-R				
TEST_016	00009610-R				
TEST_016_X	00009A0F-R				
TEST_017	00009C18-R				
TEST_017_X	0000AE63-R				
TEST_018	0000B028-R				
TEST_018_X	0000B12F-R				
TEST_019	0000B224-R				
TEST_019_X	0000B68E-R				
TEST_020	0000B814-R				
TEST_020_X	0000D797-R				
TEST_VEC	00003164-R				
TYPE_MBE	00000020				
UNEXPECTED	00002F79-R				
VALID_BIT	80000000				
VAR	0000000C				
VAR_PAT1	00000778				
VAR_PAT10	00000800				
VAR_PAT11	00006F80				
VAR_PAT12	00001000				
VAR_PAT13	00005F80				
VAR_PAT14	00002000				
VAR_PAT15	00003F80				
VAR_PAT16	00004000				
VAR_PAT17	00017800				
VAR_PAT18	00008000				
VAR_PAT19	0000F800				
VAR_PAT2	00000080				
VAR_PAT20	00010000				
VAR_PAT21	00017FF8				
VAR_PAT22	00008000				
VAR_PAT23	0000FFF8				
VAR_PAT24	00010000				
VAR_PAT3	000006F8				
VAR_PAT4	00000100				
VAR_PAT5	000005F8				
VAR_PAT6	00000200				
VAR_PAT7	000003F8				
VAR_PAT8	00000400				
VAR_PAT9	00007780				

Symbol	Value	Symbol	Value	Symbol	Value

Key for special characters above:

*	- Undefined
U	- Universal
R	- Relocatable
X	- External

Image Synopsis

```
Virtual memory allocated:                   00000200 0000D7FF 0000D600 (54784. bytes, 107. pages)
Stack size:                                        0. pages
Image binary virtual block limits:                 1.      107. (  107. blocks)
Image name and identification:              ECCAA 1.8
Number of files:                                   3.
Number of modules:                                 3.
Number of program sections:                       67.
Number of global symbols:                        701.
Number of image sections:                          1.
Image type:                                 SYSTEM.
Map format:                                 DEFAULT in file DISK$DIAGPACK02: HUTCHESON.SUPPORT32.ECCAA.NEW ECCAA.MAP;2
Estimated map length:                       117. blocks
```

Link Run Statistics

Performance Indicators	Page Faults	CPU Time	Elapsed Time
Command processing:	54	00:00:00.23	00:00:00.40
Pass 1:	132	00:00:00.98	00:00:02.13
Allocation/Relocation:	15	00:00:00.15	00:00:00.59
Pass 2:	107	00:00:03.65	00:00:05.81
Map data after object module synopsis:	22	00:00:01.33	00:00:01.37
Symbol table output:	0	00:00:00.02	00:00:00.08
Total run values:	330	00:00:06.36	00:00:10.38

```
Using a working set limited to 938 pages and 99 pages of data storage (excluding image)

Total number object records read (both passes):   802
    of which 0 were in libraries and 7 were DEBUG data records containing 1513 bytes

Number of modules extracted explicitly              = 0
    with 0 extracted to resolve undefined symbols

0 library searches were for symbols not in the library searched

A total of 0 global symbol table records was written

LINK/CONTIG/SYST=%X200/EXE=ECCAA/MAP=ECCAA ECCAA1+ECCAA2+ECCAA3
```

Table of contents

(1) 69 DECLARATIONS
(1) 423 OWN STORAGE: PROGRAM TEXT AND FORMAT STATEMENTS
(1) 763 BUFFERS
(2) 926 PRINT_NULL ROUTINE
(2) 969 REPORT AND PRINT MODULE
(2) 1066 PRINT_MAPERR
(2) 1094 PRINT_DTERR
(2) 1116 PRINT_INTERR
(2) 1133 PRINT_INTERRX
(2) 1138 PRINT_NXADR
(2) 1150 PRINT_CMDER
(2) 1168 PRINT_IBCERR
(2) 1228 PRINT_VECTERR
(2) 1246 PRINT_SILOER
(2) 1263 INITIALIZATION
(3) 1370 CODE EXECUTED AT THE END OF EACH TEST SEQUENCE
(4) 1389 MACHINE CHECK SERVICE ROUTINE
(4) 1405 RH750 INTERRUPT SERVICE ROUTINE
(4) 1445 MMREAD SUBROUTINE
(4) 1471 MM_XFER SUBROUTINE
(4) 1524 RH11TB_PRES SUBROUTINE
(4) 1553 DRIVES PRESENT

```
0000     1 .TITLE  ECCAA_1 RH750 REPAIR LEVEL MODULE 1
0000     2 .IDENT  /1.8/
0000     3
0000     4 ;
0000     5 ; COPYRIGHT (C) 1980,1983
0000     6 ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
0000     7 ;
0000     8 ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
0000     9 ; AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE
0000    10 ; AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE.  THIS
0000    11 ; SOFTWARE,  OR  ANY OTHER  COPIES  THEREOF, MAY NOT BE PROVIDED
0000    12 ; OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO
0000    13 ; AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERED.
0000    14 ;
0000    15 ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
0000    16 ; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
0000    17 ; EQUIPMENT CORPORATION.
0000    18 ;
0000    19 ; DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR  RELIABILITY
0000    20 ; OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000    21 ;
0000    22
0000    23 ;++
0000    24 ; FACILITY: VAX DIAGNOSTIC LIBRARY
0000    25 ;
0000    26 ; ABSTRACT:
0000    27 ; THIS PROGRAM VERIFIES THE RH750 MASS BUS ADAPTER FUNCTIONALITY
0000    28 ;
0000    29 ; ENVIRONMENT: DIAGNOSTIC SUPERVISOR
0000    30 ;
0000    31 ; REVISION HISTORY:
0000    32 ;
0000    33 ;       1.0     Initial Release September 1980
0000    34 ;       1.1     Don Monroe      January 1981
0000    35 ;               Modified initialization code to distinguish between an
0000    36 ;               RH750 and an MBE.
0000    37 ;       1.2     Dom Andella     May 1981
0000    38 ;               Added component callout messages for DT Busy One-Shot
0000    39 ;               and timing components used in Test 10.
0000    40 ;       1.3     Dan Milleville  June 1981
0000    41 ;               Changed ASCIC messages for MDP numbers from 0 - 7 to 1 - 8,
0000    42 ;               and changed message printed when MBE selected but not present
0000    43 ;               to a hard error.
0000    44 ;       1.4     Dan Milleville  June 1981
0000    45 ;               Changed timing in Test 10 Subtest 12 to 1 MS., and Test 10
0000    46 ;               Subtest 13 to 300 US.
0000    47 ;       1.5     Dan Milleville  Jan 1982
0000    48 ;               Added Subtest 5 to Test 1 - checks that the Unit Under Test
0000    49 ;               (UUT) responds to only 1 address between 40F28800(X) and
0000    50 ;               40F2F800(X), in steps of 1000(X)
0000    51 ;               Added Subtest 1 to Test 6 - checks that there is no "bit cross-
0000    52 ;               talk" between the 100(X) Map Registers, i.e. writing to 1
0000    53 ;               register does not result in any other bits in any other regis-
0000    54 ;               ters becoming set.
0000    55 ;       1.6     Dan Milleville  Nov 1982
0000    56 ;               Because of an ECO, a TSTL had to be added to T1S5 in the routine
0000    57 ;               that tried to c.ear all registers.
```

```
0000     58 ;        1.7     Ricardo De Andrade May 1983
0000     59 ;                Fixed Test 1, used to fail with two MBA's because of a bad
0000     60 ;                call to the P_TABLE when the UUT was RH1 or RH2.
0000     61 ;
0000     62 ;        1.8     Susan E. Hutcheson                    January 1985
0000     63 ;                Test 12 subtest 10 forces a double bit error but does not clean
0000     64 ;                up error status flags in CRS0. These bits must be cleared man-
0000     65 ;                ually. Search on  1.8 .
0000     66 ;--
```

```
                        0000     67
                        0000     68 ;--
```

```
                                0000     69 .SBTTL  DECLARATIONS
                                0000     70
                                0000     71 ;
                                0000     72 ; LIBRARY FILES:
                                0000     73 ;
                                0000     74 .LIBRARY        /SYS$LIBRARY:DIAG/
                                0000     75
                                0000     76 $DS_BGNMOD      CEP_REPAIR
                                0000        ;;;     Macro-32 Diagnostic macro library Version 7.0 "DIAG.MLB (186)"
                                0000     77 $DS_DISPATCH
                                0000                        .SAVE
                             00000000                       .PSECT  DISPATCH, LONG, NOWRT, NOEXE
                                0000        DISPATCH:
                             00000000                       .PSECT  DISPATCH_X, LONG, NOWRT, NOEXE
00000000'00000000'00000000'00000000'  0000                  .LONG   0 6
                  00000000'00000C0'   0010
                                0000                        .RESTORE
                                0000     78 $DS_HEADER      <ECCAA-REV. 1.8    RH750 DIAGNOSTIC>,1,8,6      ; 1.8
                                0000                        .SAVE
                             00000000                       .PSECT  $HEADER, PAGE, NOEXE, NOWRT
                     00000058'  0000        L$L_HEADLENGTH: .LONG    A_HEADEND-.    ; LENGTH OF THE HEADER DATA BLOCK
                     00000005   0004        L$L_ENVIRON:    .LONG    <ENV$_SUPER@ENV$V_SUPER>+$ENV  ; PROGRAM ENVIRONMENT
                     00000058'  0008        L$A_NAME:       .ADDRESS T_NAME         ; PROGRAM NAME TEXT ADDRESS
                     00000001   000C        L$L_REV:        .LONG    1              ; PROGRAM REVISION LEVEL
                     00000008   0010        L$L_UPDATE:     .LONG    8              ; DIAGNOSTIC ENGINEERING PATCH ORDER
                     00000000'  0014        L$A_LASTAD:     .ADDRESS LASTAD         ; FIRST FREE LOCATION AFTER PROGRAM
                     00000000'  0018        L$A_DTP:        .ADDRESS DISPATCH       ; TEST DISPATCH TABLE POINTER
                     0000002B'  001C        L$A_DEVP:       .ADDRESS AL_DEVTYP      ; DEVICE TYPE LIST POINTER
                     00000006   0020        L$L_UNIT:       .LONG    6              ; NUMBER OF UNITS THAT CAN BE TESTED
                     000002C2'  0024        L$A_DREG:       .ADDRESS DEV_REG        ; DEVICE REGISTER CONTENTS TABLE POINTER
                     0000001C   0028                        .LONG   28              ; Ptable length for 4.05 compatability
00000000'00000000'00000000'00000000'  002C                  .LONG   0 4             ; UNUSED OFFSETS
                     00000000'  003C        L$A_ICP:        .ADDRESS INITIALIZE     ; INITIALIZATION CODE POINTER
                     00000000'  0040        L$A_CCP:        .ADDRESS CLEAN_UP       ; CLEAN-UP CODE POINTER
                     00000000'  0044        L$A_REPP:       .ADDRESS SUMMARY        ; SUMMARY REPORT CODE POINTER
                     00000000'  0048        L$A_STATAB:     .ADDRESS 0              ; STATISTIC TABLE POINTER
                     00000000   004C        L$L_ERRTYP:     .LONG    0              ; # OF TYPES OF $ERRSOFT & $ERRPREP
                     00000010'  0050        L$A_SECNAM:     .ADDRESS SECTION        ; LIST OF SECTION NAME ADDRESSES
                     00000000'  0054        L$A_TSTCNT:     .ADDRESS L_TSTCNT       ; POINTER TO NUMBER OF TESTS
                                0058        A_HEADEND:
20 2E 56 45 52 2D 41 41 43 43 45 00'  0058  T_NAME:         .ASCIC   ECCAA-REV. 1.8    RH750 DIAGNOSTIC
30 35 37 48 52 20 20 20 20 38 2E 31   0064
   43 49 54 53 4F 4E 47 41 49 44 20   0070
                              22   0058
                             00000000                       .PSECT  _LAST, PAGE
                                0000        LASTAD:
                             00000000                       .PSECT  $TSTCNT, NOEXE, NOWRT, OVR, LONG
                                0000        L_TSTCNT:
                                0000                        .RESTORE
                                0000     79 $DS_SECTION     <ALL,MBE>
                                0000        S_DEFAULT:
            54 4C 55 41 46 45 44 00'  0000                  .ASCIC   DEFAULT
                              07   0000
                                0008        S_ALL:
                     4C 4C 41 00'  0008                     .ASCIC   ALL
                              03   0008
                                000C        S_MBE:
```

```
                45 42 4D 00'  000C                      .ASCIC   MBE
                      03      000C
                              0010      SECTION:
                00000003      0010                      .LONG    3
                00000000'     0014                      .LONG    S_DEFAULT
                00000008'     0018                      .LONG    S_ALL
                0000000C'     001C                      .LONG    S_MBE
                              0020    80 $DS_DEVTYP       <RH750,MBE>
          30 35 37 48 52 00'  0020       T_RH750:                .ASCIC   RH750
                      05      0020
                45 42 4D 00'  0026       T_MBE:          .ASCIC   MBE
                      03      0026
                      00      002A                  .BYTE    0
                00000002      002B       AL_DEVTYP:     .LONG    $$N
                00000020'     002F                      .LONG    T_RH750
                00000026'     0033                      .LONG    T_MBE
                              0037    81 ;$DS_QADEF
                              0037    82 $DS_HPODEF
                              0037               .SAVE    LOCAL_BLOCK
                              0037               .PSECT   $ABS$,ABS
                              0000               .IIF     NB,HP$Q_DEVICE, HP$Q_DEVICE:
                00000008      0000               .IIF     NB,.BLKQ,       .BLKQ   1
                              0008               .IIF     NB,HP$W_SIZE,   HP$W_SIZE:
                0000000A      0008               .IIF     NB,.BLKW,       .BLKW   1
                              000A               .IIF     NB,HP$B_FLAGS,  HP$B_FLAGS:
                0000000B      000A               .IIF     NB,.BLKB,       .BLKB   1
                              000B               .IIF     NB,HP$B_DRIVE,  HP$B_DRIVE:
                0000000C      000B               .IIF     NB,.BLKB,       .BLKB   1
                              000C               .IIF     NB,HP$T_DEVICE, HP$T_DEVICE:
                00000018      000C               .IIF     NB,.BLKB,       .BLKB   12
                              0018               .IIF     NB,HP$A_DEVICE, HP$A_DEVICE:
                0000001C      0018               .IIF     NB,.BLKL,       .BLKL   1
                              001C               .IIF     NB,HP$A_DVA,    HP$A_DVA:
                00000020      001C               .IIF     NB,.BLKL,       .BLKL   1
                              0020               .IIF     NB,HP$A_LINK,   HP$A_LINK:
                00000024      0020               .IIF     NB,.BLKL,       .BLKL   1
                              0024               .IIF     NB,HP$W_VECTOR, HP$W_VECTOR:
                00000026      0024               .IIF     NB,.BLKW,       .BLKW   1
                              0026               .IIF     NB,HP$T_TYPE,   HP$T_TYPE:
                00000032      0026               .IIF     NB,.BLKB,       .BLKB   12
                              0032               .IIF     NB,HP$A_DEPENDENT,      HP$A_DEPENDENT:
                     00000037                    .RESTORE
                              0037    83 $DS_RH750_DEF
                              0037               .SAVE    LOCAL_BLOCK
                              0037               .PSECT   $ABS$,ABS
                              0032               .IIF     NB,HP$B_RH750_BR,       HP$B_RH750_BR:
                              0032               .IIF     NB,RH750$B_BR,  RH750$B_BR:
                00000033      0032               .IIF     NB,.BLKB,       .BLKB   1
                              0033               .IIF     NB,HP$K_RH750_LEN,      HP$K_RH750_LEN:
                              0033               .IIF     NB,RH750$K_LEN, RH750$K_LEN:
                     00000037                    .RESTORE
                              0037    84 $DS_DSADEF
                              0037    85 $DS_DSSDEF       GLOBAL
                              0037    86 $DS_DSDEF        GLOBAL
                              0037    87 $DS_ERRDEF
                              0037    88 .MACRO  $DS_BERROR       ADR
                              0037    89         BLBC     R0,ADR
```

```
                0037    90 .ENDM   $DS_BERROR
                0037    91 .MACRO  $DS_BNERROR     ADR
                0037    92         BLBS    R0,ADR
                0037    93 .ENDM   $DS_BNERROR
                0037    94
                0037    95
                0037    96         .MACRO  ERRPREP, MREG_NAME= $$$YN,MREG_NO =-1,-
                0037    97                 MREG_STRING = $$$YN,SUFFIX = BLANK
                0037    98 ;++
                0037    99 ; THIS MACRO IS USED TO SET UP ERROR DISPLAY PARAMETERS
                0037   100 ; FOR THE PRINT ERROR ROUTINE
                0037   101 ;
                0037   102 ;       MREG_NAME IS A POINTER TO THE STRING DESCRIBING THE REGISTER
                0037   103 ;       WHICH PRECEEDS THE ERROR CONDITION
                0037   104 ;
                0037   105 ;       MREG_NO IS THE RH750 REGISTER NUMBER UNDER TEST.
                0037   106 ;
                0037   107 ;       MREG_STRING IS A POINTER TO THE STRING CONTAINING THE BIT
                0037   108 ;       MNEMONICS OF THE REGISTER UNDER TEST.
                0037   109 ;
                0037   110 ;
                0037   111 ;       SUFFIX IS A POINTER TO THE SUFFIX APPENDED TO THE ERROR REPORT.
                0037   112 ;--
                0037   113                 .IF     DF MREG_NAME
                0037   114                    MOVAL       MREG_NAME,REG_NAME
                0037   115                 .ENDC
                0037   116                 .IF     GE MREG_NO
                0037   117                    MOVL        #MREG_NO,REG_NO
                0037   118                 .ENDC
                0037   119                 .IF     DF MREG_STRING
                0037   120                    MOVAL       MREG_STRING,REG_STRING
                0037   121                 .ENDC
                0037   122                 .IF     DF SUFFIX
                0037   123                    MOVAL       SUFFIX,SUFFIX_PTR
                0037   124                 .ENDC
                0037   125                 .ENDM
                0037   126
                0037   127 ;++
                0037   128 ; EQUATED SYMBOLS:
                0037   129 ;--
                0037   130
                0037   131 $DS_BITDEF
                0037   132
                0037   133 ;++
                0037   134 ;BUS REQUEST LEVEL DEFINITIONS (BR4 TO BR7)
                0037   135 ;--
00000014        0037   136 BR4      == X14
00000015        0037   137 BR5      == X15
00000016        0037   138 BR6      == X16
00000017        0037   139 BR7      == X17
                0037   140
                0037   141 ;++
                0037   142 ; DEFINE INTERNAL PROCESSOR REGISTERS
                0037   143 ;--
00000012        0037   144 IPLR             ==      18              ; IPL REGISTER
                0037   145
                0037   146 ;++
```

```
                     0037   147 ; THE FOLLOWING ARE USED AS PC OFFSETS FOR ACCESSING RH750
                     0037   148 ; INTERNAL REGISTERS
                     0037   149 ;--
00000000             0037   150 CSR             ==      0               ; CONFIGURATION/STATUS REGISTER
00000004             0037   151 CR              ==      4               ; CONTROL REGISTER
00000008             0037   152 SR              ==      8               ; STATUS REGISTER
0000000C             0037   153 VAR             ==      12              ; VIRTUAL ADDRESS REGISTER
00000010             0037   154 BCR             ==      16              ; BYTE COUNT REGISTER
00000014             0037   155 DR              ==      20              ; DIAGNOSTIC REGISTER
0000001C             0037   156 CAR             ==      28              ; COMMAND/ADDRESS REGISTER
                     0037   157                                         ; (ALSO A DERIVATIVE OF THE
                     0037   158                                         ; MAPPING FUNCTION)
                     0037   159 ;++
                     0037   160 ; THE FOLLOWING ARE USED AS PC OFFSETS FOR ACCESSING
                     0037   161 ; MBE REGISTERS
                     0037   162 ;--
00000000             0037   163 MBE_CR1          ==      0               ; MBE CONTROL REGISTER 1
00000004             0037   164 MBE_SR           ==      4               ; MBE STATUS REGISTER
00000008             0037   165 MBE_ER           ==      8               ; MBE ERROR REGISTER
0000000C             0037   166 MBE_MR           ==      12              ; MBE MAINTENANCE REGISTER
00000010             0037   167 MBE_ASR          ==      16              ; MBE ATTENTION SUMMARY REGISTER
00000014             0037   168 MBE_CR2          ==      20              ; MBE CONTROL REGISTER 2
00000018             0037   169 MBE_DTR          ==      24              ; MBE DRIVE TYPE REGISTER
0000001C             0037   170 MBE_DBR          ==      28              ; MBE DATA BUFFER REGISTER
                     0037   171
                     0037   172 ;++
                     0037   173 ; THE FOLLOWING EQUATES ARE USED IN BUILDING IO ADDRESSES
                     0037   174 ;--
00000400             0037   175 EXT_REG_OFFSET  ==      X400            ; OFFSET FROM CONFIG REGISTER TO
                     0037   176                                         ; ACCESS DRIVE REGISTERS
00000080             0037   177 DRIVE_OFFSET    ==      X80             ; OFFSET FROM ONE DRIVE TO NEXT
00000410             0037   178 ASR_OFFSET      ==      X410            ; OFFSET TO ATT SUMMARY REG
00000800             0037   179 MAP_OFFSET      ==      X800            ; OFFSET FROM CONFIG REGISTER TO
                     0037   180                                         ; ACCESS MAP REGISTERS
30000000             0037   181 DISABLE_LOG     ==      X30000000       ; CODE TO DISABLE ERROR LOGS
                     0037   182
                     0037   183 ;++
                     0037   184 ; THE FOLLOWING ARE DRIVE COMMANDS RECOGNIZED BY THE RH750
                     0037   185 ;--
00000000             0037   186 NOOP            ==      0               ; DRIVE NOP COMMAND
00000002             0037   187 SEEK            ==      2               ; SEEK COMMAND
00000019             0037   188 SEARCH          ==      X19             ; SEARCH COMMAND
00000029             0037   189 WRITE_CHECK     ==      X29             ; WRITE CHECK (FORWARD) COMMAND
0000002B             0037   190 WRITE_CHECKHDR  ==      X2B             ; WRITE CHECK HEADER
0000002F             0037   191 WRITE_CHECKREV  ==      X2F             ; WRITE CHECK REVERSE
00000031             0037   192 WRITE           ==      X31             ; WRITE (FORWARD)
00000033             0037   193 WRITE_HEADER    ==      X33             ; WRITE HEADER AND DATA
00000039             0037   194 READ            ==      X39             ; READ (FORWARD)
0000003B             0037   195 READ_HEADER     ==      X3B             ; READ HEADER AND DATA
0000003F             0037   196 READ_REV        ==      X3F             ; READ REVERSE
                     0037   197
                     0037   198 ;++
                     0037   199 ; THE FOLLOWING DEFINES BIT ASSIGNMENTS FOR SINGLE BIT
                     0037   200 ; FIELDS IN THE RH750 REGISTERS.
                     0037   201
                     0037   202 ;++
                     0037   203 ; DEFINE THE STATUS REGISTER ASSIGNMENTS
```

```
                    0037   204 ;--
80000000            0037   205 DT_BUSY          ==      BIT31           ; DATA TRANSFER BUSY
20000000            0037   206 CRTED_READ_DATA  ==      BIT29           ; CORRECTED READ DATA
00800000            0037   207 CB_HUNG          ==      BIT23           ; CONTROL BUS HUNG
00080000            0037   208 PROGRAM_ERROR    ==      BIT19           ; PROGRAMMING ERROR
00040000            0037   209 NON_XIST_DRIVE   ==      BIT18           ; NON EXISTENT DRIVE
00020000            0037   210 MASS_CNTRL_PE    ==      BIT17           ; MASSBUS CONTROL PARITY ERROR
00010000            0037   211 ATTENTION        ==      BIT16           ; ATTENION FROM MASS BUS
00004000            0037   212 SILO_PE          ==      BIT14           ; SILO PARITY ERROR
00002000            0037   213 DATA_XFER_DONE   ==      BIT13           ; DATA TRANSFER COMPLETE
00001000            0037   214 DATA_XFER_ABRT   ==      BIT12           ; DATA XFER ABORT
00000800            0037   215 DATA_XFER_LATE   ==      BIT11           ; DATA XFER LATE
00000400            0037   216 WRITE_CHK_HIGH   ==      BIT10           ; WRITE CHECK UPPER ERROR
00000200            0037   217 WRITE_CHK_LOW    ==      BIT9            ; WRITE CHECK LOWER ERROR
00000100            0037   218 MISSED_XFER      ==      BIT8            ; MISSED XFER
00000080            0037   219 MASS_ECP         ==      BIT7            ; MASS BUS EXCEPTION
00000040            0037   220 MASS_DATA_PE     ==      BIT6            ; MASS BUS DATA PARITY ERROR
00000020            0037   221 MAP_PE           ==      BIT5            ; MAP PARITY ERROR
00000010            0037   222 MAP_INVALID      ==      BIT4            ; INVALID MAP
00000008            0037   223 ERR_STAT         ==      BIT3            ; ERROR STATUS
00000002            0037   224 NO_RESPONSE      ==      BIT1            ; NO RESPONSE STATUS
                    0037   225
                    0037   226 ;++
                    0037   227 ; DEFINE THE CONTROL REGISTER BIT ASSIGNMENTS
                    0037   228 ;--
00000010            0037   229 INH_BYTE_CNT     ==      BIT4            ; IGNORE BYTE COUNTER
00000008            0037   230 MAINT_MODE       ==      BIT3            ; MAINTENANCE MODE
00000004            0037   231 INTERRUPT_EN     ==      BIT2            ; INTERRUPT ENABLE
00000002            0037   232 DT_ABORT         ==      BIT1            ; ABORT TRANSFER
00000001            0037   233 PGM_INIT         ==      BIT0            ; INITIALIZE RH750 AND MASS BUS
                    0037   234
                    0037   235 ;++
                    0037   236 ; DEFINE DIAGNOSTIC REGISTER BIT ASSIGNMENTS
                    0037   237 ;--
80000000            0037   238 INVRT_MB_DPAR    ==      BIT31           ; INVERT MASS BUS DATA PARITY
40000000            0037   239 INVRT_MB_CPAR    ==      BIT30           ; INVERT MASS BUS CONTROL PARITY
20000000            0037   240 INVRT_MAP_PAR    ==      BIT29           ; INVERT MAP PARITY
10000000            0037   241 BLKSND_COMD      ==      BIT28           ; BLOCK SENDING COMMANDS TO SBI
08000000            0037   242 SIM_SCLK         ==      BIT27           ; SIMULATE SYNC CLOCK ON MASS BUS
04000000            0037   243 SIM_EBL          ==      BIT26           ; SIMULATE END BLOCK
02000000            0037   244 SIM_OCC          ==      BIT25           ; SIMULATE OCCUPIED
01000000            0037   245 SIM_ATTN         ==      BIT24           ; SIMULATE ATTENTION
00800000            0037   246 MBDIB_SEL        ==      BIT23           ; MASS BUS DATA INPUT BUFFER SELECT
                    0037   247                                            ; SELECTS UPPER OR LOWER BYTE
                    0037   248                                            ; 1 ==> UPPER BYTE
                    0037   249                                            ; 0 ==> LOWER BYTE
00400000            0037   250 INVRT_SILO_PAR   ==      BIT22           ; INVERT SILO PARITY
00200000            0037   251 SIM_EXCEPT       ==      BIT21           ; SIMULATE EXCEPTION
00100000            0037   252 MASS_FAIL        ==      BIT20           ; MASS BUS FAIL SENSE BIT
00080000            0037   253 MASS_RUN         ==      BIT19           ; MASS BUS RUN SENSE BIT
00040000            0037   254 MASS_WCLK        ==      BIT18           ; MASS BUS WRITE CLOCK SENSE BIT
00020000            0037   255 MASS_EXCP        ==      BIT17           ; MASS BUS EXCEPTION SENSE BIT
00010000            0037   256 MASS_CTOD        ==      BIT16           ; MASS BUS CONTROLLER TO DRIVE BIT
                    0037   257
                    0037   258 ;++
                    0037   259 ; DEFINE MEMORY CSR1 BIT ASSIGNMENTS
                    0037   260 ;--
```

```
10000000  0037   261 ERCE            ==      BIT28           ; ENABLE REPORTING CORRECTABLE ERRORS
08000000  0037   262 PM              ==      BIT27           ; PAGE MODE
04000000  0037   263 DCM             ==      BIT26           ; DIAGNOSTIC CHECK MODE
02000000  0037   264 EDM             ==      BIT25           ; ECC DISABLE MODE
          0037   265
          0037   266 ;++
          0037   267 ; THE FOLLOWING EQUATES ARE USED IN BRANCH BIT INSTRUCTIONS REFERENCING
          0037   268 ; THE DIAGNOSTIC REGISTER
          0037   269 ;--
0000001F  0037   270 IMBDP           ==      31              ; INVERT MASS BUS DATA PARITY BIT
0000001E  0037   271 IMBCP           ==      30              ; INVERT MASS BUS CONTROL PARITY
0000001D  0037   272 IMAPP           ==      29              ; INVERT MAP PARITY
0000001C  0037   273 BLKSCSBI        ==      28              ; BLOCK SENDING COMMAND TO SBI
0000001B  0037   274 SCLK            ==      27              ; SIM SCLK
0000001A  0037   275 EBL             ==      26              ; SIMULATE EBL
00000019  0037   276 OCC             ==      25              ; SIMULATE OCCUPIED
00000018  0037   277 ATTN            ==      24              ; SIMULATE ATTENTION
00000016  0037   278 ISP             ==      22              ; INVERT SILO PARITY
00000015  0037   279 EXCEPTION       ==      21              ; SIMULATE EXCEPTION
00000013  0037   280 FAIL            ==      19              ; FAIL SENSE BIT
          0037   281
          0037   282 ;++
          0037   283 ; STATUS BIT POSITONS FOR BBS(C) INSTRUCTIONS
          0037   284 ;--
0000000A  0037   285 WCEU            ==      10              ; WRITE CHECK ERROR UPPER
00000009  0037   286 WCEL            ==      9               ; WRITE CHECK ERROR LOWER
          0037   287
          0037   288 ;++
          0037   289 ; DEFINE MAP REGISTER BIT ASSIGNMENT
          0037   290 ;--
80000000  0037   291 VALID_BIT       ==      BIT31           ; VALID BIT
00007FFF  0037   292 PPFN_MASK       ==       X7FFF          ; PHYSICAL PFN MASK
          0037   293
          0037   294 ;++
          0037   295 ; DEFINE MBE CONTROL REGISTER 1 BIT ASSIGNMENTS
          0037   296 ;--
00000800  0037   297 DVA             ==      BIT11           ; DRIVE AVAILABLE
          0037   298
          0037   299 ;++
          0037   300 ; DEFINE MBE STATUS REGISTER BIT ASSIGNMENTS
          0037   301 ;--
00008000  0037   302 ATA             ==      BIT15           ; ATTENTION ACTIVE
00004000  0037   303 ERR             ==      BIT14           ; COMPOSITE ERROR
00002000  0037   304 PIP             ==      BIT13           ; POSITIONING IN PROGRESS
00001000  0037   305 MOL             ==      BIT12           ; MEDIUM ON LINE
00000100  0037   306 DPR             ==      BIT8            ; DRIVE PRESENT
00000080  0037   307 DRY             ==      BIT7            ; DRIVE READY
00000010  0037   308 RDPA            ==      BIT4            ; DATA PARITY RECIEVED FROM CONTROLLER
00000008  0037   309 RCPA            ==      BIT3            ; CONTROL PARITY FROM CONTROLLER
00000002  0037   310 RRUN            ==      BIT1            ; STATUS OF RUN LINE FROM CONTROLLER
          0037   311
          0037   312 ;++
          0037   313 ; DEFINE MBE ERROR REGISTER BIT ASSIGNMENTS
          0037   314 ;--
00002000  0037   315 OPI             ==      BIT13           ; OPERATION INCOMPLETE
00001000  0037   316 DTE             ==      BIT12           ; DRIVE TIMING ERROR
00000040  0037   317 R_FAIL          ==      BIT6            ; RECEIVED MASSBUS POWER FAIL
```

```
00000020  0037   318 R_MB_EXC          ==      BIT5            ; RECEIVED MASSBUS EXCEPTION
00000010  0037   319 DPE               ==      BIT4            ; DATA PARITY ERROR
00000008  0037   320 CPE               ==      BIT3            ; CONTROL PARITY ERROR
00000004  0037   321 RMR               ==      BIT2            ; REGISTER MODIFY REFUSE
00000C01  0037   322 ILF               ==      BIT0            ; ILLEGAL FUNCTION
          0037   323
          0037   324 ;++
          0037   325 ; DEFINE MBE DIAGNOSTIC REGISTER BIT ASSIGNMENTS
          0037   326 ;--
00000001  0037   327 DMD               ==      BIT0            ; SET DIAGNOSTIC MDOE
00000002  0037   328 MCLK              ==      BIT1            ; MAINTENANCE MODE CLOCK
00000004  0037   329 FERR              ==      BIT2            ; SET FORCE ERROR, USED TO
          0037   330                                            ; CAUSE MASSBUS EXCEPTION
00000100  0037   331 DOCC              ==      BIT8            ; SET DISABLE OCCUPIED
00000800  0037   332 ENABLE_PS         ==      BIT11           ; SET ENABLE PATTERN SHIFT
          0037   333
          0037   334 ;++
          0037   335 ; DEFINE MASK BITS USED TO EXTRACT REGISTER FIELDS
          0037   336 ;--
000000FF  0037   337 BYTE0             ==       X0FF           ; MASK FOR BITS 0 - 7
0000FF00  0037   338 BYTE1             ==       XFF00          ; MASK FOR BITS 8 - 15
00FF0000  0037   339 BYTE2             ==       XFF0000        ; MASK FOR BITS 16 - 23
FF000000  0037   340 BYTE3             ==       XFF000000      ; MASK FOR BITS 24 - 31
0000000F  0037   341 NIBBLE0           ==       XF             ; MASK FOR BITS 0 - 3
000000F0  0037   342 NIBBLE1           ==       XF0            ; MASK FOR BITS 4 - 7
00000F00  0037   343 NIBBLE2           ==       XF00           ; MASK FOR BITS 8 - 11
0000F000  0037   344 NIBBLE3           ==       XF000          ; MASK FOR BITS 12 - 15
000F0000  0037   345 NIBBLE4           ==       XF0000         ; MASK FOR BITS 16 - 19
00F00000  0037   346 NIBBLE5           ==       XF00000        ; MASK FOR BITS 20 - 23
0F000000  0037   347 NIBBLE6           ==       XF000000       ; MASK FOR BITS 24 - 27
F0000000  0037   348 NIBBLE7           ==       XF0000000      ; MASK FOR BITS 28 - 31
0000FFFF  0037   349 BYTE_COUNT_MSK    ==       X0FFFF         ; MASK FOR BYTE COUNT REGISTER
000001FF  0037   350 PAGE_BYTE_MSK     ==       X1FF           ; MASK FOR PAGE BYTE ADDRESS IN
          0037   351                                            ; VIRTUAL ADDRESS REGISTER
0001FE00  0037   352 MAP_PTR_MSK       ==       X1FE00         ; MASK FOR MAP POINTER IN
          0037   353                                            ; VIRTUAL ADDRESS REGISTER
0000E000  0037   354 DRIVE_SEL_MSK     ==       XE000          ; DRIVE SELECT MASK IN
          0037   355                                            ; DIAGNOSTIC REGISTER
00001F00  0037   356 REG_SEL_MSK       ==       X1F00          ; DRIVE REGISTER SELECT MASK
          0037   357                                            ; IN DIAGNOSTIC REGISTER
80007FFF  0037   358 MAP_1_PATTRN      ==       X80007FFF      ; ALL ONES PATTERN FOR WRITING
          0037   359                                            ; MAP REGISTERS
          0037   360 ;++
          0037   361 ; DEFINE PATTERNS FOR USE IN VIRTUAL ADDRESS REGISTER LOADING TEST
          0037   362 ;--
00000778  0037   363 VAR_PAT1          ==       X778           ; PATTERN ONE
00000080  0037   364 VAR_PAT2          ==       X80            ; PATTERN TWO
000006F8  0037   365 VAR_PAT3          ==       X6F8           ; PATTERN THREE
00000100  0037   366 VAR_PAT4          ==       X100           ; PATTERN FOUR
000005F8  0037   367 VAR_PAT5          ==       X5F8           ; PATTERN FIVE
00000200  0037   368 VAR_PAT6          ==       X200           ; PATTERN SIX
000003F8  0037   369 VAR_PAT7          ==       X3F8           ; PATTERN SEVEN
00000400  0037   370 VAR_PAT8          ==       X400           ; PATTERN EIGHT
00007780  0037   371 VAR_PAT9          ==       X7780          ; PATTERN NINE
00000800  0037   372 VAR_PAT10         ==       X800           ; PATTERN TEN
00006F80  0037   373 VAR_PAT11         ==       X6F80          ; PATTERN ELEVEN
00001000  0037   374 VAR_PAT12         ==       X1000          ; PATTERN TWELVE
```

```
00005F80  0037   375 VAR_PAT13        ==        X5F80          ; PATTERN THIRTEEN
00002000  0037   376 VAR_PAT14        ==        X2000          ; PATTERN FOURTEEN
00003F80  0037   377 VAR_PAT15        ==        X3F80          ; PATTERN FIFTEEN
00004000  0037   378 VAR_PAT16        ==        X4000          ; PATTERN SIXTEEN
00017800  0037   379 VAR_PAT17        ==        X17800         ; PATTERN SEVENTEEN
00008000  0037   380 VAR_PAT18        ==        X8000          ; PATTERN EIGHTEEN
0000F800  0037   381 VAR_PAT19        ==        XF800          ; PATTERN NINETEEN
00010000  0037   382 VAR_PAT20        ==        X10000         ; PATTERN TWENTY
00017FF8  0037   383 VAR_PAT21        ==        X17FF8         ; PATTERN TWENTY-ONE
00008000  0037   384 VAR_PAT22        ==        X8000          ; PATTERN TWENTY-TWO
0000FFF8  0037   385 VAR_PAT23        ==        XFFF8          ; PATTERN TWENTY-THREE
00010000  0037   386 VAR_PAT24        ==        X10000         ; PATTERN TWENTY-FOUR
          0037   387
          0037   388 ;++
          0037   389 ; DEFINE MISCELLANEOUS MASKS FOR MBE TESTS
          0037   390 ;--
0000E000  0037   391 DRV_ERRMASK      ==        XE000          ; ATA, ERR, PIP MASK
00001180  0037   392 DRV_INITMASK     ==        X1180          ; MOL, DPR, DRY MASK
00000020  0037   393 TYPE_MBE         ==        X20            ; MBE DRIVE TYPE IDENTIFIER
0000003F  0037   394 LOWBITS_MSK      ==        X3F            ; MASKS LOWER PORTION OF MBE
          0037   395                                           ; STATUS REGISTER
0000FFFF  0037   396 ALL_ONES         ==        XFFFF          ; ALL ONES DATA PATTERN
00000087  0037   397 MBE_PARAM        ==        X87            ; MBE DATA TRANSFER PARAMETERS
000F5FFA  0037   398 RHSR_CHECK       ==        XF5FFA         ; MASK OF RH750 STATUS REGISTER
          0037   399                                           ; ERROR BITS
00008259  0037   400 RANDOM           ==        X8259          ; PATTERN FOR DATA TRANSFERS
0FFFF8FF  0037   401 DS_MSK           ==        XFFFF8FF       ; MASK TO SELECT ONLY DRIVE
          0037   402                                           ; SELECT FIELD OF ATTENTION
          0037   403                                           ; SUMMARY REGISTER
0FFF07FF  0037   404 RS_MSK           ==        XFFF07FF       ; MASK TO SELECT ONLY REGISTER
          0037   405                                           ; SELECT FIELD OF ATTENTION
          0037   406                                           ; SUMMARY REGISTER OF MBE ASR
          0037   407
00000009  0037   408 PF_FIELD         ==        9              ; LOWER BIT OF PAGE FRAME FIELD
          0037   409                                           ; IN VIRTUAL ADDRESSES
00000015  0037   410 PF_WIDTH         ==        21             ; WIDTH OF PAGE FRAME FIELD
          0037   411
          0037   412 ;++
          0037   413 ; MEMORY CONTROLLER ADDRESS FOR FORCING ERRORS
          0037   414 ;--
40F20004  0037   415 MEM_CSR1         ==        X40F20004      ; ADDRESS OF MEMORY CONTROLLER 1
```

```
                    0037   416 ;
                    0037   417 ;++
                    0037   418 ; MEMORY CONTROLLER ADDRESS FOR CLEARING ERROR STATUS BITS
                    0037   419 ;--
40F20000            0037   420 MEM_CSR0        ==        X40F20000       ; 1.8 ADDRESS OF MEMORY CONTROLLER
                    0037   421                                             ; STAUS REGISTER 0
                    0037   422 ;
                    0037   423         .SBTTL  OWN STORAGE: PROGRAM TEXT AND FORMAT STATEMENTS
                00000000   424         .PSECT  OWN,WORD
                    0000   425 ;++
                    0000   426 ; TABLE OF RH750 ADDRESSES
                    0000   427 ;--
00000000            0000   428 RH_CUR_ADR::    .LONG                   ; CONTAINS ADDRESS OF RH750 UNDER TEST
00000000            0004   429 MBE_CUR_ADR::   .LONG                   ; CONTAINS ADDRESS OF MBE ATTACHED TO
                    0008   430                                         ; CURRENT RH750
00000000            0008   431 LN:             .LONG   0               ; USED IN INIT CODE
00000000            000C   432 DEVICES::       .LONG                   ; TOTAL NUMBER OF SELECTED DEVICES
00000000            0010   433 RH750_LINK::    .LONG                   ; CONTAINS ADDRESS OF RH750 PTABLE UNDER TEST
00000018            0014   434 MBA_LN:         .BLKB   4               ; CONTAINS LUN'S OF ALL PRESENT MBA'S
00000000            0018   435 RH_VECTOR::     .LONG   0               ; CONTAINS INTERRUPT VECTOR OF RH UNDER TEST
00000000            001C   436 LUN::           .LONG   0               ; CHANNEL NUMBER
          00        0020   437 NO_RH750::      .BYTE   0               ; NUMBER OF UNITS SELECTED
          00        0021   438 NO_MBE::        .BYTE   0               ; NUMBER OF MBE's SELECTED
00000000            0022   439 PTBASE::        .LONG   0               ; POINTER TO HARDWARE PARAMETERS
                    0026   440
                    0026   441 ;++
                    0026   442 ; DEFINE A LIST OF DRIVE ADDRESSES FOR THE RH750 UNDER TEST
                    0026   443 ;--
                    0026   444 DRIVE_ADR_TBL::                          ; THIS IS THE TOP OF THE LIST
00000000'           0026   445 DRIVE0::         .ADDRESS        0       ; ADDRESS OF DRIVE 0
00000000'           002A   446 DRIVE1::         .ADDRESS        0       ; ADDRESS OF DRIVE 1
00000000'           002E   447 DRIVE2::         .ADDRESS        0       ; ADDRESS OF DRIVE 2
00000000'           0032   448 DRIVE3::         .ADDRESS        0       ; ADDRESS OF DRIVE 3
00000000'           0036   449 DRIVE4::         .ADDRESS        0       ; ADDRESS OF DRIVE 4
00000000'           003A   450 DRIVE5::         .ADDRESS        0       ; ADDRESS OF DRIVE 5
00000000'           003E   451 DRIVE6::         .ADDRESS        0       ; ADDRESS OF DRIVE6
00000000'           0042   452 DRIVE7::         .ADDRESS        0       ; ADDRESS OF DRIVE 7
                    0046   453
00000000            0046   454 DRIVE_INUSE::    .LONG           0       ; DRIVE ADDRESS TO USE
                    004A   455 EXISTING_DRIVE::
FFFFFFFF            004A   456                 .LONG           -1      ; FLAG AND/OR EXISTING DRIVE TO USE
                    004E   457                                         ; FOR MASSBUS CONTROL PATH TEST
00000000            004E   458 NODRIVE::        .LONG           0       ; 0 INDICATES NO DRIVE FOUND
                    0052   459
                    0052   460 ;++
                    0052   461 ; DEFINE PROGRAM RUN TIME DATA LOCATIONS
                    0052   462 ;--
          00        0052   463 EXP_INTERRUPT:: .BYTE   0               ; INTERRUPT EXPECTED FLAG FOR RH_ISR
          00        0053   464 INT_OCCURRED::  .BYTE   0               ; INTERRUPT EVENT FLAG
          00        0054   465 NXM_RD_FLAG::   .BYTE   0               ; NON-EXISTENT MEMORY READ EXPECTED FLAG
                    0055   466
00000000            0055   467 CONT_PC::       .LONG   0               ; ADDRESS TO RETURN TO AFTER
                    0059   468                                         ; SERVICING RD NXM EXCEPTION
          00        0059   469 ABORT_IF_FAIL:: .BYTE   0               ; ABORT IF FAIL FLAG
00000000            005A   470 WAIT_TIME::     .LONG   0               ; TIME TO WAIT FOR OPERATOR TO ABORT
00000000            005E   471 CAR_SNAP::      .LONG   0               ; SNAP SHOT OF THE CAR BEFORE EBL ISSUED
                    0062   472
```

```
00000000'00000000'00000000'00000000'  0062   473 ARG_LIST::      .LONG   0 18    ; PRINT MODULE ARGUMENT LIST
00000000'00000000'00000000'00000000'  0072
00000000'00000000'00000000'00000000'  0082
00000000'00000000'00000000'00000000'  0092
                  00000000'00000000'  00A2
                           000002AA   00AA   474 REPORT_BUFFER:: .BLKB   512     ; OUTPUT BUFFER FOR PRINTING REGISTER ERRORS
                           00000000   02AA   475 NUMBER_BUFFER:: .LONG   0       ; SMALL BUFFER FOR READING NUMBERS
                                      02AE   476
                                      02AE   477 ;**
                                      02AE   478 ; THE FOLLOWING LOCATIONS ARE SET UP AT THE BEGINNING OF
                                      02AE   479 ; EACH TEST. THESE LOCATIONS ARE USED BY THE PRINT SUBROUTINE
                                      02AE   480 ; FOR PRINTING ERRORS.
                                      02AE   481 ;--
                           00000000   02AE   482 REG_NAME::      .LONG   0       ; ADDRESS OF ERROR PREFIX STRING
                           00000000   02B2   483 REG_NO::        .LONG   0       ; NUMBER OF REGISTER UNDER TEST
                           00000000   02B6   484 REG_STRING::    .LONG   0       ; ADDRESS OF REGISTER MNEMONIC STRING
                           00000000   02BA   485 FMT_PTR::       .LONG   0       ; POINTER TO FORMAT STATEMENT
                           00000000   02BE   486 SUFFIX_PTR::    .LONG   0       ; POINTER TO SUFFIX STRING TYPED AFTER ERROR
                                      02C2   487
                           00000000   02C2   488 DEV_REG::       .LONG   0
                                      02C6   489
                           00000000   02C6   490 TEMP::          .LONG   0       ; EXTRA REGISTERS
                           00000000   02CA   491 TEMP1::         .LONG   0       ;
                           00000000   02CE   492 TEMP2::         .LONG   0       ;
                           00000000   02D2   493 TEMP3::         .LONG   0       ;
                                      02D6   494
                           00000000   02D6   495 NXM_ADR::       .LONG   0       ; STORAGE FACILITY FOR NON-EXISTENT ADDRESSES
                                      02DA   496
                           00000000   02DA   497 IBC_MSG::       .LONG   0       ; STORAGE FOR TEST 19 IBC ERROR MSG POINTER
                                      02DE   498
                           00000000   02DE   499 RH11TB_MSK::    .LONG   0       ; STORAGE FACILITY FOR EXPECTED ATTN BIT IF
                                      02E2   500                                 ; IF RH11-TB PRESENT.
                                      02E2   501
00000000 00000000                     02E2   502 DEVICE_PTR::    .QUAD   0       ; GETS QUAD WORD DEVICE DESCRIPTOR FROM
                                      02EA   503                                 ; THE PTABLE
```

```
                     02EA    504 ;++
                     02EA    505 ; DEFINE FAILING GATE ARRAY MESSAGE BLOCK
                     02EA    506 ;--
                     02EA    507 ;       ARRAY CONVERSION CHART
                     02EA    508 ;
                     02EA    509 ; DC645(0)      =       MDP Bit Slice 0,1,16,17
                     02EA    510 ; DC645(1)      =       MDP Bit Slice 2,3,18,19
                     02EA    511 ; DC645(2)      =       MDP Bit Slice 4,5,20,21
                     02EA    512 ; DC645(3)      =       MDP Bit Slice 6,7,22,23
                     02EA    513 ; DC645(4)      =       MDP Bit Slice 8,9,24,25
                     02EA    514 ; DC645(5)      =       MDP Bit Slice 10,11,26,27
                     02EA    515 ; DC645(6)      =       MDP Bit Slice 12,13,28,29
                     02EA    516 ; DC645(7)      =       MDP Bit Slice 14,15,30,31
                     02EA    517 ;
                     02EA    518 ; DC646         =       MSC
                     02EA    519 ; DC647         =       MRC
                     02EA    520 ; DC648         =       MDC
                     02EA    521 ; DC649         =       MCI
                     02EA    522 ;
                     02EA    523 ;--
                     02EA    524
                     02EA    525
52 52 41 2D 47 4E 49 4C 49 41 46 00'  02EA    526 ARRAY_MSG1::    .ASCIC   FAILING-ARRAYS: ALL DC645'S,DC646,DC647,DC648,DC649
36 43 44 20 4C 4C 41 20 3A 53 59 41   02F6
44 2C 36 34 36 43 44 2C 53 27 35 34   0302
44 2C 38 34 36 43 44 2C 37 34 36 43   030E
                  39 34 36 43         031A
                              33      02EA
52 52 41 2D 47 4E 49 4C 49 41 46 00'  031E    527 ARRAY_MSG2::    .ASCIC   FAILING-ARRAY: DC646,DC647,DC648,DC649
43 44 2C 36 34 36 43 44 20 3A 59 41   032A
43 44 2C 38 34 36 43 44 2C 37 34 36   0336
                        39 34 36      0342
                              26      031E
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0345    528 ARRAY_MSG3::    .ASCIC   FAILING-ARRAY: DC649
         39 34 36 43 44 20 3A 59 41   0351
                              14      0345
52 52 41 2D 47 4E 49 4C 49 41 46 00'  035A    529 ARRAY_MSG4::    .ASCIC   FAILING-ARRAY: DC648
         38 34 36 43 44 20 3A 59 41   0366
                              14      035A
52 52 41 2D 47 4E 49 4C 49 41 46 00'  036F    530 ARRAY_MSG5::    .ASCIC   FAILING-ARRAY: DC647
         37 34 36 43 44 20 3A 59 41   037B
                              14      036F
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0384    531 ARRAY_MSG6::    .ASCIC   FAILING-ARRAY: DC646
         36 34 36 43 44 20 3A 59 41   0390
                              14      0384
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0399    532 ARRAY_MSG7::    .ASCIC   FAILING-ARRAY: DC645(1)
29 31 28 35 34 36 43 44 20 3A 59 41   03A5
                              17      0399
52 52 41 2D 47 4E 49 4C 49 41 46 00'  03B1    533 ARRAY_MSG8::    .ASCIC   FAILING-ARRAY: DC645(3)
29 33 28 35 34 36 43 44 20 3A 59 41   03BD
                              17      03B1
52 52 41 2D 47 4E 49 4C 49 41 46 00'  03C9    534 ARRAY_MSG9::    .ASCIC   FAILING-ARRAY: DC645(5)
29 35 28 35 34 36 43 44 20 3A 59 41   03D5
                              17      03C9
52 52 41 2D 47 4E 49 4C 49 41 46 00'  03E1    535 ARRAY_MSG10::   .ASCIC   FAILING-ARRAY: DC645(6)
29 36 28 35 34 36 43 44 20 3A 59 41   03ED
                              17      03E1
```

```
52 52 41 2D 47 4E 49 4C 49 41 46 00'  03F9   536 ARRAY_MSG11::   .ASCIC   FAILING-ARRAY: DC645(7)
29 37 28 35 34 36 43 44 20 3A 59 41   0405
                              17      03F9
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0411   537 ARRAY_MSG12::   .ASCIC   FAILING-ARRAY: DC645(8)
29 38 28 35 34 36 43 44 20 3A 59 41   041D
                              17      0411
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0429   538 ARRAY_MSG13::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC648
31 28 35 34 36 43 44 20 3A 53 59 41   0435
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   0441
         38 34 36 43 44 2C 29 38 2C   044D
                              2C      0429
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0456   539 ARRAY_MSG14::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC647
31 28 35 34 36 43 44 20 3A 53 59 41   0462
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   046E
         37 34 36 43 44 2C 29 38 2C   047A
                              2C      0456
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0483   540 ARRAY_MSG15::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC646
31 28 35 34 36 43 44 20 3A 53 59 41   048F
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   049B
         36 34 36 43 44 2C 29 38 2C   04A7
                              2C      0483
52 52 41 2D 47 4E 49 4C 49 41 46 00'  04B0   541 ARRAY_MSG16::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8)
31 28 35 34 36 43 44 20 3A 53 59 41   04BC
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   04C8
                        29 38 2C      04D4
                              26      04B0
52 52 41 2D 47 4E 49 4C 49 41 46 00'  04D7   542 ARRAY_MSG17::   .ASCIC   FAILING-ARRAYS: DC645(4,5,6)
34 28 35 34 36 43 44 20 3A 53 59 41   04E3
                  29 36 2C 35 2C      04EF
                              1C      04D7
52 52 41 2D 47 4E 49 4C 49 41 46 00'  04F4   543 ARRAY_MSG18::   .ASCIC   FAILING-ARRAYS: DC645(4,5,6,7)
34 28 35 34 36 43 44 20 3A 53 59 41   0500
               29 37 2C 36 2C 35 2C   050C
                              1E      04F4
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0513   544 ARRAY_MSG19::   .ASCIC   FAILING-ARRAYS: DC645(4,5,6,7,8)
34 28 35 34 36 43 44 20 3A 53 59 41   051F
         29 38 2C 37 2C 36 2C 35 2C   052B
                              20      0513
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0534   545 ARRAY_MSG20::   .ASCIC   FAILING-ARRAYS: DC645(5,6,7,8)
35 28 35 34 36 43 44 20 3A 53 59 41   0540
               29 38 2C 37 2C 36 2C   054C
                              1E      0534
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0553   546 ARRAY_MSG21::   .ASCIC   FAILING-ARRAYS: DC645(2,6)
32 28 35 34 36 43 44 20 3A 53 59 41   055F
                        29 36 2C      056B
                              1A      0553
52 52 41 2D 47 4E 49 4C 49 41 46 00'  056E   547 ARRAY_MSG22::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC647,DC649
31 28 35 34 36 43 44 20 3A 53 59 41   057A
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   0586
43 44 2C 37 34 36 43 44 2C 29 38 2C   0592
                        39 34 36      059E
                              32      056E
52 52 41 2D 47 4E 49 4C 49 41 46 00'  05A1   548 ARRAY_MSG23::   .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC647,DC648
31 28 35 34 36 43 44 20 3A 53 59 41   05AD
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   05B9
43 44 2C 37 34 36 43 44 2C 29 38 2C   05C5
                        38 34 36      05D1
```

```
                                                  32  05A1
52 52 41 2D 47 4E 49 4C 49 41 46 00'  05D4   549 ARRAY_MSG24::    .ASCIC   FAILING-ARRAYS: DC645(1,2,3,4,5,6,7,8),DC646,DC648
31 28 35 34 36 43 44 20 3A 53 59 41   05E0
37 2C 36 2C 35 2C 34 2C 33 2C 32 2C   05EC
43 44 2C 36 34 36 43 44 2C 29 38 2C   05F8
                  38 34 36 43 44 2C   0604
                                  32  05D4
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0607   550 ARRAY_MSG26::    .ASCIC   FAILING-ARRAYS: DC645(2,3,4,5,6),DC648
32 28 35 34 36 43 44 20 3A 53 59 41   0613
43 44 2C 29 36 2C 35 2C 34 2C 33 2C   061F
                  38 34 36 43 44 2C   062B
                                  26  0607
52 52 41 2D 47 4E 49 4C 49 41 46 00'  062E   551 ARRAY_MSG27::    .ASCIC   FAILING-ARRAYS: DC645(2,3,4,5),DC648
32 28 35 34 36 43 44 20 3A 53 59 41   063A
34 36 43 44 2C 29 35 2C 34 2C 33 2C   0646
                                  38  0652
                                  24  062E
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0653   552 ARRAY_MSG28::    .ASCIC   FAILING-ARRAYS: DC645(7,8),DC646,DC647,DC648,DC649
37 28 35 34 36 43 44 20 3A 53 59 41   065F
43 44 2C 36 34 36 43 44 2C 29 38 2C   066B
43 44 2C 38 34 36 43 44 2C 37 34 36   0677
                  39 34 36 43 44 2C   0683
                                  32  0653
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0686   553 ARRAY_MSG29::    .ASCIC   FAILING-ARRAYS: DC645(5,6),DC646,DC647,DC648,DC649
35 28 35 34 36 43 44 20 3A 53 59 41   0692
43 44 2C 36 34 36 43 44 2C 29 36 2C   069E
43 44 2C 38 34 36 43 44 2C 37 34 36   06AA
                  39 34 36 43 44 2C   06B6
                                  32  0686
52 52 41 2D 47 4E 49 4C 49 41 46 00'  06B9   554 ARRAY_MSG30::    .ASCIC   FAILING-ARRAYS: DC645(2,3,4,5),DC649
32 28 35 34 36 43 44 20 3A 53 59 41   06C5
34 36 43 44 2C 29 35 2C 34 2C 33 2C   06D1
                                  39  06DD
                                  24  06B9
52 52 41 2D 47 4E 49 4C 49 41 46 00'  06DE   555 ARRAY_MSG31::    .ASCIC   FAILING-ARRAYS: DC645(1),DC649
31 28 35 34 36 43 44 20 3A 53 59 41   06EA
                  39 34 36 43 44 2C   06F6
                                  1E  06DE
52 52 41 2D 47 4E 49 4C 49 41 46 00'  06FD   556 ARRAY_MSG32::    .ASCIC   FAILING-ARRAYS: DC645(1),DC648
31 28 35 34 36 43 44 20 3A 53 59 41   0709
                  38 34 36 43 44 2C   0715
                                  1E  06FD
52 52 41 2D 47 4E 49 4C 49 41 46 00'  071C   557 ARRAY_MSG33::    .ASCIC   FAILING-ARRAYS: DC645(1),DC647
31 28 35 34 36 43 44 20 3A 53 59 41   0728
                  37 34 36 43 44 2C   0734
                                  1E  071C
52 52 41 2D 47 4E 49 4C 49 41 46 00'  073B   558 ARRAY_MSG34::    .ASCIC   FAILING-ARRAYS: DC645(1),DC646
31 28 35 34 36 43 44 20 3A 53 59 41   0747
                  36 34 36 43 44 2C   0753
                                  1E  073B
52 52 41 2D 47 4E 49 4C 49 41 46 00'  075A   559 ARRAY_MSG35::    .ASCIC   FAILING-ARRAYS: DC645(2),DC649
32 28 35 34 36 43 44 20 3A 53 59 41   0766
                  39 34 36 43 44 2C   0772
                                  1E  075A
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0779   560 ARRAY_MSG36::    .ASCIC   FAILING-ARRAYS: DC645(2),DC647
32 28 35 34 36 43 44 20 3A 53 59 41   0785
                  37 34 36 43 44 2C   0791
```

```
                                                  1E  0779
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0798   561 ARRAY_MSG37::    .ASCIC   FAILING-ARRAYS: DC645(3),DC648
33 28 35 34 36 43 44 20 3A 53 59 41   07A4
               38 34 36 43 44 2C 29   07B0
                                      1E  0798
52 52 41 2D 47 4E 49 4C 49 41 46 00'  07B7   562 ARRAY_MSG38::    .ASCIC   FAILING-ARRAYS: DC645(4),DC648
34 28 35 34 36 43 44 20 3A 53 59 41   07C3
               38 34 36 43 44 2C 29   07CF
                                      1E  07B7
52 52 41 2D 47 4E 49 4C 49 41 46 00'  07D6   563 ARRAY_MSG39::    .ASCIC   FAILING-ARRAYS: DC645(4),DC647
34 28 35 34 36 43 44 20 3A 53 59 41   07E2
               37 34 36 43 44 2C 29   07EE
                                      1E  07D6
52 52 41 2D 47 4E 49 4C 49 41 46 00'  07F5   564 ARRAY_MSG40::    .ASCIC   FAILING-ARRAYS: DC645(4),DC646
34 28 35 34 36 43 44 20 3A 53 59 41   0801
               36 34 36 43 44 2C 29   080D
                                      1E  07F5
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0814   565 ARRAY_MSG41::    .ASCIC   FAILING-ARRAYS: DC645(5),DC648
35 28 35 34 36 43 44 20 3A 53 59 41   0820
               38 34 36 43 44 2C 29   082C
                                      1E  0814
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0833   566 ARRAY_MSG42::    .ASCIC   FAILING-ARRAYS: DC645(7),DC649
37 28 35 34 36 43 44 20 3A 53 59 41   083F
               39 34 36 43 44 2C 29   084B
                                      1E  0833
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0852   567 ARRAY_MSG43::    .ASCIC   FAILING-ARRAYS: DC645(7),DC648
37 28 35 34 36 43 44 20 3A 53 59 41   085E
               38 34 36 43 44 2C 29   086A
                                      1E  0852
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0871   568 ARRAY_MSG44::    .ASCIC   FAILING-ARRAYS: DC645(8),DC648
38 28 35 34 36 43 44 20 3A 53 59 41   087D
               38 34 36 43 44 2C 29   0889
                                      1E  0871
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0890   569 ARRAY_MSG45::    .ASCIC   FAILING-ARRAYS: DC645(8),DC646
38 28 35 34 36 43 44 20 3A 53 59 41   089C
               36 34 36 43 44 2C 29   08A8
                                      1E  0890
52 52 41 2D 47 4E 49 4C 49 41 46 00'  08AF   570 ARRAY_MSG46::    .ASCIC   FAILING-ARRAYS: DC645(1,5),DC649
31 28 35 34 36 43 44 20 3A 53 59 41   08BB
         39 34 36 43 44 2C 29 35 2C   08C7
                                      20  08AF
52 52 41 2D 47 4E 49 4C 49 41 46 00'  08D0   571 ARRAY_MSG47::    .ASCIC   FAILING-ARRAYS: DC645(1,7),DC649
31 28 35 34 36 43 44 20 3A 53 59 41   08DC
         39 34 36 43 44 2C 29 37 2C   08E8
                                      20  08D0
52 52 41 2D 47 4E 49 4C 49 41 46 00'  08F1   572 ARRAY_MSG48::    .ASCIC   FAILING-ARRAYS: DC645(1),DC647,DC649
31 28 35 34 36 43 44 20 3A 53 59 41   08FD
34 36 43 44 2C 37 34 36 43 44 2C 29   0909
                                      39  0915
                                      24  08F1
52 52 41 2D 47 4E 49 4C 49 41 46 00'  0916   573 ARRAY_MSG49::    .ASCIC   FAILING-ARRAYS: DC645(2),DC647,DC649
32 28 35 34 36 43 44 20 3A 53 59 41   0922
34 36 43 44 2C 37 34 36 43 44 2C 29   092E
                                      39  093A
                                      24  0916
52 52 41 2D 47 4E 49 4C 49 41 46 00'  093B   574 ARRAY_MSG50::    .ASCIC   FAILING-ARRAYS: DC645(6),DC646,DC649
36 28 35 34 36 43 44 20 3A 53 59 41   0947
```

```
34 36 43 44 2C 36 34 36 43 44 2C 29  0953
                                 39  095F
                                 24  093B
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0960    575 ARRAY_MSG51::    .ASCIC   FAILING-ARRAYS: DC645(7),DC646,DC647,DC648
37 28 35 34 36 43 44 20 3A 53 59 41  096C
34 36 43 44 2C 36 34 36 43 44 2C 29  0978
               38 34 36 43 44 2C 37  0984
                                 2A  0960
52 52 41 2D 47 4E 49 4C 49 41 46 00' 098B    576 ARRAY_MSG52::    .ASCIC   FAILING-ARRAYS: DC645(8),DC646,DC647,DC648
38 28 35 34 36 43 44 20 3A 53 59 41  0997
34 36 43 44 2C 36 34 36 43 44 2C 29  09A3
               38 34 36 43 44 2C 37  09AF
                                 2A  098B
52 52 41 2D 47 4E 49 4C 49 41 46 00' 09B6    577 ARRAY_MSG53::    .ASCIC   FAILING-ARRAYS: DC645(1),DC646,DC647,DC648
31 28 35 34 36 43 44 20 3A 53 59 41  09C2
34 36 43 44 2C 36 34 36 43 44 2C 29  09CE
               38 34 36 43 44 2C 37  09DA
                                 2A  09B6
52 52 41 2D 47 4E 49 4C 49 41 46 00' 09E1    578 ARRAY_MSG54::    .ASCIC   FAILING-ARRAYS: DC646,DC648,DC649
44 2C 36 34 36 43 44 20 3A 53 59 41  09ED
      39 34 36 43 44 2C 38 34 36 43  09F9
                                 21  09E1
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A03    579 ARRAY_MSG55::    .ASCIC   FAILING-ARRAYS: DC646,DC647,DC649
44 2C 36 34 36 43 44 20 3A 53 59 41  0A0F
      39 34 36 43 44 2C 37 34 36 43  0A1B
                                 21  0A03
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A25    580 ARRAY_MSG56::    .ASCIC   FAILING-ARRAYS: DC647,DC648,DC649
44 2C 37 34 36 43 44 20 3A 53 59 41  0A31
      39 34 36 43 44 2C 38 34 36 43  0A3D
                                 21  0A25
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A47    561 ARRAY_MSG57::    .ASCIC   FAILING-ARRAYS: DC648,DC649
44 2C 38 34 36 43 44 20 3A 53 59 41  0A53
               39 34 36 43 44 2C 37  0A5F
                                 1B  0A47
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A63    582 ARRAY_MSG58::    .ASCIC   FAILING-ARRAYS: DC647,DC649
44 2C 37 34 36 43 44 20 3A 53 59 41  0A6F
               39 34 36 43 44 2C 37  0A7B
                                 1B  0A63
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A7F    583 ARRAY_MSG59::    .ASCIC   FAILING-ARRAYS: DC646,DC649
44 2C 36 34 36 43 44 20 3A 53 59 41  0A8B
               39 34 36 43 44 2C 37  0A97
                                 1B  0A7F
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0A9B    584 ARRAY_MSG60::    .ASCIC   FAILING-ARRAYS: DC647,DC648
44 2C 37 34 36 43 44 20 3A 53 59 41  0AA7
               38 34 36 43 44 2C 37  0AB3
                                 1B  0A9B
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0AB7    585 ARRAY_MSG61::    .ASCIC   FAILING-ARRAYS: DC646,DC648
44 2C 36 34 36 43 44 20 3A 53 59 41  0AC3
               38 34 36 43 44 2C 37  0ACF
                                 1B  0AB7
52 52 41 2D 47 4E 49 4C 49 41 46 00' 0AD3    586 ARRAY_MSG62::    .ASCIC   FAILING-ARRAYS: DC646,DC647,DC648
44 2C 36 34 36 43 44 20 3A 53 59 41  0ADF
      38 34 36 43 44 2C 37 34 36 43  0AEB
                                 21  0AD3
                                     0AF5    587
                                     0AF5    588 ;++
                                     0AF5    589 ; DEFINE COMPONENT CALLOUT MESSAGE BLOCK
```

```
                                        0AF5    590 ;--
                                        0AF5    591
45 4E 4F 20 59 53 55 42 5F 54 44 00'    0AF5    592 COMP_MSG1::     .ASCIC   DT_BUSY ONE-SHOT TOO SLOW  -
4C 53 20 4F 4F 54 20 54 4F 48 53 2D     0B01
                        57 4F           0B0D
47 4E 49 4C 49 41 46 09 09 09 0D 0A     0B0F    593         <10><13><9><9><9> FAILING COMPONENT: DT_BUSY ONESHOT/TIMING COMP
20 3A 54 4E 45 4E 4F 50 4D 4F 43 20     0B1B
53 45 4E 4F 20 59 53 55 42 5F 54 44     0B27
43 20 47 4E 49 4D 49 54 2F 54 4F 48     0B33
                        50 4D 4F        0B3F
                                 4C     0AF5
45 4E 4F 20 59 53 55 42 5F 54 44 00'    0B42    594 COMP_MSG2::     .ASCIC   DT_BUSY ONE-SHOT TOO FAST  -
41 46 20 4F 4F 54 20 54 4F 48 53 2D     0B4E
                        54 53           0B5A
47 4E 49 4C 49 41 46 09 09 09 0D 0A     0B5C    595         <10><13><9><9><9> FAILING COMPONENT: DT_BUSY ONESHOT/TIMING COMP
20 3A 54 4E 45 4E 4F 50 4D 4F 43 20     0B68
53 45 4E 4F 20 59 53 55 42 5F 54 44     0B74
43 20 47 4E 49 4D 49 54 2F 54 4F 48     0B80
                        50 4D 4F        0B8C
                                 4C     0B42
                                        0B8F    596
                                        0B8F    597
                                        0B8F    598 ;++
                                        0B8F    599 ;
                                        0B8F    600 ; DEFINE FORMAT STATEMENTS FOR PRINTING REGISTER MNEMONICS
                                        0B8F    601 ;
                                        0B8F    602 ;--
                                        0B8F    603 FMT_CONTRL_REG::
30 33 54 49 42 2C 31 33 54 49 42 00'    0B8F    604                 .ASCIC   BIT31,BIT30,BIT29,BIT28,BIT27,BIT26,BIT25 -
38 32 54 49 42 2C 39 32 54 49 42 2C     0B9B
36 32 54 49 42 2C 37 32 54 49 42 2C     0BA7
                        35 32 54 49 42 2C       0BB3
33 32 54 49 42 2C 34 32 54 49 42 2C     0BB9    605                          ,BIT24,BIT23,BIT22,BIT21,BIT20,BIT19,BIT18, -
31 32 54 49 42 2C 32 32 54 49 42 2C     0BC5
39 31 54 49 42 2C 30 32 54 49 42 2C     0BD1
                        2C 38 31 54 49 42 2C    0BDD
2C 36 31 54 49 42 2C 37 31 54 49 42     0BE4    606                          BIT17,BIT16,BIT15,BIT14,BIT13,BIT12,BIT11,BIT10, -
2C 34 31 54 49 42 2C 35 31 54 49 42     0BF0
2C 32 31 54 49 42 2C 33 31 54 49 42     0BFC
2C 30 31 54 49 42 2C 31 31 54 49 42     0C08
49 42 2C 38 54 49 42 2C 39 54 49 42     0C14    607                          BIT9,BIT8,BIT7,BIT6,BIT5,IBC,MMM,IE,ABORT,INIT
35 54 49 42 2C 36 54 49 42 2C 37 54     0C20
2C 45 49 2C 4D 4D 4D 2C 43 42 49 2C     0C2C
      54 49 4E 49 2C 54 52 4F 42 41     0C38
                                 B2     0B8F
                                        0C42    608 FMT_STATUS_REG::
33 54 49 42 2C 59 53 55 42 54 44 00'    0C42    609                 .ASCIC   DTBUSY,BIT30,CRD,BIT28,BIT27,BIT26,BIT25,BIT24 -
2C 38 32 54 49 42 2C 44 52 43 2C 30     0C4E
2C 36 32 54 49 42 2C 37 32 54 49 42     0C5A
   34 32 54 49 42 2C 35 32 54 49 42     0C66
32 32 54 49 42 2C 33 32 54 49 42 2C     0C71    610                          ,BIT23,BIT22,BIT21,CBHUNG,PGE,NED,MCPE,ATTN,BIT15 -
4E 55 48 42 43 2C 31 32 54 49 42 2C     0C7D
43 4D 2C 44 45 4E 2C 45 47 50 2C 47     0C89
31 54 49 42 2C 4E 54 54 41 2C 45 50     0C95
                                 35     0CA1
50 4D 4F 43 5F 54 44 2C 45 50 53 2C     0CA2    611                          ,SPE,DT_COMP,DTABT,DLT,WCK_UP_ERR,WCK_LOW_ERR,MXF -
57 2C 54 4C 44 2C 54 42 41 54 44 2C     0CAE
```

```
                                                
```

```
      2C 38 54 49 42 2C 39 54 49 42  0EFF
49 42 2C 36 54 49 42 2C 37 54 49 42  0F09    625                          BIT7,BIT6,BIT5,BIT4,BIT3,BIT2,BIT1,BIT0
33 54 49 42 2C 34 54 49 42 2C 35 54  0F15
42 2C 31 54 49 42 2C 32 54 49 42 2C  0F21
                  30 54 49           0F2D
                     B4              0E7B
30 33 54 49 42 2C 31 33 54 49 42 00' 0F30    626 FMT_MBE_CR1::   .ASCIC   BIT31,BIT30,BIT29,BIT28,BIT27,BIT26,BIT25,BIT24, -
38 32 54 49 42 2C 39 32 54 49 42 2C  0F3C
36 32 54 49 42 2C 37 32 54 49 42 2C  0F48
34 32 54 49 42 2C 35 32 54 49 42 2C  0F54
                              2C     0F60
2C 32 32 54 49 42 2C 33 32 54 49 42  0F61    627                          BIT23,BIT22,BIT21,BIT20,BIT19,BIT18,BIT17,BIT16, -
2C 30 32 54 49 42 2C 31 32 54 49 42  0F6D
2C 38 31 54 49 42 2C 39 31 54 49 42  0F79
2C 36 31 54 49 42 2C 37 31 54 49 42  0F85
2C 34 31 54 49 42 2C 35 31 54 49 42  0F91    628                          BIT15,BIT14,BIT13,BIT12,DVA,BIT10,BIT9,BIT8, -
2C 32 31 54 49 42 2C 33 31 54 49 42  0F9D
49 42 2C 30 31 54 49 42 2C 41 56 44  0FA9
         2C 38 54 49 42 2C 39 54     0FB5
35 46 2C 36 54 49 42 2C 37 54 49 42  0FBD    629                          BIT7,BIT6,F5,F4,F3,F2,F1,GO
31 46 2C 32 46 2C 33 46 2C 34 46 2C  0FC9
                        4F 47 2C     0FD5
                           A7        0F30
30 33 54 49 42 2C 31 33 54 49 42 00' 0FD8    630 FMT_MBE_SR::    .ASCIC   BIT31,BIT30,BIT29,BIT28,BIT27,BIT26,BIT25,BIT24, -
38 32 54 49 42 2C 39 32 54 49 42 2C  0FE4
36 32 54 49 42 2C 37 32 54 49 42 2C  0FF0
34 32 54 49 42 2C 35 32 54 49 42 2C  0FFC
                              2C     1008
2C 32 32 54 49 42 2C 33 32 54 49 42  1009    631                          BIT23,BIT22,BIT21,BIT20,BIT19,BIT18,BIT17,BIT16, -
2C 30 32 54 49 42 2C 31 32 54 49 42  1015
2C 38 31 54 49 42 2C 39 31 54 49 42  1021
2C 36 31 54 49 42 2C 37 31 54 49 42  102D
31 54 49 42 2C 52 52 45 2C 41 54 41  1039    632                          ATA,ERR,BIT13,MOL,BIT11,BIT10,BIT9,DPR,DRY,BIT6, -
2C 31 31 54 49 42 2C 4C 4F 4D 2C 33  1045
44 2C 39 54 49 42 2C 30 31 54 49 42  1051
2C 36 54 49 42 2C 59 52 44 2C 52 50  105D
43 52 2C 41 50 44 52 2C 35 54 49 42  1069    633                          BIT5,RDPA,RCPA,BIT2,BIT1,RRUN
31 54 49 42 2C 32 54 49 42 2C 41 50  1075
                  4E 55 52 52 2C     1081
                              AD     0FD8
30 33 54 49 42 2C 31 33 54 49 42 00' 1086    634 FMT_MBE_ER::    .ASCIC   BIT31,BIT30,BIT29,BIT28,BIT27,BIT26.BIT25, -
38 32 54 49 42 2C 39 32 54 49 42 2C  1092
36 32 54 49 42 2C 37 32 54 49 42 2C  109E
                  2C 35 32 54 49 42  10AA
2C 33 32 54 49 42 2C 34 32 54 49 42  10B1    635                          BIT24,BIT23,BIT22,BIT21,BIT20,BIT19,BIT18, -
2C 31 32 54 49 42 2C 32 32 54 49 42  10BD
2C 39 31 54 49 42 2C 30 32 54 49 42  10C9
                  2C 38 31 54 49 42  10D5
2C 36 31 54 49 42 2C 37 31 54 49 42  10DB    636                          BIT17,BIT16,BIT15,BIT14,OPI,DTE,BIT11,BIT10, -
2C 34 31 54 49 42 2C 35 31 54 49 42  10E7
31 54 49 42 2C 45 54 44 2C 49 50 4F  10F3
         2C 30 31 54 49 42 2C 31     10FF
49 42 2C 38 54 49 42 2C 39 54 49 42  1107    637                          BIT9,BIT8,BIT7,RFAIL,RMBEXC,DPE,CPE,RMR,BIT1, -
42 4D 52 2C 4C 49 41 46 52 2C 37 54  1113
2C 45 50 43 2C 45 50 44 2C 43 58 45  111F
         2C 31 54 49 42 2C 52 4D 52  112B
                        46 4C 49     1134    638                          ILF
```

```
                                                  B0  1086
30 33 54 49 42 2C 31 33 54 49 42 00'  1137    639 FMT_MBE_ASR::    .ASCIC   BIT31,BIT30,BIT29,BIT28,BIT27,BIT26,BIT25, -
38 32 54 49 42 2C 39 32 54 49 42 2C   1143
36 32 54 49 42 2C 37 32 54 49 42 2C   114F
                  2C 35 32 54 49 42 2C 115B
2C 33 32 54 49 42 2C 34 32 54 49 42   1162    640                          BIT24,BIT23,BIT22,BIT21,BIT20,BIT19,BIT18, -
2C 31 32 54 49 42 2C 32 32 54 49 42   116E
2C 39 31 54 49 42 2C 30 32 54 49 42   117A
                     2C 38 31 54 49 42 1186
2C 36 31 54 49 42 2C 37 31 54 49 42   118C    641                          BIT17,BIT16,RS4,RS3,RS2,RS1,RS0,DS2,DS1,DS0, -
2C 32 53 52 2C 33 53 52 2C 34 53 52   1198
2C 32 53 44 2C 30 53 52 2C 31 53 52   11A4
               2C 30 53 44 2C 31 53 44 11B0
54 41 2C 36 41 54 41 2C 37 41 54 41   11B8    642                          ATA7,ATA6,ATA5,ATA4,ATA3,ATA2,ATA1,ATA0
33 41 54 41 2C 34 41 54 41 2C 35 41   11C4
41 2C 31 41 54 41 2C 32 41 54 41 2C   11D0
                              30 41 54 11DC
                                 A7   1137
                                      11DF    643 ;++
                                      11DF    644 ; DEFINE INFORMATION MESSAGE BLOCK
                                      11DF    645 ;--
45 49 46 49 54 4E 45 44 49 2F 21 00'  11DF    646 FMT_RHS::        .ASCIC   !/IDENTIFIED !UB RH(S).
2E 29 53 28 48 52 20 42 55 21 20 44   11EB
                                 17   11DF
42 55 21 20 30 35 37 48 52 2F 21 00'  11F7    647 FMT_LOCATION::   .ASCIC   !/RH750 !UB IS LOCATED AT !XL.
20 44 45 54 41 43 4F 4C 20 53 49 20   1203
               2E 4C 58 21 20 54 41   120F
                                 1E   11F7
20 45 53 41 42 20 53 41 21 2F 21 00'  1216    648 FMT_INIT1::      .ASCIC   !/!AS BASE ADDRESS IS: !XL
20 3A 53 49 20 53 53 45 52 44 44 41   1222
                              4C 58 21 122E
                                 1A   1216
                                      1231    649 ;++
                                      1231    650 ; DEFINE FATAL ERROR MESSAGE BLOCK
                                      1231    651 ;--
4E 41 4E 45 54 4E 49 41 4D 2F 21 00'  1231    652 FMT_FATAL_MMM:: .ASCIC   !/MAINTENANCE MODE MALFUNCTIONING
46 4C 41 4D 20 45 44 4F 4D 20 45 43   123D
      47 4E 49 4E 4F 49 54 43 4E 55   1249
                                 21   1231
                                      1253    653
                                      1253    654 ;++
                                      1253    655 ; DEFINE ERROR MESSAGE FORMAT BLOCK
                                      1253    656 ;--
4C 41 4E 52 45 54 4E 49 20 2F 21 00'  1253    657 FMT_IBUS1::      .ASCIC   !/ INTERNAL BUS ERROR !AC : STUCK AT ONE
21 20 52 4F 52 52 45 20 53 55 42 20   125F
41 20 4B 43 55 54 53 20 3A 20 43 41   126B
                        45 4E 4F 20 54 1277
                                 28   1253
20 4C 41 4E 52 45 54 4E 49 2F 21 00'  127C    658 FMT_IBUS0::      .ASCIC   !/INTERNAL BUS ERROR !AC : STUCK AT ZERO
41 21 20 52 4F 52 52 45 20 53 55 42   1288
54 41 20 4B 43 55 54 53 20 3A 20 43   1294
                        4F 52 45 5A 20 12A0
                                 28   127C
                                      12A5    659
42 55 21 5B 45 56 49 52 44 2F 21 00'  12A5    660 FMT_DRIVERES::   .ASCIC   !/DRIVE !UB  RESPONDED
   44 45 44 4E 4F 50 53 45 52 20 5D   12B1
                                 16   12A5
```

```
                                                12BC    661 FMT_DRIVENORES::
20 45 56 49 52 44 20 4F 4E 2F 21 00'            12BC    662                     .ASCIC  !/NO DRIVE RESPONDED.
      2E 44 45 44 4E 4F 50 53 45 52             12C8
                                  15            12BC
                                                12D2    663 FMT_ALLDRVPRES::
45 56 49 52 44 20 4C 4C 41 2F 21 00'            12D2    664                     .ASCIC  !/ALL DRIVES PRESENT-- -
   2D 2D 54 4E 45 53 45 52 50 20 53             12DE
4F 46 52 45 50 20 54 4F 4E 4E 41 43             12E9    665                             CANNOT PERFORM SUBTEST 2
32 20 54 53 45 54 42 55 53 20 4D 52             12F5
                                  2E            12D2
20 54 41 20 53 55 58 45 4E 2F 21 00'            1301    666 FMT_BADNEXUS::      .ASCIC  !/NEXUS AT LOCATION !XL RESPONDED -
4C 58 21 20 4E 4F 49 54 41 43 4F 4C             130D
      44 45 44 4E 4F 50 53 45 52 20             1319
41 20 4C 41 43 49 53 59 48 50 28 20             1323    667                              (PHYSICAL ADDRESS HAD BIT 12 SET).!/
42 20 44 41 48 20 53 53 45 52 44 44             132F
21 2E 29 54 45 53 20 32 31 20 54 49             133B
                                  2F            1347
                                  46            1301
20 52 45 46 53 4E 41 52 54 2F 21 00'            1348    668 FMT_XFERERR::       .ASCIC  !/TRANSFER ERROR!/PHYSICAL ADDRESS = !XL -
49 53 59 48 50 2F 21 52 4F 52 52 45             1354
20 53 53 45 52 44 44 41 20 4C 41 43             1360
                        4C 58 21 20 3D          136C
43 45 50 58 45 20 41 54 41 44 2F 21             1371    669                             !/DATA EXPECTED: !XL DATA  RECEIVED: !XL
54 41 44 20 4C 58 21 20 3A 44 45 54             137D
3A 44 45 56 49 45 43 45 52 20 20 41             1389
                        4C 58 21 20             1395
                                  50            1348
52 4F 52 52 45 20 52 41 56 2F 21 00'            1399    670 FMT_VARER::         .ASCIC  !/VAR ERROR /IMPROPER SEQUENCING/!/ -
53 20 52 45 50 4F 52 50 4D 49 2F 20             13A5
2F 21 2F 47 4E 49 43 4E 45 55 51 45             13B1
44 45 54 43 45 4C 45 53 20 50 41 4D             13BD    671                             MAP SELECTED = !XB  MAP EXPECTED = !XB
20 50 41 4D 20 20 42 58 21 20 3D 20             13C9
21 20 3D 20 44 45 54 43 45 50 58 45             13D5
                                  42 58         13E1
                                  49            1399
41 20 4C 41 55 54 52 49 56 2F 21 00'            13E3    672 FMT_VARINCERR:: .ASCIC  !/VIRTUAL ADDRESS INCREMENTAL ERROR
45 52 43 4E 49 20 53 53 45 52 44 44             13EF
52 4F 52 52 45 20 4C 41 54 4E 45 4D             13FB
                                  23            13E3
                                                1407    673 FMT_DUALWT_ERR::
45 52 20 50 41 4D 20 4C 41 55 44 00'            1407    674                     .ASCIC  DUAL MAP REGISTER WRITING ERROR
49 54 49 52 57 20 52 45 54 53 49 47             1413
               52 4F 52 52 45 20 47 4E          141F
                                  1F            1407
                                                1427    675 FMT_DUALRS_ERR::
44 4E 4F 50 53 45 52 20 54 55 55 00'            1427    676                     .ASCIC  UUT RESPONDED TO MORE THAN 1 ADDRESS
54 20 45 52 4F 4D 20 4F 54 20 44 45             1433
53 45 52 44 44 41 20 31 20 4E 41 48             143F
                                  53            144B
                                  24            1427
52 44 44 41 20 4C 41 55 44 2F 21 00'            144C    677 FMT_DUADR_ERR:: .ASCIC  !/DUAL ADDRESSING ERROR:!/WROTE MAP !UB AND MAP -
52 4F 52 52 45 20 47 4E 49 53 53 45             1458
50 41 4D 20 45 54 4F 52 57 2F 21 3A             1464
50 41 4D 20 44 4E 41 20 42 55 21 20             1470
43 41 20 4F 53 4C 41 20 42 55 21 20             147C    678                              !UB ALSO ACCEPTED DATA.
2E 41 54 41 44 20 44 45 54 50 45 43             1488
                                  47            144C
```

```
21 20 3D 20 29 43 41 21 28 2F 21 00'  1494   679 FMT_HEADER::     .ASCIC   !/(!AC) = !XL ERROR:
       20 3A 52 4F 52 52 45 20 4C 58   14A0
                                 15    1494
5D 42 58 21 5B 43 41 21 28 2F 21 00'  14AA   680 FMT_MAP_HEAD::   .ASCIC   !/(!AC !XB ) = !XL ERROR:
4F 52 52 45 20 4C 58 21 20 3D 20 29   14B6
                        20 3A 52      14C2
                                 1A   14AA
   43 41 21 20 20 43 41 21 2F 21 00'  14C5   681 FMT_BODY::       .ASCIC   !/!AC  !AC
                                 0A   14C5
         4F 52 45 5A 20 20 20 20 00'  14D0   682 FMT_NOTHING::    .ASCIC       ZERO
                                 08   14D0
21 20 4C 58 21 20 43 41 21 2F 21 00'  14D9   683 FMT_COMP_ERR::   .ASCIC   !/!AC !XL !AC !XL !AC !XL!/!/
21 20 43 41 21 20 4C 58 21 20 43 41   14E5
                  2F 21 2F 21 4C 58   14F1
                                 1D   14D9
4E 41 52 54 20 41 54 41 44 2F 21 00'  14F7   684 FMT_SILOERROR:: .ASCIC   !/DATA TRANSFER ABORT ERROR:!/EXPECTED -
45 20 54 52 4F 42 41 20 52 45 46 53   1503
43 45 50 58 45 2F 21 3A 52 4F 52 52   150F
                        44 45 54      151B
58 21 3A 44 52 4F 57 44 41 55 51 20   151E   685                          QUADWORD:!XL!XL !/RECEIVED QUADWORD:!XL!XL -
49 45 43 45 52 2F 21 20 4C 58 21 4C   152A
44 52 4F 57 44 41 55 51 20 44 45 56   1536
               4C 58 21 4C 58 21 3A   1542
54 41 20 44 45 54 41 43 4F 4C 20 20   1549   686                           LOCATED AT ADDRESS: !XL!/
58 21 20 3A 53 53 45 52 44 44 41 20   1555
                        2F 21 4C      1561
                                 6C   14F7
                                      1564   687 FMT_MISSEDXFER::
20 44 45 56 49 45 43 45 52 2F 21 00'  1564   688                    .ASCIC   !/RECEIVED MISSED TRANSFER WHILE ASSERTING -
53 4E 41 52 54 20 44 45 53 53 49 4D   1570
53 41 20 45 4C 49 48 57 20 52 45 46   157C
               47 4E 49 54 52 45 53   1588
44 20 44 4E 41 20 53 4B 4C 43 53 20   158F   689                           SCLKS AND DT_BUSY SET.
   2E 54 45 53 20 59 53 55 42 5F 54   159B
                                 41   1564
4E 4F 49 54 50 45 43 58 45 2F 21 00   15A6   690 FMT_EXCP_ERR::   .ASCIC   !/EXCEPTION FAILED TO RETURN TO ZERO AFTER EBL!/
52 20 4F 54 20 44 45 4C 49 41 46 20   15B2
52 45 5A 20 4F 54 20 4E 52 55 54 45   15BE
21 4C 42 45 20 52 45 54 46 41 20 4F   15CA
                                 2F   15D6
                                 30   15A6
                                      15D7   691 FMT_MAPFUN_ERR::
54 43 4E 55 46 20 50 41 4D 2F 21 00'  15D7   692                    .ASCIC   !/MAP FUNCTION ERROR:
   20 3A 52 4F 52 52 45 20 4E 4F 49   15E3
                                 16   15D7
20 44 45 54 43 45 50 58 45 2F 21 00'  15EE   693 FMT_MAP_EXERR:: .ASCIC   !/EXPECTED MAP = !XL   MAP SELECTED = !XL !/ -
20 20 20 4C 58 21 20 3D 20 50 41 4D   15FA
44 45 54 43 45 4C 45 53 20 50 41 4D   1606
         2F 21 20 4C 58 21 20 3D 20   1612
44 44 41 20 4C 41 43 49 53 59 48 50   161B   694                           PHYSICAL ADDRESS EXPECTED: !XL  ADDRESS -
45 54 43 45 50 58 45 20 53 53 45 52   1627
52 44 44 41 20 20 4C 58 21 20 3A 44   1633
                        53 53 45      163F
44 45 54 43 55 52 54 53 4E 4F 43 20   1642   695                           CONSTRUCTED: !XL
                  20 4C 58 21 20 3A   164E
                                 65   15EE
                                      1654   696 FMT_EXTREG_ERR::
```

```
20 4C 41 4E 52 45 54 58 45 2F 21 00'  1654   697                    .ASCIC   !/EXTERNAL REGISTER !XL  = !XW DATA ERROR
4C 58 21 5B 52 45 54 53 49 47 45 52   1660
41 54 41 44 20 57 58 21 20 3D 20 5D   166C
                  52 4F 52 52 45 20   1678
                                 29   1654
                                      167E   698 FMT_MBEREG_ERR::
53 49 47 45 52 20 45 42 4D 2F 21 00'  167E   699                    .ASCIC   !/MBE REGISTER !XL  = !XW UNEXPECTED VALUE
21 20 3D 20 5D 4C 58 21 5B 52 45 54   168A
45 54 43 45 50 58 45 4E 55 20 57 58   1696
               45 55 4C 41 56 20 44   16A2
                                 2A   167E
50 4D 4F 43 20 41 54 41 44 2F 21 00'  16A9   700 FMT_DTERR::        .ASCIC   !/DATA COMPARE ERROR::!/EXPECTED:: !XL -
21 3A 3A 52 4F 52 52 45 20 45 52 41   16B5
20 3A 3A 44 45 54 43 45 50 58 45 2F   16C1
                              4C 58   16CD
20 3A 3A 44 45 56 49 45 43 45 52 20   16D0   701                     RECEIVED:: !XL -
                              4C 58   16DC
54 43 45 4C 45 53 20 50 41 4D 2F 21   16DF   702                    !/MAP SELECTED = !UB VIRTUAL ADDRESS = !XL
52 49 56 20 42 55 21 20 3D 20 44 45   16EB
53 53 45 52 44 44 41 20 4C 41 55 54   16F7
                  4C 58 21 20 3D 20   1703
                                 5F   16A9
50 4D 4F 43 20 41 54 41 44 2F 21 00'  1709   703 FMT_DCMPERR::      .ASCIC   !/DATA COMPARE ERROR::!/EXPECTED:: -
21 3A 3A 52 4F 52 52 45 20 45 52 41   1715
   3A 3A 44 45 54 43 45 50 58 45 2F   1721
45 56 49 45 43 45 52 20 42 58 21 20   172C   704                     !XB RECEIVED:: !XB -
                  42 58 21 20 3A 3A   1738
41 20 53 59 48 50 20 50 58 45 2F 21   173F   705                    !/EXP PHYS ADDRESS:: !XL  REC PHYS ADDRESS:: !XL -
4C 58 21 20 3A 3A 53 53 45 52 44 44   174B
41 20 53 59 48 50 20 43 45 52 20 20   1757
4C 58 21 20 3A 3A 53 53 45 52 44 44   1763
45 42 4D 55 4E 20 45 54 59 42 20 20   176F   706                       BYTE NUMBER:: !ZL
                  4C 5A 21 20 3A 3A   177B
                                 78   1709
54 41 44 20 4C 41 54 4F 54 2F 21 00'  1782   707 FMT_ERRSUM::       .ASCIC   !/TOTAL DATA COMPARE ERRORS:: !ZL
52 45 20 45 52 41 50 4D 4F 43 20 41   178E
      4C 5A 21 20 3A 3A 53 52 4F 52   179A
                                 21   1782
                                      17A4   708 FMT_CMICMD_ERR::
41 4D 4D 4F 43 20 49 4D 43 2F 21 00'  17A4   709                    .ASCIC   !/CMI COMMAND ERROR
               52 4F 52 52 45 20 44   17B0
                                 13   17A4
                                      17B8   710
                                      17B8   711 FMT_BYTMSK_ER::
20 45 54 59 42 20 52 41 43 2F 21 00'  17B8   712                    .ASCIC   !/CAR BYTE MASK ERROR::
3A 3A 52 4F 52 52 45 20 4B 53 41 4D   17C4
                                 17   17B8
                                      17D0   713
52 4F 43 4E 49 20 52 43 42 2F 21 00'  17D0   714 FMT_IBCER1::       .ASCIC   !/BCR INCORRECT
                           54 43 45   17DC
                                 0F   17D0
52 4F 43 4E 49 20 52 41 56 2F 21 00'  17E0   715 FMT_IBCER2::       .ASCIC   !/VAR INCORRECT
                           54 43 45   17EC
                                 0F   17E0
47 50 20 3E 39 31 3C 52 53 2F 21 00'  17F0   716 FMT_IBCER3::       .ASCIC   !/SR<19> PGE DID NOT SET
45 53 20 54 4F 4E 20 44 49 44 20 45   17FC
                                 54   1808
```

```
                                                18  17F0
52 4F 43 4E 49 20 52 43 42 2F 21                00' 1809    717 FMT_IBCER4::    .ASCIC   !/BCR INCORRECT
                                  54 43 45 52       1815
                                                0F  1809
54 50 55 52 52 45 54 4E 49 2F 21                00' 1819    718 FMT_IBCER5::    .ASCIC   !/INTERRUPT SHOULD HAVE OCCURRED AT BCR OVERFLOW
45 56 41 48 20 44 4C 55 4F 48 53 20             1825
54 41 20 44 45 52 52 55 43 43 4F 20             1831
4F 4C 46 52 45 56 4F 20 52 43 42 20             183D
                                                57  1849
                                                30  1819
41 44 20 3E 31 33 3C 52 53 2F 21                00' 184A    719 FMT_IBCER6::    .ASCIC   !/SR<31> DATA BUSY DID NOT SET
20 44 49 44 20 59 53 55 42 20 41 54             1856
                  54 45 53 20 54 4F 4E          1862
                                                1E  184A
54 50 55 52 52 45 54 4E 49 2F 21                00' 1869    720 FMT_IBCER7::    .ASCIC   !/INTERRUPT SHOULD NOT HAVE OCCURRED -
20 54 4F 4E 20 44 4C 55 4F 48 53 20             1875
45 52 52 55 43 43 4F 20 45 56 41 48             1881
                                                44  188D
4D 20 4F 54 20 45 54 49 52 57 2F 21             188E    721                         !/WRITE TO MAP REGISTER SHOULD CLEAR INTER. CONDITION
20 52 45 54 53 49 47 45 52 20 50 41             189A
52 41 45 4C 43 20 44 4C 55 4F 48 53             18A6
44 4E 4F 43 20 2E 52 45 54 4E 49 20             18B2
                        4E 4F 49 54 49          18BE
                                                59  1869
4C 55 4F 48 53 20 45 47 50 2F 21                00' 18C3    722 FMT_IBCER8::    .ASCIC   !/PGE SHOULD NOT HAVE OCCURRED
4F 20 45 56 41 48 20 54 4F 4E 20 44             18CF
                  44 45 52 52 55 43 43          18DB
                                                1E  18C3
53 49 47 45 52 20 50 41 4D 2F 21                00' 18E2    723 FMT_IBCER9::    .ASCIC   !/MAP REGISTER 0 WAS NOT WRITTEN IN IBC MODE
4F 4E 20 53 41 57 20 30 20 52 45 54             18EE
4E 49 20 4E 45 54 54 49 52 57 20 54             18FA
            45 44 4F 4D 20 43 42 49 20          1906
                                                2C  18E2
                                                    190F    724
3A 44 45 54 43 45 50 58 45 2F 21                00' 190F    725 FMT_VAL::       .ASCIC   !/EXPECTED::!XL  RECEIVED::!XL  XOR::!XL!/
56 49 45 43 45 52 20 20 4C 58 21 3A             191B
52 4F 58 20 20 4C 58 21 3A 3A 44 45             1927
                  2F 21 4C 58 21 3A 3A          1933
                                                2A  190F
                                                    193A    726
50 20 45 42 4D 20 4F 4E 20 2F 21                00' 193A    727 FMT_NO_MBE::    .ASCIC   !/ NO MBE PRESENT ON THIS MASSBUS
48 54 20 4E 4F 20 54 4E 45 53 45 52             1946
      53 55 42 53 53 41 4D 20 53 49             1952
                                                21  193A
53 20 45 42 4D 20 4F 4E 20 2F 21                00' 195C    728 FMT_NO_MBE_SEL::.ASCIC   !/ NO MBE SELECTED ON THIS MASSBUS
54 20 4E 4F 20 44 45 54 43 45 4C 45             1968
   53 55 42 53 53 41 4D 20 53 49 48             1974
                                                22  195C
                        45 42 4D                00' 197F    729 ASCII_MBE::     .ASCIC   MBE
                                                03  197F
                              20                00' 1983    730 MBE_REG_MSG::   .ASCIC
                                                01  1983
                              20                00' 1985    731 EXT_REG_MSG::   .ASCIC
                                                01  1985
                              20                00' 1987    732 BLANK::         .ASCIC
                                                01  1987
45 54 43 45 4C 45 53 20 45 42 4D                00' 1989    733 NO_MBE_MSG::    .ASCIC   MBE SELECTED, NONE PRESENT
```

```
53 45 52 50 20 45 4E 4F 4E 20 2C 44  1995
                  20 54 4E 45  19A1
                        1B  1989
54 54 49 52 57 20 47 45 52 2F 21 00' 19A5   734 FMT_DUALWT_MSG::.ASCIC   !/REG WRITTEN::!XL  REG CHANGED::!XL <10><13>-
47 45 52 20 20 4C 58 21 3A 3A 4E 45  19B1
58 21 3A 3A 44 45 47 4E 41 48 43 20  19BD
                        0A 4C  19C9
54 54 49 52 57 20 47 45 52 20 20 0D  19CB   735                          REG WRITTEN DATA::!XL  REG CHANGED DATA::!XL!/
4C 58 21 3A 3A 41 54 41 44 20 4E 45  19D7
45 47 4E 41 48 43 20 47 45 52 20 20  19E3
21 4C 58 21 3A 3A 41 54 41 44 20 44  19EF
                        2F  19FB
                        56  19A5
45 52 44 44 41 20 54 55 55 2F 21 00' 19FC   736 FMT_DUALRS_MSG::.ASCIC   !/UUT ADDRESS::!XL  DUAL RESPONDING ADDRESS::!XL  XOR::!XL!
41 55 44 20 20 4C 58 21 3A 3A 53 53  1A08
47 4E 49 44 4E 4F 50 53 45 52 20 4C  1A14
58 21 3A 3A 53 53 45 52 44 44 41 20  1A20
21 4C 58 21 3A 3A 52 4F 58 20 20 4C  1A2C
                        2F  1A38
                        3C  19FC
   54 4C 55 41 46 20 41 20 53 49 00' 1A39   737 SA0_MSG::        .ASCIC   IS A FAULT
                        0A  1A39
   54 4C 55 41 46 20 41 20 53 49 00' 1A44   738 SA1_MSG::        .ASCIC   IS A FAULT
                        0A  1A44
44 45 54 43 45 50 58 45 4E 55 20 00' 1A4F   739 UNEXPECTED::     .ASCIC    UNEXPECTED
                        20  1A5B
                        0C  1A4F
20 3A 44 45 56 49 45 43 45 52 20 00' 1A5C   740 REC_MSG::        .ASCIC    RECEIVED:
                        0B  1A5C
20 3A 44 45 54 43 45 50 58 45 20 00' 1A68   741 EXP_MSG::        .ASCIC    EXPECTED:
                        0B  1A68
20 20 20 20 20 20 3A 52 4F 58 20 00' 1A74   742 DIF_MSG::        .ASCIC    XOR:
                        0B  1A74
            20 52 43 48 52 20 00' 1A80   743 RHCR_MSG::        .ASCIC    RHCR
                        06  1A80
            20 52 53 48 52 20 00' 1A87   744 RHSR_MSG::        .ASCIC    RHSR
                        06  1A87
         20 52 41 56 48 52 20 00' 1A8E   745 RHVAR_MSG::       .ASCIC    RHVAR
                        07  1A8E
         20 52 43 42 48 52 20 00' 1A96   746 RHBCR_MSG::       .ASCIC    RHBCR
                        07  1A96
            20 52 44 48 52 20 00' 1A9E   747 RHDR_MSG::        .ASCIC    RHDR
                        06  1A9E
      20 52 50 41 4D 48 52 20 00' 1AA5   748 RHMAPR_MSG::      .ASCIC    RHMAPR
                        08  1AA5
         20 3A 5D 4C 58 21 5B 00' 1AAE   749 INDEX_MSG::      .ASCIC    !XL :
                        07  1AAE
54 53 49 58 45 2D 4E 4F 4E 2F 21 00' 1AB6   750 FMT_NX_ER::      .ASCIC   !/NON-EXISTENT REGISTER, FAILING ADDRESS=!XL
52 45 54 53 49 47 45 52 20 54 4E 45  1AC2
44 41 20 47 4E 49 4C 49 41 46 20 2C  1ACE
         4C 58 21 3D 53 53 45 52 44  1ADA
                        2C  1AB6
54 50 55 52 52 45 54 4E 49 2F 21 00' 1AE3   751 FMT_VECERR::     .ASCIC   !/INTERRUPT VECTOR ADDRESS IS INCORREC
52 44 44 41 20 52 4F 54 43 45 56 20  1AEF
52 4F 43 4E 49 20 53 49 20 53 53 45  1AFB
                  20 54 43 45 52  1B07
                        28  1AE3
```

```
54 43 45 52 52 4F 43 4E 49 2F 21 00'  1B0C    752 FMT_SERR::      .ASCIC   !/INCORRECT DATA TRANSFER TO MEMORY
46 53 4E 41 52 54 20 41 54 41 44 20   1B18
59 52 4F 4D 45 4D 20 4F 54 20 52 45   1B24
                                 20   1B30
                                 24   1B0C
27 30 35 37 48 52 20 4F 4E 2F 21 00'  1B31    753 FMT_NO_RH750::  .ASCIC   !/NO RH750'S SELECTED FOR TEST
46 20 44 45 54 43 45 4C 45 53 20 53   1B3D
                  54 53 45 54 20 52 4F  1B49
                                 1E   1B31
                                      1B50    754
                                      1B50    755 ;++
                                      1B50    756 ; DEFINE INTERRUPT INFORMATION MESSAGES
                                      1B50    757 ;--
45 54 43 45 50 58 45 4E 55 2F 21 00'  1B50    758 FMT_UNEXP_INT:: .ASCIC   !/UNEXPECTED INTERRUPT OCCURRED AT PC = !XL!/
20 54 50 55 52 52 45 54 4E 49 20 44   1B5C
20 54 41 20 44 45 52 52 55 43 43 4F   1B68
   20 2F 21 4C 58 21 20 3D 20 43 50   1B74
                                 2E   1B50
54 50 55 52 52 45 54 4E 49 2F 21 00'  1B7F    759 FMT_INTFAILED:: .ASCIC   !/INTERRUPT FAILED FOR STATUS -
20 52 4F 46 20 44 45 4C 49 41 46 20   1B8B
                  53 55 54 41 54 53   1B97
4F 43 20 52 45 54 53 49 47 45 52 20   1B9D    760                          REGISTER CONDITION: !/!AC!/
21 2F 21 20 3A 4E 4F 49 54 49 44 4E   1BA9
                        2F 21 43 41   1BB5
                                 39   1B7F
54 4F 4E 20 44 4C 55 4F 43 2F 21 00'  1BB9    761 FMT_NOINTEST::  .ASCIC   !/COULD NOT TEST INTERRUPT ASSOCIATED -
52 52 45 54 4E 49 20 54 53 45 54 20   1BC5
54 41 49 43 4F 53 53 41 20 54 50 55   1BD1
                                 44 45   1BDD
58 45 20 4E 4F 4E 20 48 54 49 57 20   1BDF    762                          WITH NON EXISTENT DRIVE.!/
```

```
45 56 49 52 44 20 54 4E 45 54 53 49  1BEB
                  2F 21 2E           1BF7
                        40           1BB9
                                     1BFA   763          .SBTTL  BUFFERS
                                 00000000   764          .PSECT  BUFFERS, PAGE, NOEXE, RD, WRT
                             00000200  0000   765 DATA_!BUFFER::  .BLKB   512
                             00000400  0200   766 DATA_OBUFFER::  .BLKB   512
                             00000600  0400   767 MIOBUFFER::     .BLKB   512
                                       0600   768
                                   00001BFA   769          .PSECT  OWN, BYTE
                             00000110  1BFA   770 RH_VECS::        .LONG   X110            ; RH0 VECTOR
                             00000150  1BFE   771                 .LONG   X150            ; RH0 VECTOR
                             00000190  1C02   772                 .LONG   X190            ; RH0 VECTOR
                             000001D0  1C06   773                 .LONG   X1D0            ; RH0 VECTOR
                             00000114  1C0A   774                 .LONG   X114            ; RH1 VECTOR
                             00000154  1C0E   775                 .LONG   X154            ; RH1 VECTOR
                             00000194  1C12   776                 .LONG   X194            ; RH1 VECTOR
                             000001D4  1C16   777                 .LONG   X1D4            ; RH1 VECTOR
                             00000118  1C1A   778                 .LONG   X118            ; RH2 VECTOR
                             00000158  1C1E   779                 .LONG   X158            ; RH2 VECTOR
                             00000198  1C22   780                 .LONG   X198            ; RH2 VECTOR
                             000001D8  1C26   781                 .LONG   X1D8            ; RH2 VECTOR
                             0000011C  1C2A   782                 .LONG   X11C            ; RH3 VECTOR
                             0000015C  1C2E   783                 .LONG   X15C            ; RH3 VECTOR
                             0000019C  1C32   784                 .LONG   X19C            ; RH3 VECTOR
                             000001DC  1C36   785                 .LONG   X1DC            ; RH3 VECTOR
                                       1C3A   786
                             00000000  1C3A   787 TEST_VEC::      .LONG   0               ; USED FOR INTERRUPT TEST T17_S1
                                       1C3E   788
                                       1C3E   789 ;
                                       1C3E   790 ; THE FOLLOWING TABLE CONTAINS THE ADDRESS OF ALL THE INTERRUPT SERVICE
                                       1C3E   791 ; ROUTINES FOR EACH OF THE POSSIBLE INTERRUPT LEVELS
                                       1C3E   792 ;
                             00000000' 1C3E   793 ISR_TBL::       .LONG   ISR0
                             00000000' 1C42   794                 .LONG   ISR1
                             00000000' 1C46   795                 .LONG   ISR2
                             00000000' 1C4A   796                 .LONG   ISR3
                             00000000' 1C4E   797                 .LONG   ISR4
                             00000000' 1C52   798                 .LONG   ISR5
                             00000000' 1C56   799                 .LONG   ISR6
                             00000000' 1C5A   800                 .LONG   ISR7
                             00000000' 1C5E   801                 .LONG   ISR8
                             00000000' 1C62   802                 .LONG   ISR9
                             00000000' 1C66   803                 .LONG   ISR10
                             00000000' 1C6A   804                 .LONG   ISR11
                             00000000' 1C6E   805                 .LONG   ISR12
                             00000000' 1C72   806                 .LONG   ISR13
                             00000000' 1C76   807                 .LONG   ISR14
                             00000000' 1C7A   808                 .LONG   ISR15
```

```
                     1C7E    809 ;++
                     1C7E    810 ; THE FOLLOWING 16 INTERRUPT ROUTINES ARE USED IN TEST 17 SUBTEST 1 TO
                     1C7E    811 ; CHECK FOR THE CORRECT INTERRUPT VECTOR ADDRESS.
                     1C7E    812 ;--
                     1C7E    813
                 00000000    814         .PSECT  ISR0, LONG
00001C3A'EF  00000110 8F    D0  0000    815 ISR0::  MOVL    # X110,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    816         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    817         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    818         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    819         REI                             ; RETURN FROM INTERRUPT
                               001D    620
                 00000000    821         .PSECT  ISR1, LONG
00001C3A'EF  00000150 8F    D0  0000    822 ISR1::  MOVL    # X150,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    823         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    824         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    825         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    826         REI                             ; RETURN FROM INTERRUPT
                               001D    827
                 00000000    828         .PSECT  ISR2, LONG
00001C3A'EF  00000190 8F    D0  0000    829 ISR2::  MOVL    # X190,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    830         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    831         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    832         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    833         REI                             ; RETURN FROM INTERRUPT
                               001D    834
                 00000000    835         .PSECT  ISR3, LONG
00001C3A'EF  000001D0 8F    D0  0000    836 ISR3::  MOVL    # X1D0,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    837         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    838         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    839         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    840         REI                             ; RETURN FROM INTERRUPT
                               001D    841
                 00000000    842         .PSECT  ISR4, LONG
00001C3A'EF  00000114 8F    D0  0000    843 ISR4::  MOVL    # X114,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    844         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    845         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    846         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    847         REI                             ; RETURN FROM INTERRUPT
                               001D    848
                 00000000    849         .PSECT  ISR5, LONG
00001C3A'EF  00000154 8F    D0  0000    850 ISR5::  MOVL    # X154,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    851         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    852         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    853         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    854         REI                             ; RETURN FROM INTERRUPT
                               001D    855
                 00000000    856         .PSECT  ISR6, LONG
00001C3A'EF  00000194 8F    D0  0000    857 ISR6::  MOVL    # X194,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF    94  000B    858         CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
      00000053'EF    01    90  0011    859         MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
            04 A2    04    CA  0018    860         BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C    861         REI                             ; RETURN FROM INTERRUPT
```

```
                    00000000   862          .PSECT  ISR7, LONG
00001C3A'EF  000001D4 8F   D0  0000   863 ISR7::  MOVL   # X1D4,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   864          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   865          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   866          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   867          REI                           ; RETURN FROM INTERRUPT
                                 001D   868
                    00000000   869          .PSECT  ISR8, LONG
00001C3A'EF  00000118 8F   D0  0000   870 ISR8::  MOVL   # X118,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   871          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   872          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   873          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   874          REI                           ; RETURN FROM INTERRUPT
                                 001D   875
                    00000000   876          .PSECT  ISR9, LONG
00001C3A'EF  00000158 8F   D0  0000   877 ISR9::  MOVL   # X158,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   878          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   879          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   880          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   881          REI                           ; RETURN FROM INTERRUPT
                                 001D   882
                    00000000   883          .PSECT  ISR10, LONG
00001C3A'EF  00000198 8F   D0  0000   884 ISR10:: MOVL   # X198,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   885          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   886          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   887          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   888          REI                           ; RETURN FROM INTERRUPT
                                 001D   889
                    00000000   890          .PSECT  ISR11, LONG
00001C3A'EF  000001D8 8F   D0  0000   891 ISR11:: MOVL   # X1D8,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   892          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   893          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   894          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   895          REI                           ; RETURN FROM INTERRUPT
                                 001D   896
                    00000000   897          .PSECT  ISR12, LONG
00001C3A'EF  0000011C 8F   D0  0000   898 ISR12:: MOVL   # X11C,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   899          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   900          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   901          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   902          REI                           ; RETURN FROM INTERRUPT
                                 001D   903
                    00000000   904          .PSECT  ISR13, LONG
00001C3A'EF  0000015C 8F   D0  0000   905 ISR13:: MOVL   # X15C,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   906          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   907          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   908          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   909          REI                           ; RETURN FROM INTERRUPT
                                 001D   910
                    00000000   911          .PSECT  ISR14, LONG
00001C3A'EF  0000019C 8F   D0  0000   912 ISR14:: MOVL   # X19C,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF   94  000B   913          CLRB   EXP_INTERRUPT          ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF   01   90  0011   914          MOVB   #1,INT_OCCURRED        ; SET INTERRUPT EVENT FLAG
                04 A2   04   CA  0018   915          BICL   #INTERRUPT_EN,CR(R2)   ; CLEAR IE BIT
                             02  001C   916          REI                           ; RETURN FROM INTERRUPT
```

```
                        00000000   917            .PSECT  ISR15, LONG
00001C3A'EF  000001DC 8F  D0  0000   918 ISR15:: MOVL    # X1DC,TEST_VEC          ; LOAD RH INTERRUPT VECTOR
             00000052'EF  94  000B   919          CLRB    EXP_INTERRUPT           ; CLEAR INTERRUPT EXPECTED FLAG
          00000053'EF  01  90  0011   920          MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
                04 A2  04  CA  0018   921          BICL    #INTERRUPT_EN,CR(R2)    ; CLEAR IE BIT
                           02  001C   922          REI                             ; RETURN FROM INTERRUPT
```

```
                          001D   923 $DS_BGNSUMMARY
                          001D                       .SAVE
                      00000000                       .PSECT  SUMMARY,LONG
                          0000       SUMMARY:
                     0000 0000                       .WORD    M<>              ; ENTRY MASK
                          0002   924 $DS_ENDSUMMARY
                          0002       SUMMARY_X:
        00010058 9F   6E  FA 0002                    CALLG   (SP), a#DS$BREAK
                       04 0009                       RET                       ; RETURN TO COMMAND MODE
                      0000001D                       .RESTORE
```

```
                     001D   926          .SBTTL  PRINT_NULL ROUTINE
                  00000000  927          .PSECT  PRINT_NULL, WORD
                     0000   928 ;++
                     0000   929 ; FUNCTIONAL DESCRIPTION:
                     0000   930 ;
                     0000   931 ;       THIS ROUTINE IS CALLED BY THE SUPERVISOR FROM ERRHARD TO REPORT
                     0000   932 ;       NO MBE  WHEN  IT HAS BEEN  SELECTED BY THE USER.  IT REPORTS NO
                     0000   933 ;       DATA, AND ONLY A MESSAGE IS PRINTED.
                     0000   934 ;
                     0000   935 ; CALLING SEQUENCE:
                     0000   936 ;
                     0000   937 ;       CALLED BY THE DIAGNOSTIC SUPERVISOR
                     0000   938 ;
                     0000   939 ; INPUT PARAMETERS:
                     0000   940 ;
                     0000   941 ;       NONE
                     0000   942 ;
                     0000   943 ; IMPLICIT INPUTS:
                     0000   944 ;
                     0000   945 ;       NONE
                     0000   946 ;
                     0000   947 ; OUTPUT PARAMETERS:
                     0000   948 ;
                     0000   949 ;       NONE
                     0000   950 ;
                     0000   951 ; IMPLICIT OUTPUTS:
                     0000   952 ;
                     0000   953 ;       NONE
                     0000   954 ;
                     0000   955 ; COMPLETION CODES:
                     0000   956 ;
                     0000   957 ;       NONE
                     0000   958 ;
                     0000   959 ; SIDE EFFECTS:
                     0000   960 ;
                     0000   961 ;       NONE
                     0000   962 ;
                     0000   963 ;--
                     0000   964 PRINT_NULL::
                0000 0000   965          .WORD   0              ; REGISTER MASK OF REGISTERS TO SAVE    ;DPM001
                     0002   966 PRINT_NULL_X::                                                          ;DPM001
                  04 0002   967          RET                    ; EXIT - NO DATA TO PRINT               ;DPM001
```

```
                    00000000          .PSECT  PRINT_SBE, WORD
                        0000          .SBTTL  REPORT AND PRINT MODULE
                        0000  PRINT_SBE::
                  01C0  0000                  .WORD    M<R6,R7,R8>
                        0002  ;++
                        0002  ; THIS ROUTINE IS THE ERROR REPORT SUBROUTINE
                        0002  ; IT PRINTS THE ERROR HEADER WHICH CONSISTS OF THE NAME OF THE
                        0002  ; REGISTER FOLLOWED BY THE CONTENTS OF THAT REGISTER.
                        0002  ; IF THE REGISTER IS A MAP REGISTER THEN THE
                        0002  ; MAP REGISTER NUMBER IS PRINTED IN BETWEEN    .
                        0002  ; IF THE REGISTER IS AN EXTERNAL REGISTER THE ADDRESS
                        0002  ; OF THAT REGISTER IS PRINTED BETWEEN    .
                        0002  ; IF THE REGISTER IS AN MBE REGISTER THEN THE ADDRESS
                        0002  ; OF THAT REGISTER IS PRINTED BETWEEN    .
                        0002  ;
                        0002  ;
                        0002  ; LOCATION REG_NO DETERMINES THE FORMAT TO USE:
                        0002  ;
                        0002  ;       0 <= REG_NO <= 9  => INTERNAL RH750 REGISTER ERROR
                        0002  ;
                        0002  ;               REG_NO = 10 => MAP REGISTER ERROR
                        0002  ;               REG_NO = 11 => EXTERNAL REGISTER ERROR
                        0002  ;               REG_NO = 12 => MBE REGISTER ERROR
                        0002  ;
                        0002  ;
                        0002  ; R3 AND R4 ARE COMPARED AND ANY DIFFERENCE IS DISPLAYED IN
                        0002  ; THE SUBSEQUENT LINE FOLLOWING THE BIT MNEMONICS. IF
                        0002  ; THERE IS NO DIFFERENCE THEN NOTHING IS TYPED.
                        0002  ;--
000002B2'EF    0A   D1  0002          CMPL    #10,REG_NO                ; IS THIS A MAP REGISTER ERROR
               19   12  0009          BNEQ    10$                       ; IF NOT, PRINT NORMAL ERROR HEADER
                        000B          $DS_PRINTB_S    FMT_MAP_HEAD,-    ; PRINT MAP REGISTER  INDEX  =
                        000B                          REG_NAME,R10,R3   ; VALUE OF R3
               53   DD  000B                  PUSHL   R3
               5A   DD  000D                  PUSHL   R10
     000002AE'EF    DD  000F                  PUSHL   REG_NAME
     000014AA'EF    9F  0015                  PUSHAB  FMT_MAP_HEAD
000100E0 9F    04   FB  001B                  CALLS   #$$N, @#DS$PRINTB
               5A   11  0022          BRB     30$                       ; SKIP REGULAR REGISTER ERROR HEADER
000002B2'EF    0B   D1  0024  10$:    CMPL    #11,REG_NO                ; IS THIS AN EXTERNAL REGISTER
               39   13  002B          BEQL    20$                       ; BRANCH IF IT IS
000002B2'EF    0C   D1  002D          CMPL    #12,REG_NO                ; IS THIS AN MBE REGISTER?
               19   12  0034          BNEQ    15$                       ; SKIP IF NOT
         57    56   D0  0036          MOVL    R6,R7
         57    58   C0  0039          ADDL    R8,R7                     ; CALCULATE MBE REGISTER ADDRESS
                        003C          $DS_PRINTB_S    FMT_MBEREG_ERR,R7,R3    ; PRINT MBE ERROR HEADER
               53   DD  003C                  PUSHL   R3
               57   DD  003E                  PUSHL   R7
     0000167E'EF    9F  0040                  PUSHAB  FMT_MBEREG_ERR
000100E0 9F    03   FB  0046                  CALLS   #$$N, @#DS$PRINTB
               2F   11  004D          BRB     30$                       ; SKIP EXTERNAL REGISTER HANDLING
                        004F  15$:    $DS_PRINTB_S    FMT_HEADER,-      ; PRINT ERROR HEADER
                        004F                          REG_NAME,R3
               53   DD  004F                  PUSHL   R3
     000002AE'EF    DD  0051                  PUSHL   REG_NAME
     00001494'EF    9F  0057                  PUSHAB  FMT_HEADER
000100E0 9F    03   FB  005D                  CALLS   #$$N, @#DS$PRINTB
```

```
                  18   11  0064  1012          BRB     30$                          ; SKIP EXTERNAL REGISTER
                           0066  1013                                               ; ADDRESS CALCULATION
        57   56   04   C5  0066  1014 20$:     MULL3   #4,R6,R7                     ; MULTIPLY EXT REG INDEX BY 4
             57   58   C0  006A  1015          ADDL    R8,R7                        ; CALCULATE EXT REGISTER ADDRESS
                           006D  1016          $DS_PRINTB_S    FMT_EXTREG_ERR,R7,R3     ; PRINT EXTERNAL REGISTER
                  53   DD  006D                          PUSHL   R3
                  57   DD  006F                          PUSHL   R7
        00001654'EF   9F  0071                          PUSHAB  FMT_EXTREG_ERR
   000100E0 9F    03   FB  0077                          CALLS   #$$N, @#DS$PRINTB
                           007E  1017 30$:                                          ; ERROR HEADER
        57   54   53   CD  007E  1018 40$:     XORL3   R3,R4,R7                     ; FIND DIFFERENCE BETWEEN EXPECTED
                           0082  1019                                               ; AND RECEIVED VALUES
                  54   D5  0082  1020          TSTL    R4                           ; IF EXPECTED VALUE IS NON ZERO
                  1C   12  0084  1021          BNEQ    50$                          ; BRANCH TO CONVERT REGISTER BITS
                           0086  1022          $DS_PRINTX_S    FMT_BODY,-           ; DISPLAY 'ZERO'
                           0086  1023                          #EXP_MSG,-
                           0086  1024                          #FMT_NOTHING
        000014D0'8F   DD  0086                          PUSHL   #FMT_NOTHING
        00001A68'8F   DD  008C                          PUSHL   #EXP_MSG
        000014C5'EF   9F  0092                          PUSHAB  FMT_BODY
   000100E8 9F    03   FB  0098                          CALLS   #$$N, @#DS$PRINTX
                0042   31  009F  1025          BRW     60$                          ; SKIP TO TEST RECEIVED VALUE
                           00A2  1026 50$:     $DS_CVTREG_S    #31,R4,-             ; CONVERT REGISTER BITS TO MNEMONICS
                           00A2  1027                          @REG_STRING,-
                           00A2  1028                          REPORT_BUFFER,-
                           00A2  1029                          #512
                  00   DD  00A2                          PUSHL   #0
                  00   DD  00A4                          PUSHL   #0
                  00   DD  00A6                          PUSHL   #0
                  00   DD  00A8                          PUSHL   #0
                  00   DD  00AA                          PUSHL   #0
                  00   DD  00AC                          PUSHL   #0
        00000200 8F   DD  00AE                          PUSHL   #512
        000000AA'EF   9F  00B4                          PUSHAB  REPORT_BUFFER
        000002B6'FF   9F  00BA                          PUSHAB  @REG_STRING
                  54   DD  00C0                          PUSHL   R4
                  1F   DD  00C2                          PUSHL   #31
   000100B0 9F    0B   FB  00C4                          CALLS   #11, @#DS$CVTREG
                           00CB  1030          $DS_PRINTX_S    FMT_BODY,-           ; PRINT REGISTER MNEMONICS
                           00CB  1031                          #EXP_MSG,-
                           00CB  1032                          #REPORT_BUFFER
        000000AA'8F   DD  00CB                          PUSHL   #REPORT_BUFFER
        00001A68'8F   DD  00D1                          PUSHL   #EXP_MSG
        000014C5'EF   9F  00D7                          PUSHAB  FMT_BODY
   000100E8 9F    03   FB  00DD                          CALLS   #$$N, @#DS$PRINTX
                  53   D5  00E4  1033 60$:     TSTL    R3                           ; TEST RECEIVED VALUE AND
                  1C   12  00E6  1034          BNEQ    70$                          ; BRANCH IF NON ZERO
                           00E8  1035          $DS_PRINTX_S    FMT_BODY,-           ; DISPLAY ZERO RECEIVED
                           00E8  1036                          #REC_MSG,-
                           00E8  1037                          #FMT_NOTHING
        000014D0'8F   DD  00E8                          PUSHL   #FMT_NOTHING
        00001A5C'8F   DD  00EE                          PUSHL   #REC_MSG
        000014C5'EF   9F  00F4                          PUSHAB  FMT_BODY
   000100E8 9F    03   FB  00FA                          CALLS   #$$N, @#DS$PRINTX
                0042   31  0101  1038          BRW     80$                          ; BRANCH TO TEST XOR
                           0104  1039 70$:     $DS_CVTREG_S    #31,R3,-             ; CONVERT REGISTER BITS TO MNEMONICS
                           0104  1040                          @REG_STRING,-
```

```
                                 0104  1041                        REPORT_BUFFER,-
                                 0104  1042                        #512
                          00     DD  0104                  PUSHL   #0
                          00     DD  0106                  PUSHL   #0
                          00     DD  0108                  PUSHL   #0
                          00     DD  010A                  PUSHL   #0
                          00     DD  010C                  PUSHL   #0
                          00     DD  010E                  PUSHL   #0
                 00000200 8F     DD  0110                  PUSHL   #512
                 000000AA'EF     9F  0116                  PUSHAB  REPORT_BUFFER
                 000002B6'FF     9F  011C                  PUSHAB  @REG_STRING
                          53     DD  0122                  PUSHL   R3
                          1F     DD  0124                  PUSHL   #31
             000100B0 9F  0B     FB  0126                  CALLS   #11, a#DS$CVTREG
                                 012D  1043        $DS_PRINTX_S    FMT_BODY,-      ; PRINT REGISTER MNEMONICS
                                 012D  1044                        #REC_MSG,-
                                 012D  1045                        #REPORT_BUFFER
                 000000AA'8F     DD  012D                  PUSHL   #REPORT_BUFFER
                 00001A5C'8F     DD  0133                  PUSHL   #REC_MSG
                 000014C5'EF     9F  0139                  PUSHAB  FMT_BODY
             000100E8 9F  03     FB  013F                  CALLS   #$$N, a#DS$PRINTX
                          57     D5  0146  1046 80$:   TSTL    R7                      ; TEST XOR OF REC AND EXP VALUE
                          1C     12  0148  1047        BNEQ    90$                     ; BRANCH IF XOR IS NON ZERO
                                 014A  1048        $DS_PRINTX_S    FMT_BODY,-      ; DISPLAY XOR ZERO
                                 014A  1049                        #DIF_MSG,-
                                 014A  1050                        #FMT_NOTHING
                 000014D0'8F     DD  014A                  PUSHL   #FMT_NOTHING
                 00001A74'8F     DD  0150                  PUSHL   #DIF_MSG
                 000014C5'EF     9F  0156                  PUSHAB  FMT_BODY
             000100E8 9F  03     FB  015C                  CALLS   #$$N, a#DS$PRINTX
                        0042     31  0163  1051        BRW     100$                    ; SKIP CONVERSION
                                 0166  1052 90$:     $DS_CVTREG_S    #31,R7,-        ; CONVERT REGISTER BITS TO MNEMONICS
                                 0166  1053                        @REG_STRING,-
                                 0166  1054                        REPORT_BUFFER,-
                                 0166  1055                        #512
                          00     DD  0166                  PUSHL   #0
                          00     DD  0168                  PUSHL   #0
                          00     DD  016A                  PUSHL   #0
                          00     DD  016C                  PUSHL   #0
                          00     DD  016E                  PUSHL   #0
                          00     DD  0170                  PUSHL   #0
                 00000200 8F     DD  0172                  PUSHL   #512
                 000000AA'EF     9F  0178                  PUSHAB  REPORT_BUFFER
                 000002B6'FF     9F  017E                  PUSHAB  @REG_STRING
                          57     DD  0184                  PUSHL   R7
                          1F     DD  0186                  PUSHL   #31
             000100B0 9F  0B     FB  0188                  CALLS   #11, a#DS$CVTREG
                                 018F  1056        $DS_PRINTX_S    FMT_BODY,-      ; PRINT REGISTER MNEMONICS
                                 018F  1057                        #DIF_MSG,-
                                 018F  1058                        #REPORT_BUFFER
                 000000AA'8F     DD  018F                  PUSHL   #REPORT_BUFFER
                 00001A74'8F     DD  0195                  PUSHL   #DIF_MSG
                 000014C5'EF     9F  019B                  PUSHAB  FMT_BODY
             000100E8 9F  03     FB  01A1                  CALLS   #$$N, a#DS$PRINTX
                                 01A8  1059 100$:    $DS_PRINTX_S    FMT_COMP_ERR,-  ; PRINT EXPECTED, RECEIVED AND
                                 01A8  1060                        #EXP_MSG,R4,-   ; XOR OF EXPECTED AND RECEIVED
                                 01A8  1061                        #REC_MSG,R3,-   ; IN LONGWORDS
```

```
                                01A8  1062                          #DIF_MSG,R7
                        57   DD  01A8                      PUSHL   R7
              00001A74'8F    DD  01AA                      PUSHL   #DIF_MSG
                        53   DD  01B0                      PUSHL   R3
              00001A5C'8F    DD  01B2                      PUSHL   #REC_MSG
                        54   DD  01B8                      PUSHL   R4
              00001A68'8F    DD  01BA                      PUSHL   #EXP_MSG
              000014D9'EF    9F  01C0                      PUSHAB  FMT_COMP_ERR
        000100E8 9F    07   FB  01C6                      CALLS   #$$N, @#DS$PRINTX
                                01CD  1063 PRINT_SBEX:
                           04   01CD  1064         RET                                ; EXIT
                                01CE  1065
                                01CE  1066         .SBTTL  PRINT_MAPERR
                           00000000  1067         .PSECT  PRINT_MAPERR, WORD
                                0000  1068 PRINT_MAPERR::
                          0000  0000  1069                 .WORD    M<>
                                0002  1070 ;++
                                0002  1071 ; FUNCTIONAL DESCRIPTION:
                                0002  1072 ;
                                0002  1073 ;       THIS PROGRAM IS INVOKED AS A RESULT OF A MAPPING FUNCTION ERROR
                                0002  1074 ;       IT IS CALLED BY CEP WHEN THE DIAGNOSTIC DETECTS A MAPPING
                                0002  1075 ;       ERROR IN THE MUT
                                0002  1076 ;
                                0002  1077 ;       CALLING SEQUENCE:
                                0002  1078 ;
                                0002  1079 ;       R3 = MAP REGISTER ACTUALLY SELECTED
                                0002  1080 ;       R4 = MAP REGISTER EXPECTED TO BE SELECTED
                                0002  1081 ;       R6 = VALUE THAT SHOULD HAVE BEEN  IN THE MAP REGISTER (PFN)
                                0002  1082 ;       R7 = PAGE FRAME NUMBER PRODUCED BY THE MAPPING HARDWARE
                                0002  1083 ;       R8 = PHYSICAL ADDRESS EXPECTED
                                0002  1084 ;       R9 = ACTUAL PHYSICAL ADDRESS CONSTRUCTED
                                0002  1085 ;
                                0002  1086 ;       CALLS   #0,PRINT_MAPERR
                                0002  1087 ;--
                                0002  1088         $DS_PRINTB_S    FMT_MAPFUN_ERR  ; PRINT MAP ERROR HEADER
              000015D7'EF    9F  0002                      PUSHAB  FMT_MAPFUN_ERR
        000100E0 9F    01   FB  0008                      CALLS   #$$N, @#DS$PRINTB
                                000F  1089         $DS_PRINTX_S    FMT_MAP_EXERR,- ; PRINT EXPECTED MAP, SELECTED MAP
                                000F  1090                         R4,R3,R8,R9     ; EXPECTED AND ACTUAL PHYSICAL ADDRESS
                        59   DD  000F                      PUSHL   R9
                        58   DD  0011                      PUSHL   R8
                        53   DD  0013                      PUSHL   R3
                        54   DD  0015                      PUSHL   R4
              000015EE'EF    9F  0017                      PUSHAB  FMT_MAP_EXERR
        000100E8 9F    05   FB  001D                      CALLS   #$$N, @#DS$PRINTX
                                0024  1091 PRINT_MAPERRX:
                           04   0024  1092         RET                                ; EXIT
                                0025  1093
                                0025  1094         .SBTTL  PRINT_DTERR
                           00000000  1095         .PSECT  PRINT_DTERR, WORD
                                0000  1096 PRINT_DTERR::
                          02C0  0000  1097                 .WORD    M<R6,R7,R9>
                                0002  1098 ;++
                                0002  1099 ; THIS ROUTINE IS FOR PRINTING DATA COMPARE ERRORS.
                                0002  1100 ;
                                0002  1101 ; CALLING SEQUENCE:
                                0002  1102 ;
```

```
                        0002  1103 ;       R4 = EXPECTED DATA
                        0002  1104 ;       R3 = RECEIVED DATA
                        0002  1105 ;       R6 = EXPECTED BUFFER PHYSICAL ADDRESS
                        0002  1106 ;       R7 = RECEIVED BUFFER PHYSICAL ADDRESS
                        0002  1107 ;       R9 = BYTE COUNT
                        0002  1108 ;--
                56  59  C0  0002  1109         ADDL    R9,R6                   ; CALCULATE FAILING PHYS ADDRESS
                57  59  C0  0005  1110         ADDL    R9,R7                   ; CALCULATE FAILING PHYS ADDRESS
                        0008  1111         $DS_PRINTB_S    FMT_DCMPERR,-       ; PRINT ERROR
                        0008  1112                         R4,R3,R6,R7,R9
                59  DD  0008                       PUSHL   R9
                57  DD  000A                       PUSHL   R7
                56  DD  000C                       PUSHL   R6
                53  DD  000E                       PUSHL   R3
                54  DD  0010                       PUSHL   R4
      00001709'EF   9F  0012                       PUSHAB  FMT_DCMPERR
 000100E0 9F    06  FB  0018                       CALLS   #$$N, a#DS$PRINTB
                        001F  1113 PRINT_DTERRX:
                04  001F  1114         RET                                     ; EXIT
                        0020  1115
                        0020  1116         .SBTTL  PRINT_INTERR
                00000000  1117         .PSECT  PRINT_INTERR, WORD
                        0000  1118 PRINT_INTERR::
              0010  0000  1119                   .WORD   M<R4>
                        0002  1120 ;++
                        0002  1121 ; THIS ROUTINE IS USED FOR PRINTING INTERRUPT FAILURE ERRORS
                        0002  1122 ;
                        0002  1123 ; IMPLICIT INPUTS:
                        0002  1124 ;
                        0002  1125 ;       R4 = STATUS REGISTER BITS CAUSING THE INTERRUPT
                        0002  1126 ;--
                        0002  1127
                        0002  1128         $DS_CVTREG_S    #31,R4,-
                        0002  1129                         FMT_STATUS_REG,-
                        0002  1130                         REPORT_BUFFER,#512
                00  DD  0002                       PUSHL   #0
                00  DD  0004                       PUSHL   #0
                00  DD  0006                       PUSHL   #0
                00  DD  0008                       PUSHL   #0
                00  DD  000A                       PUSHL   #0
                00  DD  000C                       PUSHL   #0
      00000200  8F  DD  000E                       PUSHL   #512
      000000AA'EF   9F  0014                       PUSHAB  REPORT_BUFFER
      00000C42'EF   9F  001A                       PUSHAB  FMT_STATUS_REG
                54  DD  0020                       PUSHL   R4
                1F  DD  0022                       PUSHL   #31
 000100B0 9F    0B  FB  0024                       CALLS   #11, a#DS$CVTREG
                        002B  1131         $DS_PRINTB_S    FMT_INTFAILED,-
                        002B  1132                         #REPORT_BUFFER
      000000AA'8F   DD  002B                       PUSHL   #REPORT_BUFFER
      00001B7F'EF   9F  0031                       PUSHAB  FMT_INTFAILED
 000100E0 9F    02  FB  0037                       CALLS   #$$N, a#DS$PRINTB
                        003E  1133         .SBTTL  PRINT_INTERRX
                00000000  1134         .PSECT  PRINT_INTERRX, WORD
                        0000  1135 PRINT_INTERRX::
                04  0000  1136         RET
                        0001  1137
```

```
                     0001  1138          .SBTTL  PRINT_NXADR
                  00000000  1139          .PSECT  PRINT_NXADR, WORD
                     0000  1140 PRINT_NXADR::
                0000 0000  1141                  .WORD    M<>
                     0002  1142
                     0002  1143 ;++
                     0002  1144 ; THIS ROUTINE IS FOR PRINTING NON-EXISTENT REGISTER ADDRESSES
                     0002  1145 ;--
                     0002  1146
                     0002  1147          $DS_PRINTX_S    FMT_NX_ER,NXM_ADR ; PRINT NON-EXISTENT REGISTER
          000002D6'EF   DD  0002                  PUSHL   NXM_ADR
          00001AB6'EF   9F  0008                  PUSHAB  FMT_NX_ER
     000100E8 9F    02   FB  000E                  CALLS   #$$N, @#DS$PRINTX
                     0015  1148 PRINT_NXADRX:
                     04  0015  1149          RET                             ; EXIT
```

```
                        0016  1150          .SBTTL  PRINT_CMDER
                    00000000  1151          .PSECT  PRINT_CMDER, WORD
                        0000  1152 PRINT_CMDER::
                   0000 0000  1153                  .WORD    M<>                ; ENTRY MASK
                        0002  1154 ;++
                        0002  1155 ; THIS ROUTINE IS FOR PRINTING CMI COMMAND ERROR MESSAGES
                        0002  1156 ;--
                        0002  1157
          52    14 AC   D0    0002  1158          MOVL     ERR$_P1(AP),R2           ; P1 CONTAINS EXP CMD
          53    18 AC   D0    0006  1159          MOVL     ERR$_P2(AP),R3           ; P2 CONTAINS RCVD CMD
       54    53   52     CD    000A  1160          XORL3    R2,R3,R4                 ; R4 CONTAINS BAD BIT POS.
                        000E  1161          $DS_PRINTX_S     FMT_CMICMD_ERR  ; PRINT ERROR MESSAGE
        000017A4'EF     9F    000E                        PUSHAB   FMT_CMICMD_ERR
 000100E8 9F     01     FB    0014                        CALLS    #$$N, a#DS$PRINTX
                        001B  1162          $DS_PRINTX_S     FMT_VAL,-
                        001B  1163                       R2,R3,R4
                    54  DD    001B                        PUSHL    R4
                    53  DD    001D                        PUSHL    R3
                    52  DD    001F                        PUSHL    R2
        0000190F'EF     9F    0021                        PUSHAB   FMT_VAL
 000100E8 9F     04     FB    0027                        CALLS    #$$N, a#DS$PRINTX
                        002E  1164 PRINT_CMDERX:                                ; EXIT
                    04  002E  1165          RET
                        002F  1166
                        002F  1167
                        002F  1168          .SBTTL  PRINT_IBCERR
                    00000000  1169          .PSECT  PRINT_IBCERR, WORD
                        0000  1170 PRINT_IBCERR::
                   0000 0000  1171                  .WORD    M<>                ; ENTRY MASK
                        0002  1172 ;++
                        0002  1173 ; THIS ROUTINE IS FOR PRINTING ERROR MESSAGES FOR TEST 19,
                        0002  1174 ; IGNORE BYTE COUNTER MODE TEST.
                        0002  1175 ;--
                        0002  1176
09   01   000002DA'EF    CF    0002  1177          CASEL    IBC_MSG,#1,#9            ; READ LOC IBC_MSG TO
                    0012' 000A  1178 10$:     .WORD    100$-10$                 ; DETERMINE MESSAGE PRINTOUT
                    0041' 000C  1179          .WORD    200$-10$                 ;
                    0070' 000E  1180          .WORD    300$-10$                 ;
                    0080' 0010  1181          .WORD    400$-10$                 ;
                    00AF' 0012  1182          .WORD    500$-10$                 ;
                    00BF' 0014  1183          .WORD    600$-10$                 ;
                    00CF' 0016  1184          .WORD    700$-10$                 ;
                    00DF' 0018  1185          .WORD    800$-10$                 ;
                    00EF' 001A  1186          .WORD    900$-10$                 ;
                        001C  1187
          55    14 AC   D0    001C  1188 100$:    MOVL     ERR$_P1(AP),R5           ; P1=EXP BCR CONTENTS
          54    18 AC   D0    0020  1189          MOVL     ERR$_P2(AP),R4           ; P2=RCV BCR CONTENTS
       56   54   55     CD    0024  1190          XORL3    R5,R4,R6                 ; R6 CONTAINS BAD BIT POS.
                        0028  1191          $DS_PRINTX_S     FMT_IBCER1      ; PRINT ERROR MESSAGE
        000017D0'EF     9F    0028                        PUSHAB   FMT_IBCER1
 000100E8 9F     01     FB    002E                        CALLS    #$$N, a#DS$PRINTX
                        0035  1192          $DS_PRINTX_S     FMT_VAL,-       ; AND EXP, RCVD, AND
                        0035  1193                       R5,R4,R6            ; XOR RESULTS
                    56  DD    0035                        PUSHL    R6
                    54  DD    0037                        PUSHL    R4
                    55  DD    0039                        PUSHL    R5
        0000190F'EF     9F    003B                        PUSHAB   FMT_VAL
```

```
000100E8 9F   04   FB  0041                      CALLS   #$$N, a#DS$PRINTX
            00DA   31  0048  1194          BRW     PRINT_IBCERRX           ; RETURN
      58   14 AC   D0  004B  1195 200$:    MOVL    ERR$_P1(AP),R8          ; P1=EXP VAR CONTENTS
      55   18 AC   D0  004F  1196          MOVL    ERR$_P2(AP),R5          ; P2=RCV VAR CONTENTS
   56   55   58   CD  0053  1197          XORL3   R8,R5,R6                ; R6 CONTAINS BAD BIT POS.
                    0057  1198          $DS_PRINTX_S    FMT_IBCER2      ; PRINT ERROR MESSAGE
    000017E0'EF   9F  0057                      PUSHAB  FMT_IBCER2
000100E8 9F   01   FB  005D                      CALLS   #$$N, a#DS$PRINTX
                    0064  1199          $DS_PRINTX_S    FMT_VAL,-       ; AND EXP, RCVD, AND
                    0064  1200                  R8,R5,R6                ; XOR RESULTS
               56   DD  0064                      PUSHL   R6
               55   DD  0066                      PUSHL   R5
               58   DD  0068                      PUSHL   R8
    0000190F'EF   9F  006A                      PUSHAB  FMT_VAL
000100E8 9F   04   FB  0070                      CALLS   #$$N, a#DS$PRINTX
            00AB   31  0077  1201          BRW     PRINT_IBCERRX           ; RETURN
                    007A  1202 300$:    $DS_PRINTX_S    FMT_IBCER3      ; PRINT ERROR MESSAGE
    000017F0'EF   9F  007A                      PUSHAB  FMT_IBCER3
000100E8 9F   01   FB  0080                      CALLS   #$$N, a#DS$PRINTX
            009B   31  0087  1203          BRW     PRINT_IBCERRX           ; RETURN
      58   14 AC   D0  008A  1204 400$:    MOVL    ERR$_P1(AP),R8          ; P1=EXP BCR CONTENTS
      55   18 AC   D0  008E  1205          MOVL    ERR$_P2(AP),R5          ; R5=RCV BCR CONTENTS
   56   55   58   CD  0092  1206          XORL3   R8,R5,R6                ; R6 CONTAINS BAD BIT POS.
                    0096  1207          $DS_PRINTX_S    FMT_IBCER4      ; PRINT ERROR MESSAGE
    00001809'EF   9F  0096                      PUSHAB  FMT_IBCER4
000100E8 9F   01   FB  009C                      CALLS   #$$N, a#DS$PRINTX
                    00A3  1208          $DS_PRINTX_S    FMT_VAL,-       ; AND EXP,RCVD, AND
                    00A3  1209                  R8,R5,R6                ; XOR RESULTS
               56   DD  00A3                      PUSHL   R6
               55   DD  00A5                      PUSHL   R5
               58   DD  00A7                      PUSHL   R8
    0000190F'EF   9F  00A9                      PUSHAB  FMT_VAL
000100E8 9F   04   FB  00AF                      CALLS   #$$N, a#DS$PRINTX
            006C   31  00B6  1210          BRW     PRINT_IBCERRX           ; RETURN
                    00B9  1211 500$:    $DS_PRINTX_S    FMT_IBCER5      ; PRINT ERROR MESSAGE
    00001819'EF   9F  00B9                      PUSHAB  FMT_IBCER5
000100E8 9F   01   FB  00BF                      CALLS   #$$N, a#DS$PRINTX
            005C   31  00C6  1212          BRW     PRINT_IBCERRX           ; RETURN
                    00C9  1213 600$:    $DS_PRINTX_S    FMT_IBCER6      ; PRINT ERROR MESSAGE
    0000184A'EF   9F  00C9                      PUSHAB  FMT_IBCER6
000100E8 9F   01   FB  00CF                      CALLS   #$$N, a#DS$PRINTX
            004C   31  00D6  1214          BRW     PRINT_IBCERRX           ; RETURN
                    00D9  1215 700$:    $DS_PRINTX_S    FMT_IBCER7      ; PRINT ERROR MESSAGE
    00001869'EF   9F  00D9                      PUSHAB  FMT_IBCER7
000100E8 9F   01   FB  00DF                      CALLS   #$$N, a#DS$PRINTX
            003C   31  00E6  1216          BRW     PRINT_IBCERRX           ; RETURN
                    00E9  1217 800$:    $DS_PRINTX_S    FMT_IBCER8      ; PRINT ERROR MESSAGE
    000018C3'EF   9F  00E9                      PUSHAB  FMT_IBCER8
C00100E8 9F   01   FB  00EF                      CALLS   #$$N, a#DS$PRINTX
            002C   31  00F6  1218          BRW     PRINT_IBCERRX           ; RETURN
      5B   14 AC   D0  00F9  1219 900$:    MOVL    ERR$_P1(AP),R11         ; P1=EXP CONTENTS OF MAP REG
      53   18 AC   D0  00FD  1220          MOVL    ERR$_P2(AP),R3          ; P2=RCVD CONTENTS OF MAP REG
   56   53   5B   CD  0101  1221          XORL3   R11,R3,R6               ; R6= BAD BIT POSITIONS
                    0105  1222          $DS_PRINTX_S    FMT_IBCER9      ; PRINT ERROR MESSAGE
    000018E2'EF   9F  0105                      PUSHAB  FMT_IBCER9
000100E8 9F   01   FB  010B                      CALLS   #$$N, a#DS$PRINTX
                    0112  1223          $DS_PRINTX_S    FMT_VAL,-       ; AND EXP, RCVD AND
```

```
                          0112  1224                          R11,R3,R6        ; XOR DATA
                 56   DD  0112                      PUSHL   R6
                 53   DD  0114                      PUSHL   R3
                 5B   DD  0116                      PUSHL   R11
      0000190F'EF     9F  0118                      PUSHAB  FMT_VAL
 000100E8 9F     04   FB  011E                      CALLS   #$$N, @#DS$PRINTX
                          0125  1225 PRINT_IBCERRX:                          ; EXIT
                     04   0125  1226          RET
                          0126  1227
                          0126  1228          .SBTTL  PRINT_VECTERR
                     00000000  1229          .PSECT  PRINT_VECTERR, WORD
                          0000  1230 PRINT_VECTERR::
                   0000   0000  1231                  .WORD    M<>             ; ENTRY MASK
                          0002  1232 ;++
                          0002  1233 ; THIS ROUTINE IS FOR PRINTING THE EXP AND RCVD VECTOR ADDRESSES
                          0002  1234 ; IN T17_S1
                          0002  1235 ;--
                          0002  1236
        57    14 AC   D0  0002  1237          MOVL    ERR$_P1(AP),R7           ; P1 CONTAINS THE RCVD VECTOR
        58    18 AC   D0  0006  1238          MOVL    ERR$_P2(AP),R8           ; P2 CONTAINS THE EXP VECTOR
     59    58    57   CD  000A  1239          XORL3   R7,R8,R9                 ; R9 CONTAINS XOR DATA
                          000E  1240          $DS_PRINTX_S    FMT_VECERR       ; PRINT ERROR
      00001AE3'EF     9F  000E                      PUSHAB  FMT_VECERR
 000100E8 9F     01   FB  0014                      CALLS   #$$N, @#DS$PRINTX
                          001B  1241          $DS_PRINTX_S    FMT_VAL,-        ; PRINT EXP, RCVD, AND XOR DATA
                          001B  1242                          R8,R7,R9
                 59   DD  001B                      PUSHL   R9
                 57   DD  001D                      PUSHL   R7
                 58   DD  001F                      PUSHL   R8
      0000190F'EF     9F  0021                      PUSHAB  FMT_VAL
 000100E8 9F     04   FB  0027                      CALLS   #$$N, @#DS$PRINTX
                          002E  1243 PRINT_VECTERRX:                         ; EXIT
                     04   002E  1244                  RET
                          002F  1245
                          002F  1246          .SBTTL  PRINT_SILOER
                     00000000  1247          .PSECT  PRINT_SILOER, WORD
                          0000  1248 PRINT_SILOER::
                   0000   0000  1249                  .WORD    M<>             ; ENTRY MASK
                          0002  1250 ;++
                          0002  1251 ; THIS ROUTINE IS FOR PRINTING AN ERROR MESSAGE AND EXP AND
                          0002  1252 ; RECEIVED DATA FOR TEST 16_S2, SILO TEST.
                          0002  1253 ;--
                          0002  1254
        58    14 AC   D0  0002  1255          MOVL    ERR$_P1(AP),R8           ; P1 CONTAINS RCVD DATA
        56    18 AC   D0  0006  1256          MOVL    ERR$_P2(AP),R6           ; P2 CONTAINS EXP DATA
     57    56    58   CD  000A  1257          XORL3   R8,R6,R7                 ; R7 CONTAINS XOR DATA
                          000E  1258          $DS_PRINTX_S    FMT_SERR         ; PRINT MESSAGE
      00001B0C'EF     9F  000E                      PUSHAB  FMT_SERR
 000100E8 9F     01   FB  0014                      CALLS   #$$N, @#DS$PRINTX
                          001B  1259          $DS_PRINTX_S    FMT_VAL,-        ; PRINT DATA RESULTS
                          001B  1260                          R6,R8,R7
                 57   DD  001B                      PUSHL   R7
                 58   DD  001D                      PUSHL   R8
                 56   DD  001F                      PUSHL   R6
      0000190F'EF     9F  0021                      PUSHAB  FMT_VAL
 000100E8 9F     04   FB  0027                      CALLS   #$$N, @#DS$PRINTX
                          002E  1261 PRINT_SILOERX:                          ; EXIT
```

```
                04  002E  1262                RET
```

```
                                002F  1263          .SBTTL  INITIALIZATION
                                002F  1264
                                002F  1265 $DS_BGNINIT
                                002F                      .SAVE
                             00000000                     .PSECT  INITIALIZE,LONG
                                0000                 INITIALIZE:
                           0000 0000                     .WORD   M<>             ; ENTRY MASK
                                0002  1266 ;++
                                0002  1267 ; THIS ROUTINE IS CALLED AT THE BEGINNING OF EACH TEST SEQUENCE
                                0002  1268 ; FROM THE DIAGNOSTIC SUPERVISOR TEST SEQUENCER.
                                0002  1269 ;--
      52   00000014'EF    DE   0002  1270          MOVAL   MBA_LN,R2       ; GET ADDRESS OF LUN TABLE
                                0009  1271          $DS_BPASS0      4$      ; BRANCH IF PASS 0
                      1D    E0   0009                  BBS     #DSA$V_PASS0,-  ; BR IF PASS 0
        03 0000FE00 9F          000B                          a#DSA$GL_FLAGS, 4$
                    0088    31   0011  1272          BRW     8$              ; NOT PASS 0
           00000008'EF      D4   0014  1273 4$:      CLRL    LN              ; INIT THE LUN COUNTER
           00000020'EF      94   001A  1274          CLRB    NO_RH750        ; INIT THE NUMBER OF UNITS COUNTER
           00000021'EF      94   0020  1275          CLRB    NO_MBE          ; INIT THE NUMBER OF MBE's COUNTER
               62    00     D2   0026  1276          MCOML   #0,(R2)         ; INIT THE TABLE
                                0029  1277 1$:      $DS_GPHARD_S    LN,PTBASE ; IS VALUE OF LN PRESENT?
           00000022'EF      DF   0029                  PUSHAL  PTBASE
           00000008'EF      DD   002F                  PUSHL   LN
      00010018 9F    02    FB   0035                  CALLS   #2, a#DS$GPHARD
                                003C  1278          $DS_BERROR      5$      ; BRANCH IF NO
                 36 50      E9   003C                  BLBC    R0,5$
      51   00000022'EF      D0   003F  1279          MOVL    PTBASE,R1       ; GET ADDRESS OF P TABLE
                   26 A1    D1   0046  1280          CMPL    HP$T_TYPE(R1),-
           0000197F'EF              0049  1281                  ASCII_MBE       ; IS THIS AN MBE P TABLE?
                     08     12   004E  1282          BNEQ    2$              ; BRANCH IF NO
           00000021'EF      96   0050  1283          INCB    NO_MBE          ; COUNT THIS MBE
                     15     11   0056  1284          BRB     3$              ; CONTINUE
                                0058  1285
        51   00000020'EF    9A   0058  1286 2$:      MOVZBL  NO_RH750,R1     ; GET CURRENT NUMBER OF UNITS
      6241   00000008'EF    90   005F  1287          MOVB    LN,(R2) R1      ; SAVE LUN OF THIS MBA IN TABLE
           00000020'EF      96   0067  1288          INCB    NO_RH750        ; COUNT THE NUMBER OF SELECTED UNITS
                                006D  1289
           00000008'EF      D6   006D  1290 3$:      INCL    LN              ; ...
                     B4     11   0073  1291          BRB     1$              ; CONTINUE
                                0075  1292 ;
                                0075  1293 ; Scanned all p tables and built a table of RH750 logical unit numbers.
                                0075  1294 ;
           00000020'EF      95   0075  1295 5$:      TSTB    NO_RH750        ; ANY RH750'S SELECTED?
                     14     12   007B  1296          BNEQ    7$              ; BRANCH IF YES
                                007D  1297          $DS_PRINTF_S    FMT_NO_RH750
           00001B31'EF      9F   007D                  PUSHAB  FMT_NO_RH750
        000100F0 9F    01    FB   0083                  CALLS   #$$N, a#DS$PRINTF
                                008A  1298          $DS_ABORT       ARG=PROGRAM
        00010020 9F    6C    FA   008A                  CALLG   (AP), a#DS$ABORT
  0000000C'EF   00000008'EF    D0   0091  1299 7$:      MOVL    LN,DEVICES      ; SAVE TOTAL NUMBER OF DEVICES SELECTED
                                009C  1300 ;
                                009C  1301 ; Enter here if not pass zero.
                                009C  1302 ;
           0000001C'EF      D6   009C  1303 8$:      INCL    LUN             ; SELECT THE NEXT UNIT FOR TEST
                                00A2  1304          $DS_BNPASS0     10$     ; BRANCH IF NOT PASS 0
                     1D     E1   00A2                  BBC     #DSA$V_PASS0,-  ; BR IF NOT PASS 0
          06 0000FE00 9F          00A4                          a#DSA$GL_FLAGS, 10$
```

```
                0000001C'EF   D4  00AA  1305          CLRL    LUN             ; INIT THE LUN UNDER TEST
                              00B0  1306
    00000020'EF 0000001C'EF   91  00B0  1307 10$:     CMPB    LUN,NO_RH750    ; TESTED ALL SELECTED UNITS?
                          0D  19  00BB  1308          BLSS    15$             ; BRANCH IF NO
                0000001C'EF   D4  00BD  1309          CLRL    LUN             ; INIT THE LUN
                              00C3  1310          $DS_ENDPASS_S               ; REPORT END OF PASS
            00010010 9F   00  FB  00C3                      CALLS   #0, a#DS$ENDPASS
                              00CA  1311
            51  0000001C'EF   D0  00CA  1312 15$:     MOVL    LUN,R1          ; GET THE LUN UNDER TEST
                  53   6241   9A  00D1  1313          MOVZBL  (R2) R1 ,R3     ; GET LUN FROM TABLE
                              00D5  1314          $DS_GPHARD_S    R3,PTBASE ; GET P TABLE BASE ADDRESS
                00000022'EF   DF  00D5                      PUSHAL  PTBASE
                          53  DD  00DB                      PUSHL   R3
            00010018 9F   02  FB  00DD                      CALLS   #2, a#DS$GPHARD
            53  00000022'EF   D0  00E4  1315          MOVL    PTBASE,R3       ; ...
                        18 A3   D0  00EB  1316          MOVL    HP$A_DEVICE(R3),-
                00000000'EF       00EE  1317                  RH_CUR_ADR ; GET CSR ADDRESS OF RH UNDER TEST
        00000018'EF   24 A3   B0  00F3  1318          MOVW    HP$W_VECTOR(R3),RH_VECTOR ; AND THE INTERRUPT VECTOR
          00000010'EF   53    D0  00FB  1319          MOVL    R3,RH750_LINK   ; SAVE P TABLE ADDRESS FOR MBE CHECK
                              0102  1320 ;
                              0102  1321 ; IF TRACE FLAG IS SET OR IF PASS 0/1 THEN PRINT RHn ADDR =
                              0102  1322 ;
            0000FE54'EF   01  D1  0102  1323          CMPL    #1,DSA$GL_PASSNO; PASS 0 OR 1?
                          0D  18  0109  1324          BGEQ    17$             ; BRANCH IF YES
    0000FE00'EF   00000400 8F   D3  010B  1325          BITL    #DSA$M_TRACE,DSA$GL_FLAGS
                              0116  1326                                      ; TRACE FLAG SET?
                          2F  13  0116  1327          BEQL    30$             ; BRANCH IF NO
                          63  7D  0118  1328 17$:     MOVQ    HP$Q_DEVICE(R3),-
                000002E2'EF       011A  1329                  DEVICE_PTR      ; GET DEVICE NAME DESCRIPTOR
                000002E2'EF   D7  011F  1330          DECL    DEVICE_PTR      ; DELEATE THE UNDERSCORE FROM
                000002E6'EF   D6  0125  1331          INCL    DEVICE_PTR+4    ; THE DEVICE NAME
            50  000002E2'EF   7E  012B  1332          MOVAQ   DEVICE_PTR,R0
                              0132  1333          $DS_PRINTF_S    FMT_INIT1,- ; PRINT RH ADDRESS AND DEVICE NAME
                              0132  1334                          R0,-
                              0132  1335                          RH_CUR_ADR
                00000000'EF   DD  0132                      PUSHL   RH_CUR_ADR
                          50  DD  0138                      PUSHL   R0
                00001216'EF   9F  013A                      PUSHAB  FMT_INIT1
            000100F0 9F   03  FB  0140                      CALLS   #$$N, a#DS$PRINTF
                              0147  1336
                          50  D4  0147  1337 30$:     CLRL    R0              ; INIT THE DRIVE ADDRESS TABLE
            55  00000000'EF   D0  0149  1338          MOVL    RH_CUR_ADR,R5   ; ...
            55  00000400 8F   C0  0150  1339          ADDL    #EXT_REG_OFFSET,R5 ; ...
        00000026'EF40     55  D0  0157  1340 35$:     MOVL    R5,DRIVE_ADR_TBL R0  ; ...
            55  00000080 8F   C0  015F  1341          ADDL    #DRIVE_OFFSET,R5; ...
             ED 50        07  F3  0166  1342          AOBLEQ  #7,R0,35$
                              016A  1343
    00000046'EF 00000026'EF   D0  016A  1344          MOVL    DRIVE0,DRIVE_INUSE ; INIT ADDRESS OF DRIVE IN USE
                              0175  1345          $DS_SETVEC_S    RH_VECTOR,RH_ISR ; CAPTURE THE INTERRUPT VECTOR
                          00  DD  0175                      PUSHL   #0
                00000000'EF   DF  0177                      PUSHAL  RH_ISR
                00000018'EF   DD  017D                      PUSHL   RH_VECTOR
            00010160 9F   03  FB  0183                      CALLS   #3, a#DS$SETVEC
                              018A  1346 ;
                              018A  1347 ; IF THERE ARE ANY MBE'S SELECTED, SEE IF THEY ARE ATTACHED TO THIS RH750
                              018A  1348 ;
                00000004'EF   D4  018A  1349          CLRL    MBE_CUR_ADR     ; INIT CURRENT ADDRESS OF MBE
```

```
                00000021'EF   95   0190  1350          TSTB    NO_MBE              ; ANY MBE'S SELECTED?
                          4D   13   0196  1351          BEQL    60$                 ; BRANCH IF NO
             00000008'EF    01   CE   0198  1352          MNEGL   #1,LN               ; INIT LUN COUNTER
                          38   11   019F  1353          BRB     45$                 ; CHECK IF DONE
                               01A1  1354 40$:          $DS_GPHARD_S    LN,PTBASE ; GET A P TABLE POINTER
                00000022'EF   DF   01A1                PUSHAL  PTBASE
                00000008'EF   DD   01A7                PUSHL   LN
             00010018 9F      02   FB   01AD          CALLS   #2, a#DS$GPHARD
             51  00000022'EF  D0   01B4  1355          MOVL    PTBASE,R1           ; GET BASE ADDRESS OF PTABLE
          0000197F'EF   26 A1  D1   01BB  1356          CMPL    HP$T_TYPE(R1),ASCII_MBE
                               01C3  1357                                          ; IS THIS AN MBE P TABLE?
                          14   12   01C3  1358          BNEQ    45$                 ; BRANCH IF NO
          00000010'EF   20 A1  D1   01C5  1359          CMPL    HP$A_LINK(R1),RH750_LINK
                               01CD  1360                                          ; IS IT CONNECTED TO THIS RH750?
                          0A   12   01CD  1361          BNEQ    45$                 ; BRANCH IF NO
                       18 A1   D0   01CF  1362          MOVL    HP$A_DEVICE(R1),-
                00000004'EF        01D2  1363                  MBE_CUR_ADR         ; GET BASE ADDRESS OF SELECTED MBE
                          0C   11   01D7  1364          BRB     60$                 ; EXIT (ONLY ONE MBE PER MASS BUS)
BC 00000008'EF  0000000C'EF   F2   01D9  1365 45$:     AOBLSS  DEVICES,LN,40$      ; CHECK ALL P TABLES
                               01E5  1366
                               01E5  1367 60$:
                               01E5  1368 $DS_ENDINIT
                               01E5       INITIALIZE_X:
             00010058 9F      6E   FA   01E5                  CALLG   (SP), a#DS$BREAK
                               04   01EC                  RET                     ; RETURN TO DIAGNOSTIC SUPERVISOR
                               0000002F                    .RESTORE
```

```
                                 002F  1370 .SBTTL  CODE EXECUTED AT THE END OF EACH TEST SEQUENCE
                                 002F  1371 $DS_BGNCLEAN
                                 002F                          .SAVE
                            00000000                          .PSECT  CLEANUP,LONG
                                 0000       CLEAN_UP:
                           0000  0000                          .WORD    M<>              ; ENTRY MASK
                     52    D4    0002  1372          CLRL     R2                  ; SETUP
      53   00000014'EF    DE    0004  1373          MOVAL    MBA_LN,R3           ; ...
             54   6342    98    000B  1374 1$:      CVTBL    (R3) R2 ,R4         ; GET LUN FROM TABLE
                     31    19    000F  1375          BLSS     20$                 ; BRANCH IF LAST ITEM IN TABLE
                                 0011  1376          $DS_GPHARD_S     R4,PTBASE ; GET P TABLE BASE ADDRESS
           00000022'EF    DF    0011                          PUSHAL   PTBASE
                     54    DD    0017                          PUSHL    R4
      00010018 9F    02    FB    0019                          CALLS    #2, @#DS$GPHARD
      57   00000022'EF    D0    0020  1377          MOVL     PTBASE,R7           ; GET ADDRESS OF P TABLE
             58   18 A7    D0    0027  1378          MOVL     HP$A_DEVICE(R7),R8 ; GET ADDRESS OF RH CSR0
             04 A8    01    D0    002B  1379          MOVL     #PGM_INIT,CR(R8) ; INIT THE ADAPTER
                     59    D4    002F  1380          CLRL     R9                  ; SETUP TO CLEAR ALL THE MAP REGISTERS
             0800 C849     D4    0031  1381 7$:      CLRL      X800(R8) R9        ; CLEAR A MAP
   F3 59   000000FF 8F    F3    0036  1382          AOBLEQ   #255,R9,7$          ; ...
                     52    D6    003E  1383 10$:     INCL     R2                  ; SELECT NEXT DEVICE
                     C9    11    0040  1384          BRB      1$
                                 0042  1385
                                 0042  1386 20$:     $DS_CLRVEC_S     RH_VECTOR ; RELEASE THE CURRENT INTERRUPT VECTOR
           00000018'EF    DD    0042                          PUSHL    RH_VECTOR
      00010168 9F    01    FB    0048                          CALLS    #1, @#DS$CLRVEC
                                 004F  1387 $DS_ENDCLEAN
                                 004F       CLEAN_UP_X:
      00010058 9F    6E    FA    004F                          CALLG    (SP), @#DS$BREAK
                           04    0056                          RET                          ; RETURN TO DIAGNOSTIC SUPERVISOR
                           0000002F                          .RESTORE
```

```
                                002F  1389          .SBTTL  MACHINE CHECK SERVICE ROUTINE
                                002F  1390
                             00000000  1391          .PSECT  MACH_CHK_SRV, LONG
                                0000  1392 MACH_CHK_SRV::
                                0000  1393 ;++
                                0000  1394 ; THIS ROUTINE IS INVOKED AS A RESULT OF AN EXPECTED MACHINE CHECK DUE
                                0000  1395 ; TO A NXM READ.
                                0000  1396 ;
                                0000  1397 ; UNEXPECTED MACHINE CHECKS WILL BE HANDLED BY THE DIAGNOSTIC SUPERVISOR.
                                0000  1398 ;
                                0000  1399 ;--
                                0000  1400
          00000054'EF   94      0000  1401          CLRB    NXM_RD_FLAG             ; CLEAR FLAG
             5E    8E   C0      0006  1402          ADDL2   (SP)+,SP                ; RESET STACK POINTER
       6E    00000055'EF   D0  0009  1403          MOVL    CONT_PC,(SP)            ; MODIFY RETURN PC TO CONTINUE TEST
                           02   0010  1404          REI                             ; EXECUTE RETURN FROM EXCEPTION
```

```
                                0011  1405          .SBTTL  RH750 INTERRUPT SERVICE ROUTINE
                           00000000   1406          .PSECT  RH_ISR, LONG
                                0000  1407 RH_ISR::
                                0000  1408 ;++
                                0000  1409 ; FUNCTIONAL DESCRIPTION:
                                0000  1410 ;
                                0000  1411 ; THIS ROUTINE HANDLES INTERRUPTS CAUSED BY THE TEST CODE.
                                0000  1412 ; IT EXPECTS THAT THE INT_EXPECTED FLAG IS SET AND THAT R4
                                0000  1413 ; CONTAINS THE EXPECTED STATUS CONDITION.
                                0000  1414 ;
                                0000  1415 ; UNEXPECTED INTERRUPTS ARE NOTED ON THE CONSOLE TERMINAL.
                                0000  1416 ;
                                0000  1417 ; CALLING SEQUENCE:
                                0000  1418 ;
                                0000  1419 ; $DS_SETVEC_S RH_VECTOR
                                0000  1420 ; SET INT_EXPECTED FLAG
                                0000  1421 ; CLEAR INTERRUPT OCCURRED
                                0000  1422 ; CAUSE INTERRUPT CONDITION
                                0000  1423 ;
                                0000  1424 ; OUTPUTS:
                                0000  1425 ;
                                0000  1426 ; INT_OCCURRED <-- 1
                                0000  1427 ; EXP_INTERRUPT <-- 0
                                0000  1428 ; REI
                                0000  1429 ;
                                0000  1430 ; THE VALUE OF THE PC IS PRINTED IN THE EVENT OF AN UNEXPECTED INTERRUPT.
                                0000  1431 ;--
                          0F    BB    0000  1432          PUSHR   #^M<R0,R1,R2,R3>        ; SAVE THESE REGISTERS
2A 00000052'EF            00    E4    0002  1433          BBSC    #0,EXP_INTERRUPT,10$    ; IF INTERRUPT IS EXPECTED PROCEED
                                      000A  1434          $DS_ERRSYS_S    #2000           ; FATAL ERROR
                                00    DD    000A                PUSHL   #0
                                00    DD    000C                PUSHL   #0
                                00    DD    000E                PUSHL   #0
                           000007D0 8F    DD    0010                PUSHL   #2000
                           000100C0 9F    04    FB    0016          CALLS   #^$$M, a#DS$ERRSYS
                                                001D  1435          $DS_PRINTX_S    FMT_UNEXP_INT,- ; PRINT PC
                                                001D  1436                          20(SP)
                                   14 AE    DD    001D                PUSHL   20(SP)
                              00001B50'EF   9F    0020                PUSHAB  FMT_UNEXP_INT
                           000100E8 9F    02    FB    0026          CALLS   #^$$N, a#DS$PRINTX
                                                002D  1437          $DS_ABORT                       ; AND ABORT
                           00010020 9F    6C    FA    002D                CALLG   (AP), a#DS$ABORT
                           00000053'EF    01    90    0034  1438 10$:    MOVB    #1,INT_OCCURRED         ; SET INTERRUPT EVENT FLAG
                                                003B  1439 RH_ISRX:
                                                003B  1440          $DS_CANWAIT_G                   ; ABORT ANY DELAYS IN PROGRESS
                           00010070 9F    6E    FA    003B                CALLG   (SP), a#DS$CANWAIT
                           50   00000000'EF    D0    0042  1441          MOVL    RH_CUR_ADR,R0           ; GET RH750 ADDRESS
                                 04 A0    04    CA    0049  1442          BICL    #INTERRUPT_EN,CR(R0)    ; CLEAR INTERRUPT ENABLE
                                          0F    BA    004D  1443          POPR    #^M<R0,R1,R2,R3>        ; POP SAVED REGISTERS
                                          02    004F  1444          REI                             ; EXIT
```

```
                        0050  1445          .SBTTL  MMREAD SUBROUTINE
                    00000000  1446          .PSECT  ISSUE_MMREAD, WORD
                        0000  1447 ISSUE_MMREAD::
                    0030 0000  1448                  .WORD    M<R4,R5>
                        0002  1449 ;++
                        0002  1450 ; FUNCTIONAL DESCRIPTION:
                        0002  1451 ;
                        0002  1452 ;       THIS ROUTINE IS USED TO PLACE THE RH750 UNDER TEST
                        0002  1453 ;       INTO THE MAINTENANCE MODE, ISSUE A READ COMMAND
                        0002  1454 ;       TO A PSEUDO DRIVE. THIS
                        0002  1455 ;       INSURES THAT CTOD IS SET AND RUN IS ASSERTED
                        0002  1456 ;       AND THAT DT BUSY IS SET
                        0002  1457 ;
                        0002  1458 ; CALLING SEQUENCE:
                        0002  1459 ;
                        0002  1460 ;       PUSHL   RH750 ADDRESS
                        0002  1461 ;       PUSHL   DRIVE REGISTER ADDRESS
                        0002  1462 ;       CALLS   #2,ISSUE_MMREAD
                        0002  1463 ;--
      55  00000000'EF   D0  0002  1464          MOVL    RH_CUR_ADR,R5           ; MOVE ADDRESS OF RH750 TO R5
      54  00000026'EF   D0  0009  1465          MOVL    DRIVE0,R4               ; MOVE ADDRESS OF DRIVE0 TO R4
        04 A5    08     C8  0010  1466          BISL    #MAINT_MODE,4(R5)       ; PUT RH INTO MAINTENANCE MODE
           64    39     D0  0014  1467          MOVL    #READ,(R4)              ; ISSUE READ TO DRIVE
    14 A5   02000000 8F   C8  0017  1468          BISL    #SIM_OCC,20(R5)         ; ASSERT OCCUPIED
                        001F  1469 ISSUE_MMREADX:
                    04  001F  1470          RET                             ; RETURN
```

```
                          0020  1471          .SBTTL  MM_XFER SUBROUTINE
                      00000000  1472          .PSECT  MM_XFER, WORD
                          0000  1473 MM_XFER::
                        0FFC  0000  1474                  .WORD    M<R2,R3,R4,R5,R6,R7>   ; ENTRY MASK
                          0002  1475 ;++
                          0002  1476 ; FUNCTIONAL DESCRIPTION:
                          0002  1477 ;
                          0002  1478 ;       THIS ROUTINE PUTS THE RH750 UNDER TEST INTO THE MAINTENANCE
                          0002  1479 ;       MODE, ISSUES OCCUPIED, AND THEN ASSERTS-DEASSERTS SIMSCLK
                          0002  1480 ;       N MOD 2 TIMES (N IS AN ARGUMENT WHICH IS PASSED AS A PARAMETER).
                          0002  1481 ;
                          0002  1482 ;       THE TRANSFER IS STOPPED WHEN N MODULO 2 SCLKS ARE ASSERTED-DEASSERTED
                          0002  1483 ;       AND EBL IS SUBSEQUENTLY ISSUED.
                          0002  1484 ;
                          0002  1485 ;CALLING SEQUENCE:
                          0002  1486 ;
                          0002  1487 ;       PUSHL   DRIVE COMMAND
                          0002  1488 ;       PUSHL   N ( NUMBER OF BYTES  TO XFER)
                          0002  1489 ;       PUSHL   RH_ADDRESS
                          0002  1490 ;       CALLS   #3,MM_XFER
                          0002  1491 ;--
            53    08 AC   D0    0002  1492          MOVL    8(AP),R3                ; MOVE BYTE COUNT TO R3
                    03    12    0006  1493          BNEQ    1$                      ; IF BYTE COUNT = 0 THEN EXIT
                  006F    31    0008  1494          BRW     MM_XFERX                ; BRANCH TO EXIT
                    53    D6    000B  1495 1$:      INCL    R3                      ; REDUCE BYTE COUNT MODULO 2
          53    53    FF 8F   78    000D  1496          ASHL    #-1,R3,R3               ; REDUCE BYTE COUNT MOD 2
                          0012  1497                                                ; FOR EVERY SCLK ASSERTED
            52    04 AC   D0    0012  1498          MOVL    4(AP),R2                ; MOVE RH750 ADDRESS INTO R2
       08 A2    08 A2   D0    0016  1499          MOVL    SR(R2),SR(R2)           ; CLEAR STATUS BITS
          04 A2    08    C8    001B  1500          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
       54   00000046'EF   D0    001F  1501          MOVL    DRIVE_INUSE,R4          ; PUT DRIVE ADDRESS IN R4
            64    0C AC   D0    0026  1502          MOVL    12(AP),(R4)             ; ISSUE DRIVE COMMAND
    14 A2   02000000 8F   C8    002A  1503          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                          0032  1504          $DS_WAITUS_S    #1                      ; WAIT 10 MICRO SECONDS
                      00  DD    0032                PUSHL   #0
                      01  DD    0034                PUSHL   #1
       00010068 9F    02  FB    0036                CALLS   #2, a#DS$WAITUS
            55    14 A2   D0    003D  1505          MOVL    DR(R2),R5               ; MOVE DIAGNOSTIC REGISTER TO R5
                          0041  1506                                                ; TO GET THE FASTEST SCLK POSSIBLE
  56    55    08000000 8F   C9    0041  1507          BISL3   #SIM_SCLK,R5,R6         ; R5 = SCLK CLEAR. R6 = SCLK SET
            57    14 A2   DE    0049  1508          MOVAL   DR(R2),R7               ; R7 = ADDR OF DR (TO SPEED UP SCLK)
               67    56   D0    004D  1509 10$:     MOVL    R6,(R7)                 ; ASSERT SCLK
                      01  0050  1510          NOP
                      01  0051  1511          NOP
                      01  0052  1512          NOP
               67    55   D0    0053  1513          MOVL    R5,(R7)                 ; NEGATE SCLK
                 F4  53   F5    0056  1514          SOBGTR  R3,10$                  ; IF BYTE COUNT NOT EXHAUSTED LOOP
                          0059  1515
    0000005E'EF    1C A2   D0    0059  1516 20$:     MOVL    CAR(R2),CAR_SNAP        ; DO THE SAME FOR THE CAR
    14 A2   04000000 8F   C8    0061  1517          BISL    #SIM_EBL,DR(R2)         ; SIMULATE END OF SECTOR
    14 A2   04000000 8F   CA    0069  1518          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
    14 A2   02000000 8F   CA    0071  1519          BICL    #SIM_OCC,DR(R2)         ; NEGATE OCCUPIED
                      01  0079  1520          NOP
                          007A  1521 MM_XFERX:
                          007A  1522 30$:
                      04  007A  1523          RET                             ; EXIT
```

```
                        007B  1524          .SBTTL  RH11TB_PRES SUBROUTINE
                    00000000  1525          .PSECT  RH11TB_PRES, WORD
                        0000  1526 RH11TB_PRES::
                        0000  1527
                        0000  1528 ;++
                        0000  1529 ;       SUBROUTINE RH11TB_PRES
                        0000  1530 ;
                        0000  1531 ;               THIS SUBROUTINE IS CALLED AT THE BEGINNING OF TESTS 7-19
                        0000  1532 ;       TO DETERMINE IF AN RH11TB IS PRESENT. IF PRESENT, THE ROUTINE THEN
                        0000  1533 ;       SETS BIT<16> IN THE LOC. RH11TB_MSK. IF NOT PRESENT, THE MASK
                        0000  1534 ;       LOCATION IS CLEARED. THIS ROUTINE IS NECESSARY SINCE THE ATTENTION
                        0000  1535 ;       BIT, SR<16> WILL BE ASSERTED WHENEVER MB MAINTENANCE MODE IS SET
                        0000  1536 ;       AND AN RH11TB IS PRESENT.
                        0000  1537 ;--
                        0000  1538
     52 00000000'EF     D0    0000  1539          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF RH750
      04 A2    01      C8    0007  1540          BISL    #PGM_INIT,CR(R2)        ; INIT RH750
     58 00000004'EF     D0    000B  1541          MOVL    MBE_CUR_ADR,R8          ; IS MBE SELECTED?
                07      12    0012  1542          BNEQ    5$                      ; BRANCH IF YES
     58 00000042'EF     D0    0014  1543          MOVL    DRIVE7,R8               ; GET ADDRESS OF DRIVE 7
        54    20        D0    001B  1544 5$:      MOVL    #TYPE_MBE,R4            ; EXPECT MBE IDENTIFIER
      50    18 A8      D0    001E  1545          MOVL    MBE_DTR(R8),R0          ; READ MBE DRIVE TYPE REGISTER
         53    50      3C    0022  1546          MOVZWL  R0,R3                   ; EXTRACT CONTENTS OF REGISTER
         54    53      D1    0025  1547          CMPL    R3,R4                   ; IS DRIVE 7 AN MBE?
                0D      13    0028  1548          BEQL    10$                     ; BRANCH IF IT IS
000002DE'EF     00010000 6F    CA    002A  1549          BICL    #ATTENTION,RH11TB_MSK   ; CLEAR EXPECTED BIT
                0B      11    0035  1550          BRB     20$                     ; GO TO RETURN
000002DE'EF     00010000 8F    C8    0037  1551 10$:     BISL    #ATTENTION,RH11TB_MSK   ; SET EXPECTED BIT
                05      0042  1552 20$:     RSB                             ; RETURN
```

```
                        0043  1553          .SBTTL  DRIVES PRESENT
                    00000000  1554          .PSECT  DRIVE_PRES, WORD
                        0000  1555 ;++
                        0000  1556 ; FUNCTIONAL DESCRIPTION:
                        0000  1557 ;
                        0000  1558 ;       THIS ROUTINE ACCESSES ALL DRIVES OF THE RH750 UNDER TEST
                        0000  1559 ;       TO DETERMINE IF ANY DRIVE IS PRESENT. IF DRIVE 7 IS AN MBE
                        0000  1560 ;       IT IS NOT COUNTED AS 'ANY DRIVE'.
                        0000  1561 ;
                        0000  1562 ; ROUTINE LINKAGE:
                        0000  1563 ;
                        0000  1564 ;       JSB
                        0000  1565 ;
                        0000  1566 ; EXPLICIT INPUTS:
                        0000  1567 ;
                        0000  1568 ;       R2 - CONTAINS BASE ADDRESS OF RH750 UNDER TEST.
                        0000  1569 ;
                        0000  1570 ; ROUTINE VALUE:
                        0000  1571 ;
                        0000  1572 ;       R0 - 1 IF NO DRIVES PRESENT
                        0000  1573 ;            0 OTHERWISE
                        0000  1574 ;--
                        0000  1575
                        0000  1576 DRIVES_PRESENT::
                  38    BB    0000  1577          PUSHR   #^M<R3,R4,R5>
      53  00000026'EF   D0    0002  1578          MOVL    DRIVE0,R3               ; GET ADDRESS OF DRIVE 0
                  54    D4    0009  1579          CLRL    R4                      ; CLEAR DRIVE COUNTER
           04 A2  01    C8    000B  1580 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
              55  63    D0    000F  1581          MOVL    (R3),R5                 ; TRY TO READ DRIVE REGISTER
           55  08 A2    D0    0012  1582          MOVL    SR(R2),R5               ; READ STATUS REG
           09 55   12    E0    0016  1583          BBS     #18,R5,20$              ; BRANCH IF NED SET
                        001A  1584 ;
                        001A  1585 ; DRIVE IS PRESENT, CHECK IF THIS IS AN MBE
                        001A  1586 ;
           55  18 A3    D0    001A  1587          MOVL    MBE_DTR(R3),R5          ; GET DRIVE TYPE
              55  20    B1    001E  1588          CMPW    #TYPE_MBE,R5            ; IS THIS AN MBE?
                  12    12    0021  1589          BNEQ    30$                     ; NO, TAKE FAILURE EXIT
      53  00000080 8F    C0    0023  1590 20$:     ADDL    #DRIVE_OFFSET,R3        ; SELECT NEXT DRIVE ADDRESS
FFDB  54  01        07    3D    002A  1591          ACBW    #7,#1,R4,10$            ; TRY ALL 8 DRIVES
              50  01    D0    0030  1592          MOVL    #1,R0                   ; SET SUCCESS RETURN
                  02    11    0033  1593          BRB     40$                     ; EXIT
                  50    D4    0035  1594 30$:     CLRL    R0                      ; SET FAILURE RETURN CODE
                  38    BA    0037  1595 40$:     POPR    #^M<R3,R4,R5>
                  05    0039  1596          RSB
                        003A  1597
                        003A  1598 $DS_ENDMOD
                        003A  1599          .END
```

```
$$ARGS                          = 0000000A
$$E                             = 00000001
$$M                             = 00000004
$$N                             = 00000002
$$S                             = FFFFFFFF
$$T1                            = 0000002C
$ENV                            = 00000001  G
$ER                             = 00000001
$MO                             = 00000001  G
$ST                             = 00000000
$TN                             = 00000000
ABORT_IF_FAIL                     00000059 RG     08
ALL                             = 00000001  G
ALL_ONES                        = 0000FFFF  G
AL_DEVTYP                         0000002B R      01
ARG_LIST                          00000062 RG     08
ARRAY_MSG1                        000002EA RG     08
ARRAY_MSG10                       000003E1 RG     08
ARRAY_MSG11                       000003F9 RG     08
ARRAY_MSG12                       00000411 RG     08
ARRAY_MSG13                       00000429 RG     08
ARRAY_MSG14                       00000456 RG     08
ARRAY_MSG15                       00000483 RG     08
ARRAY_MSG16                       000004B0 RG     08
ARRAY_MSG17                       000004D7 RG     08
ARRAY_MSG18                       000004F4 RG     08
ARRAY_MSG19                       00000513 RG     08
ARRAY_MSG2                        0000031E RG     08
ARRAY_MSG20                       00000534 RG     08
ARRAY_MSG21                       00000553 RG     08
ARRAY_MSG22                       0000056E RG     08
ARRAY_MSG23                       000005A1 RG     08
ARRAY_MSG24                       000005D4 RG     08
ARRAY_MSG26                       00000607 RG     08
ARRAY_MSG27                       0000062E RG     08
ARRAY_MSG28                       00000653 RG     08
ARRAY_MSG29                       00000686 RG     08
ARRAY_MSG3                        00000345 RG     08
ARRAY_MSG30                       000006B9 RG     08
ARRAY_MSG31                       000006DE RG     08
ARRAY_MSG32                       000006FD RG     08
ARRAY_MSG33                       0000071C RG     08
ARRAY_MSG34                       0000073B RG     08
ARRAY_MSG35                       0000075A RG     08
ARRAY_MSG36                       00000779 RG     08
ARRAY_MSG37                       00000798 RG     08
ARRAY_MSG38                       000007B7 RG     08
ARRAY_MSG39                       000007D6 RG     08
ARRAY_MSG4                        0000035A RG     08
ARRAY_MSG40                       000007F5 RG     08
ARRAY_MSG41                       00000814 RG     08
ARRAY_MSG42                       00000833 RG     08
ARRAY_MSG43                       00000852 RG     08
ARRAY_MSG44                       00000871 RG     08
ARRAY_MSG45                       00000890 RG     08
ARRAY_MSG46                       000008AF RG     08
ARRAY_MSG47                       000008D0 RG     08
ARRAY_MSG48                       000008F1 RG     08
ARRAY_MSG49                       00000916 RG     08
ARRAY_MSG5                        0000036F RG     08
ARRAY_MSG50                       0000093B RG     08
ARRAY_MSG51                       00000960 RG     08
ARRAY_MSG52                       0000098B RG     08
ARRAY_MSG53                       000009B6 RG     08
ARRAY_MSG54                       000009E1 RG     08
ARRAY_MSG55                       00000A03 RG     08
ARRAY_MSG56                       00000A25 RG     08
ARRAY_MSG57                       00000A47 RG     08
ARRAY_MSG58                       00000A63 RG     08
ARRAY_MSG59                       00000A7F RG     08
ARRAY_MSG6                        00000384 RG     08
ARRAY_MSG60                       00000A9B RG     08
ARRAY_MSG61                       00000AB7 RG     08
ARRAY_MSG62                       00000AD3 RG     08
ARRAY_MSG7                        00000399 RG     08
ARRAY_MSG8                        000003B1 RG     08
ARRAY_MSG9                        000003C9 RG     08
ASCII_MBE                         0000197F RG     08
ASR_OFFSET                      = 00000410  G
ATA                             = 00008000  G
ATTENTION                       = 00010000  G
ATTN                            = 00000018  G
A_HEADEND                         00000058 R      04
BCR                             = 00000010  G
BIT...                          = 00660130
BIT0                            = 00000001
BIT1                            = 00000002
BIT10                           = 00000400
BIT11                           = 00000800
BIT12                           = 00001000
BIT13                           = 00002000
BIT14                           = 00004000
BIT15                           = 00008000
BIT16                           = 00010000
BIT17                           = 00020000
BIT18                           = 00040000
BIT19                           = 00080000
BIT2                            = 00000004
BIT20                           = 00100000
BIT21                           = 00200000
BIT22                           = 00400000
BIT23                           = 00800000
BIT24                           = 01000000
BIT25                           = 02000000
BIT26                           = 04000000
BIT27                           = 08000000
BIT28                           = 10000000
BIT29                           = 20000000
BIT3                            = 00000008
BIT30                           = 40000000
BIT31                           = 80000000
BIT4                            = 00000010
BIT5                            = 00000020
BIT6                            = 00000040
```

```
BIT7                            = 00000080
BIT8                            = 00000100
BIT9                            = 00000200
BLANK                             00001987 RG     08
BLKSCSBI                        = 0000001C  G
BLKSND_COMD                     = 10000000  G
BR4                             = 00000014  G
BR5                             = 00000015  G
BR6                             = 00000016  G
BR7                             = 00000017  G
BYTE0                           = 000000FF  G
BYTE1                           = 0000FF00  G
BYTE2                           = 00FF0000  G
BYTE3                           = FF000000  G
BYTE_COUNT_MSK                  = 0000FFFF  G
CAR                             = 0000001C  G
CAR_SNAP                          0000005E RG     08
CB_HUNG                         = 00800000  G
CEP_FUNCTIONAL                  = 00000000
CEP_REPAIR                      = 00000001
CLEAN_UP                          00000000 R      27
CLEAN_UP_X                        0000004F R      27
COMP_MSG1                         00000AF5 RG     08
COMP_MSG2                         00000B42 RG     08
CONT_PC                           00000055 RG     08
CPE                             = 00000008  G
CR                              = 00000004  G
CRTED_READ_DATA                 = 20000000  G
CSR                             = 00000000  G
DATA_IBUFFER                      00000000 RG     09
DATA_OBUFFER                      00000200 RG     09
DATA_XFER_ABRT                  = 00001000  G
DATA_XFER_DONE                  = 00002000  G
DATA_XFER_LATE                  = 00000800  G
DCM                             = 04000000  G
DEFAULT                         = 00000000  G
DEVICES                           0000000C RG     08
DEVICE_PTR                        000002E2 RG     08
DEV_REG                           000002C2 RG     08
DIF_MSG                           00001A74 RG     08
DISABLE_LOG                     = 30000000  G
DISPATCH                          00000000 R      02
DMD                             = 00000001  G
DOCC                            = 00000100  G
DPE                             = 00000010  G
DPR                             = 00000100  G
DR                              = 00000014  G
DRIVE0                            00000026 RG     08
DRIVE1                            0000002A RG     08
DRIVE2                            0000002E RG     08
DRIVE3                            00000032 RG     08
DRIVE4                            00000036 RG     08
DRIVE5                            0000003A RG     08
DRIVE6                            0000003E RG     08
DRIVE7                            00000042 RG     08
DRIVES_PRESENT                    00000000 RG     2D
DRIVE_ADR_TBL                     00000026 RG     08
DRIVE_INUSE                       00000046 RG     08
DRIVE_OFFSET                    = 00000080  G
DRIVE_SEL_MSK                   = 0000E000  G
DRV_ERRMASK                     = 0000E000  G
DRV_INITMASK                    = 00001180  G
DRY                             = 00000080  G
DS$ABORT                          00010020  G
DS$ASKADR                         00010090  G
DS$ASKDATA                        00010080  G
DS$ASKLGCL                        00010098  G
DS$ASKSTR                         000100A0  G
DS$ASKVLD                         00010088  G
DS$ATTACH                         000101A8  G
DS$BGNSUB                         00010030  G
DS$BRANCH                         000100A8  G
DS$BREAK                          00010058  G
DS$CANWAIT                        00010070  G
DS$CHANNEL                        00010180  G
DS$CKLOOP                         00010040  G
DS$CLRVEC                         00010168  G
DS$CNTRLC                         00010078  G
DS$CVTREG                         000100B0  G
DS$ENDPASS                        00010010  G
DS$ENDSUB                         00010038  G
DS$ERRDEV                         000100C8  G
DS$ERRHARD                        000100D0  G
DS$ERRPREP                        00010108  G
DS$ERRSOFT                        000100D8  G
DS$ERRSYS                         000100C0  G
DS$ESCAPE                         00010050  G
DS$FREEDBGSYM                     000101B8  G
DS$GETBUF                         00010120  G
DS$GETMEM                         00010130  G
DS$GETTERM                        000101C8  G
DS$GPHARD                         00010018  G
DS$HELP                           000101B0  G
DS$INITSCB                        00010170  G
DS$INLOOP                         00010048  G
DS$K_ERROR                      = 00000002  G
DS$K_NORMAL                     = 00000001  G
DS$K_SEVERE                     = 00000004  G
DS$K_SUBSYS                     = 00000066  G
DS$K_WARNING                    = 00000000  G
DS$LOAD                           00010198  G
DS$LOADPCS                        000101C0  G
DS$MAPDBGBLOCK                    00010118  G
DS$MMOFF                          00010158  G
DS$MMON                           00010150  G
DS$MOVPHY                         00010148  G
DS$MOVVRT                         00010140  G
DS$PARSE                          000100B8  G
DS$PRINTB                         000100E0  G
DS$PRINTF                         000100F0  G
DS$PRINTS                         000100F8  G
DS$PRINTSIG                       00010100  G
DS$PRINTX                         000100E8  G
DS$PROBE                          000101A0  G
```

```
DS$RELBUF                       00010128  G
DS$RELMEM                       00010138  G
DS$SERVICE_DISPATCHER           000101D0  G
DS$SETIPL                       00010178  G
DS$SETMAP                       00010188  G
DS$SETPRIEXV                    00010110  G
DS$SETVEC                       00010160  G
DS$SHOCHAN                      00010190  G
DS$SUMMARY                      00010028  G
DS$WAITMS                       00010060  G
DS$WAITUS                       00010068  G
DS$_ARITH                   =   006600D0  G
DS$_ASBE                    =   00660118  G
DS$_BADLINK                 =   006600F0  G
DS$_BADTYPE                 =   006600E8  G
DS$_BIIC                    =   00660120  G
DS$_CHME                    =   006600A8  G
DS$_CHMK                    =   006600E0  G
DS$_DEVNAME                 =   00660108  G
DS$_ERROR                   =   00660002  G
DS$_FHWE                    =   00660068  G
DS$_FRAGBUF                 =   00660080  G
DS$_ICBUSY                  =   006600C8  G
DS$_ICERR                   =   006600C0  G
DS$_IHWE                    =   00660060  G
DS$_ILLCHAR                 =   00660018  G
DS$_ILLPAGCNT               =   00660078  G
DS$_ILLUNIT                 =   00660100  G
DS$_INSFMEM                 =   00660050  G
DS$_IPL2HI                  =   006600B8  G
DS$_IVADDR                  =   00660040  G
DS$_IVVECT                  =   00660038  G
DS$_KRNLSTK                 =   00660090  G
DS$_LOGIC                   =   00660070  G
DS$_MCHK                    =   00660088  G
DS$_MMOFF                   =   00660058  G
DS$_NEEDUNIT                =   006600F8  G
DS$_NODE                    =   00660128  G
DS$_NOPCS                   =   00660110  G
DS$_NORMAL                  =   00660001  G
DS$_NOSUPPORT               =   006600B1  G
DS$_NOTDON                  =   00660030  G
DS$_NOTIMP                  =   006600B0  G
DS$_NULLSTR                 =   00660010  G
DS$_OVERFLOW                =   00660008  G
DS$_POWER                   =   00660098  G
DS$_PROGERR                 =   00660020  G
DS$_SEVERE                  =   00660004  G
DS$_TRANSL                  =   006600A0  G
DS$_TRUNCATE                =   00660028  G
DS$_UNEXPINT                =   006600D8  G
DS$_VASFULL                 =   00660048  G
DS$_WARNING                 =   00660000  G
DSA$AL_APTMAIL                  0000FE00
DSA$AT_APTTXT                   0000FA00
DSA$GL_APTCOM                   0000FE04
DSA$GL_DEVLEN                   0000FE58
DSA$GL_ERRNO                    0000FE44
DSA$GL_EVENT                    0000FE48
DSA$GL_FLAGS                    0000FE00
DSA$GL_MSGTYP                   0000FE40
DSA$GL_PASSES                   0000FE08
DSA$GL_PASSNO                   0000FE54
DSA$GL_SECTNO                   0000FE10
DSA$GL_SID                      0000FE14
DSA$GL_SUBTNO                   0000FE4C
DSA$GL_TESTNO                   0000FE50
DSA$GL_UNITS                    0000FE0C
DSA$GQ_MSGPTR                   0000FE68
DSA$GT_DEVNAM                   0000FE5C
DSA$M_TRACE                 =   00000400
DSA$V_PASS0                 =   0000001D
DS_MSK                      =   0FFFF8FF  G
DTE                         =   00001000  G
DT_ABORT                    =   00000002  G
DT_BUSY                     =   80000000  G
DVA                         =   00000800  G
EBL                         =   0000001A  G
EDM                         =   02000000  G
ENABLE_PS                   =   00000800  G
ENV$M_DOMAIN                =   00000002
ENV$M_LEVEL                 =   00000001
ENV$M_SUPER                 =   000003FC
ENV$S_DOMAIN                =   00000001
ENV$S_LEVEL                 =   00000001
ENV$S_SUPER                 =   00000008
ENV$V_DOMAIN                =   00000001
ENV$V_LEVEL                 =   00000000
ENV$V_SUPER                 =   00000002
ENV$_CPU                    =   00000000
ENV$_FUNCTIONAL             =   00000000
ENV$_REPAIR                 =   00000001
ENV$_SUPER                  =   00000001
ENV$_SYSTEM                 =   00000001
ERCE                        =   10000000  G
ERR                         =   00004000  G
ERR$_MSGADR                 =   0000000C
ERR$_NARGS                  =   0000000A
ERR$_NUM                    =   00000004
ERR$_P1                     =   00000014
ERR$_P2                     =   00000018
ERR$_P3                     =   0000001C
ERR$_P4                     =   00000020
ERR$_P5                     =   00000024
ERR$_P6                     =   00000028
ERR$_POINTER                =   00000010
ERR$_UNIT                   =   00000008
ERR_STAT                    =   00000008  G
EXCEPTION                   =   00000015  G
EXISTING_DRIVE                  0000004A RG     08
EXP_INTERRUPT                   00000052 RG     08
EXP_MSG                         00001A68 RG     08
EXT_REG_MSG                     00001985 RG     08
EXT_REG_OFFSET              =   00000400  G
```

FAIL	=	00000013	G		FMT_SERR		00001B0C	RG	08		
FERR	=	00000004	G		FMT_SILOERROR		000014F7	RG	08		
FMT_ALLDRVPRES		000012D2	RG	08	FMT_STATUS_REG		00000C42	RG	08		
FMT_ANY_REG		00000DC5	RG	08	FMT_UNEXP_INT		00001B50	RG	08		
FMT_BADNEXUS		00001301	RG	08	FMT_VAL		0000190F	RG	08		
FMT_BODY		000014C5	RG	08	FMT_VARER		00001399	RG	08		
FMT_BYTMSK_ER		000017B8	RG	08	FMT_VARINCERR		000013E3	RG	08		
FMT_CMICMD_ERR		000017A4	RG	08	FMT_VECERR		00001AE3	RG	08		
FMT_COMP_ERR		000014D9	RG	08	FMT_XFERERR		00001348	RG	08		
FMT_CONTRL_REG		00000B8F	RG	08	HP$A_DEPENDENT		00000032				
FMT_DCMPERR		00001709	RG	08	HP$A_DEVICE		00000018				
FMT_DIAG_REG		00000D09	RG	08	HP$A_DVA		0000001C				
FMT_DRIVENORES		000012BC	RG	08	HP$A_LINK		00000020				
FMT_DRIVERES		000012A5	RG	08	HP$B_DRIVE		0000000B				
FMT_DTERR		000016A9	RG	08	HP$B_FLAGS		0000000A				
FMT_DUADR_ERR		0000144C	RG	08	HP$B_RH750_BR		00000032				
FMT_DUALRS_ERR		00001427	RG	08	HP$K_RH750_LEN		00000033				
FMT_DUALRS_MSG		000019FC	RG	08	HP$Q_DEVICE		00000000				
FMT_DUALWT_ERR		00001407	RG	08	HP$T_DEVICE		0000000C				
FMT_DUALWT_MSG		000019A5	RG	08	HP$T_TYPE		00000026				
FMT_ERRSUM		00001782	RG	08	HP$W_SIZE		00000008				
FMT_EXCP_ERR		000015A6	RG	08	HP$W_VECTOR		00000024				
FMT_EXTREG_ERR		00001654	RG	08	IBC_MSG		000002DA	RG	08		
FMT_FATAL_MMM		00001231	RG	08	ILF	=	00000001	G			
FMT_HEADER		00001494	RG	08	IMAPP	=	0000001D	G			
FMT_IBCER1		000017D0	RG	08	IMBCP	=	0000001E	G			
FMT_IBCER2		000017E0	RG	08	IMBDP	=	0000001F	G			
FMT_IBCER3		000017F0	RG	08	INDEX_MSG		00001AAE	RG	08		
FMT_IBCER4		00001809	RG	08	INH_BYTE_CNT	=	00000010	G			
FMT_IBCER5		00001819	RG	08	INITIALIZE		00000000	R	26		
FMT_IBCER6		0000184A	RG	08	INITIALIZE_X		000001E5	R	26		
FMT_IBCER7		00001869	RG	08	INTERRUPT_EN	=	00000004	G			
FMT_IBCER8		000018C3	RG	08	INT_OCCURRED		00000053	RG	08		
FMT_IBCER9		000018E2	RG	08	INVRT_MAP_PAR	=	20000000	G			
FMT_IBUS0		0000127C	RG	08	INVRT_MB_CPAR	=	40000000	G			
FMT_IBUS1		00001253	RG	08	INVRT_MB_DPAR	=	80000000	G			
FMT_INIT1		00001216	RG	08	INVRT_SILO_PAR	=	00400000	G			
FMT_INTFAILED		00001B7F	RG	08	IPLR	=	00000012	G			
FMT_LOCATION		000011F7	RG	08	ISP	=	00000016	G			
FMT_MAPFUN_ERR		000015D7	RG	08	ISR0		00000000	RG	0A		
FMT_MAP_EXERR		000015EE	RG	08	ISR1		00000000	RG	0B		
FMT_MAP_HEAD		000014AA	RG	08	ISR10		00000000	RG	14		
FMT_MAP_REG		00000E7B	RG	08	ISR11		00000000	RG	15		
FMT_MBEREG_ERR		0000167E	RG	08	ISR12		00000000	RG	16		
FMT_MBE_ASR		00001137	RG	08	ISR13		00000000	RG	17		
FMT_MBE_CR1		00000F30	RG	08	ISR14		00000000	RG	18		
FMT_MBE_ER		00001086	RG	08	ISR15		00000000	RG	19		
FMT_MBE_SR		00000FD8	RG	08	ISR2		00000000	RG	0C		
FMT_MISSEDXFER		00001564	RG	08	ISR3		00000000	RG	0D		
FMT_NOINTEST		00001BB9	RG	08	ISR4		00000000	RG	0E		
FMT_NOTHING		000014D0	RG	08	ISR5		00000000	RG	0F		
FMT_NO_MBE		0000193A	RG	08	ISR6		00000000	RG	10		
FMT_NO_MBE_SEL		0000195C	RG	08	ISR7		00000000	RG	11		
FMT_NO_RH750		00001B31	RG	08	ISR8		00000000	RG	12		
FMT_NX_ER		00001AB6	RG	08	ISR9		00000000	RG	13		
FMT_PTR		000002BA	RG	08	ISR_TBL		00001C3E	RG	08		
FMT_RHS		000011DF	RG	08	ISSUE_MMREAD		00000000	RG	2A		

```
ISSUE_MMREADX                   0000001F R      2A
L$A_CCP                         00000040 R      04
L$A_DEVP                        0000001C R      04
L$A_DREG                        00000024 R      04
L$A_DTP                         00000018 R      04
L$A_ICP                         0000C03C R      04
L$A_LASTAD                      00000014 R      04
L$A_NAME                        00000008 R      04
L$A_REPP                        00000044 R      04
L$A_SECNAM                      00000050 R      04
L$A_STATAB                      00000048 R      04
L$A_TSTCNT                      00000054 R      04
L$L_ENVIRON                     00000004 R      04
L$L_ERRTYP                      0000004C R      04
L$L_HEADLENGTH                  00000000 R      04
L$L_REV                         0000000C R      04
L$L_UNIT                        00000020 R      04
L$L_UPDATE                      00000010 R      04
LASTAD                          00000000 R      05
LN                              00000008 R      08
LOWBITS_MSK                   = 0000003F  G
LUN                             0000001C RG     08
L_TSTCNT                        00000000 R      06
MACH_CHK_SRV                    00000000 RG     28
MAINT_MODE                    = 00000008  G
MAP_1_PATTRN                  = 80007FFF  G
MAP_INVALID                   = 00000010  G
MAP_OFFSET                    = 00000800  G
MAP_PE                        = 00000020  G
MAP_PTR_MSK                   = 0001FE00  G
MASS_CNTRL_PE                 = 00020000  G
MASS_CTOD                     = 00010000  G
MASS_DATA_PE                  = 00000040  G
MASS_ECP                      = 00000080  G
MASS_EXCP                     = 00020000  G
MASS_FAIL                     = 00100000  G
MASS_RUN                      = 00080000  G
MASS_WCLK                     = 00040000  G
MBA_LN                          00000014 R      08
MBDIB_SEL                     = 00800000  G
MBE                           = 00000002  G
MBE_ASR                       = 00000010  G
MBE_CR1                       = 00000000  G
MBE_CR2                       = 00000014  G
MBE_CUR_ADR                     00000004 RG     08
MBE_DBR                       = 0000001C  G
MBE_DTR                       = 00000018  G
MBE_ER                        = 00000008  G
MBE_MR                        = 0000000C  G
MBE_PARAM                     = 00000087  G
MBE_REG_MSG                     00001983 RG     08
MBE_SR                        = 00000004  G
MCLK                          = 00000002  G
MEM_CSR0                      = 40F20000  G
MEM_CSR1                      = 40F20004  G
MIOBUFFER                       00000400 RG     09
MISSED_XFER                   = 00000100  G
MM_XFER                         00000000 RG     2B
MM_XFERX                        0000007A R      2B
MOL                           = 00001000  G
NIBBLE0                       = 0000000F  G
NIBBLE1                       = 000000F0  G
NIBBLE2                       = 00000F00  G
NIBBLE3                       = 0000F000  G
NIBBLE4                       = 000F0000  G
NIBBLE5                       = 00F00000  G
NIBBLE6                       = 0F000000  G
NIBBLE7                       = F0000000  G
NODRIVE                         0000004E RG     08
NON_XIST_DRIVE                = 00040000  G
NOOP                          = 00000000  G
NO_MBE                          00000021 RG     08
NO_MBE_MSG                      00001989 RG     08
NO_RESPONSE                   = 00000002  G
NO_RH750                        00000020 RG     08
NUMBER_BUFFER                   000002AA RG     08
NXM_ADR                         000002D6 RG     08
NXM_RD_FLAG                     00000054 RG     08
OCC                           = 00000019  G
OPI                           = 00002000  G
PAGE_BYTE_MSK                 = 000001FF  G
PF_FIELD                      = 00000009  G
PF_WIDTH                      = 00000015  G
PGM_INIT                      = 00000001  G
PIP                           = 00002000  G
PM                            = 08000000  G
PPFN_MASK                     = 00007FFF  G
PRINT_CMDER                     00000000 RG     22
PRINT_CMDERX                    0000002E R      22
PRINT_DTERR                     00000000 RG     1E
PRINT_DTERRX                    0000001F R      1E
PRINT_IBCERR                    00000000 RG     23
PRINT_IBCERRX                   00000125 R      23
PRINT_INTERR                    00000000 RG     1F
PRINT_INTERRX                   00000000 RG     20
PRINT_MAPERR                    00000000 RG     1D
PRINT_MAPERRX                   00000024 R      1D
PRINT_NULL                      00000000 RG     1B
PRINT_NULL_X                    00000002 RG     1B
PRINT_NXADR                     00000000 RG     21
PRINT_NXADRX                    00000015 R      21
PRINT_SBE                       00000000 RG     1C
PRINT_SBEX                      000001CD R      1C
PRINT_SILOER                    00000000 RG     25
PRINT_SILOERX                   0000002E R      25
PRINT_VECTERR                   00000000 RG     24
PRINT_VECTERRX                  0000002E R      24
PROGRAM_ERROR                 = 00080000  G
PTBASE                          00000022 RG     08
RANDOM                        = 00008259  G
RCPA                          = 00000008  G
RDPA                          = 00000010  G
READ                          = 00000039  G
READ_HEADER                   = 0000003B  G
```

```
READ_REV                        = 0000003F  G
REC_MSG                           00001A5C RG     08
REG_NAME                          000002AE RG     08
REG_NO                            000002B2 RG     08
REG_SEL_MSK                     = 00001F00  G
REG_STRING                        000002B6 RG     08
REPORT_BUFFER                     000000AA RG     08
RH11TB_MSK                        000002DE RG     08
RH11TB_PRES                       00000000 RG     2C
RH750                           = 00000001
RH750$B_BR                        00000032
RH750$K_LEN                       00000033
RH750_LINK                        00000010 RG     08
RHBCR_MSG                         00001A96 RG     08
RHCR_MSG                          00001A80 RG     08
RHDR_MSG                          00001A9E RG     08
RHMAPR_MSG                        00001AA5 RG     08
RHSR_CHECK                      = 000F5FFA  G
RHSR_MSG                          00001A87 RG     08
RHVAR_MSG                         00001A8E RG     08
RH_CUR_ADR                        00000000 RG     08
RH_ISR                            00000000 RG     29
RH_ISRX                           0000003B R      29
RH_VECS                           00001BFA RG     08
RH_VECTOR                         00000018 RG     08
RMR                             = 00000004  G
RRUN                            = 00000002  G
RS_MSK                          = 0FFF07FF  G
R_FAIL                          = 00000040  G
R_MB_EXC                        = 00000020  G
SA0_MSG                           00001A39 RG     08
SA1_MSG                           00001A44 RG     08
SCLK                            = 0000001B  G
SEARCH                          = 00000019  G
SECTION                           00000010 R      01
SEEK                            = 00000002  G
SEP_FUNCTIONAL                  = 00000002
SEP_REPAIR                      = 00000003
SILO_PE                         = 00004000  G
SIM_ATTN                        = 01000000  G
SIM_EBL                         = 04000000  G
SIM_EXCEPT                      = 00200000  G
SIM_OCC                         = 02000000  G
SIM_SCLK                        = 08000000  G
SIZ...                          = 00000001
SR                              = 00000008  G
SUFFIX_PTR                        000002BE RG     08
SUMMARY                           00000000 R      1A
SUMMARY_X                         00000002 R      1A
SYS$ALLOC                         00010238  G
SYS$ASCTIM                        00010248  G
SYS$ASSIGN                        00010250  G
SYS$BINTIM                        00010258  G
SYS$CANCEL                        00010260  G
SYS$CANTIM                        00010268  G
SYS$CLOSE                         000105B8  G
SYS$CLREF                         00010298  G
SYS$CONNECT                       000105C0  G
SYS$DALLOC                        000102D8  G
SYS$DASSGN                        000102E0  G
SYS$DISCONNECT                    000105D0  G
SYS$FAO                           00010350  G
SYS$FAOL                          00010358  G
SYS$GET                           00010580  G
SYS$GETCHN                        000104C8  G
SYS$GETTIM                        00010378  G
SYS$LKWSET                        000103A0  G
SYS$NUMTIM                        000103B8  G
SYS$OPEN                          00010608  G
SYS$QIO                           000103C8  G
SYS$QIOW                          00010200  G
SYS$READ                          00010590  G
SYS$READEF                        000103D0  G
SYS$SETAST                        000103F8  G
SYS$SETEF                         00010400  G
SYS$SETIMR                        00010420  G
SYS$SETPRI                        00010428  G
SYS$SETPRT                        00010430  G
SYS$SETRWM                        00010438  G
SYS$SETSWM                        00010448  G
SYS$ULKPAG                        00010460  G
SYS$ULWSET                        00010468  G
SYS$UNWIND                        00010470  G
SYS$WAITFR                        00010478  G
SYS$WFLAND                        00010488  G
SYS$WFLOR                         00010490  G
S_ALL                             00000008 R      01
S_DEFAULT                         000000C0 R      01
S_MBE                             0000000C R      01
TEMP                              000002C6 RG     08
TEMP1                             000002CA RG     08
TEMP2                             000002CE RG     08
TEMP3                             000002D2 RG     08
TEST_VEC                          00001C3A RG     08
TYPE_MBE                        = 00000020  G
T_MBE                             00000026 R      01
T_NAME                            00000058 R      04
T_RH750                           00000020 R      01
UNEXPECTED                        00001A4F RG     08
VALID_BIT                       = 80000000  G
VAR                             = 0000000C  G
VAR_PAT1                        = 00000778  G
VAR_PAT10                       = 00000800  G
VAR_PAT11                       = 00006F80  G
VAR_PAT12                       = 00001000  G
VAR_PAT13                       = 00005F80  G
VAR_PAT14                       = 00002000  G
VAR_PAT15                       = 00003F80  G
VAR_PAT16                       = 00004000  G
VAR_PAT17                       = 00017800  G
VAR_PAT18                       = 00008000  G
VAR_PAT19                       = 0000F800  G
VAR_PAT2                        = 00000080  G
VAR_PAT20                       = 00010000  G
```

```
VAR_PAT21                       = 00017FF8  G
VAR_PAT22                       = 00008000  G
VAR_PAT23                       = 0000FFF8  G
VAR_PAT24                       = 00010000  G
VAR_PAT3                        = 000006F8  G
VAR_PAT4                        = 00000100  G
VAR_PAT5                        = 000005F8  G
VAR_PAT6                        = 00000200  G
VAR_PAT7                        = 000003F8  G
VAR_PAT8                        = 00000400  G
VAR_PAT9                        = 00007780  G
WAIT_TIME                         0000005A RG    08
WCEL                            = 00000009  G
WCEU                            = 0000000A  G
WRITE                           = 00000031  G
WRITE_CHECK                     = 00000029  G
WRITE_CHECKHDR                  = 0000002B  G
WRITE_CHECKREV                  = 0000002F  G
WRITE_CHK_HIGH                  = 00000400  G
WRITE_CHK_LOW                   = 00000200  G
WRITE_HEADER                    = 00000033  G
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000037 (55.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
DISPATCH	00000000 (0.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
DISPATCH_X	00000018 (24.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
$HEADER	0000007B (123.)	04 (4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC PAGE
_LAST	00000000 (0.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE
$TSTCNT	00000000 (0.)	06 (6.)	NOPIC USR OVR REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
ABS	00010610 (67088.)	07 (7.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
OWN	00001C7E (7294.)	08 (8.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
BUFFERS	00000600 (1536.)	09 (9.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
ISR0	0000001D (29.)	0A (10.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR1	0000001D (29.)	0B (11.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR2	0000001D (29.)	0C (12.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR3	0000001D (29.)	0D (13.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR4	0000001D (29.)	0E (14.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR5	0000001D (29.)	0F (15.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR6	0000001D (29.)	10 (16.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR7	0000001D (29.)	11 (17.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR8	0000001D (29.)	12 (18.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR9	0000001D (29.)	13 (19.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR10	0000001D (29.)	14 (20.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR11	0000001D (29.)	15 (21.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR12	0000001D (29.)	16 (22.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR13	0000001D (29.)	17 (23.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR14	0000001D (29.)	18 (24.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISR15	0000001D (29.)	19 (25.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
SUMMARY	0000000A (10.)	1A (26.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
PRINT_NULL	00000003 (3.)	1B (27.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_SBE	000001CE (462.)	1C (28.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_MAPERR	00000025 (37.)	1D (29.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_DTERR	00000020 (32.)	1E (30.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_INTERR	0000003E (62.)	1F (31.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_INTERRX	00000001 (1.)	20 (32.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_NXADR	00000016 (22.)	21 (33.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_CMDER	0000002F (47.)	22 (34.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_IBCERR	00000126 (294.)	23 (35.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_VECTERR	0000002F (47.)	24 (36.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
PRINT_SILOER	0000002F (47.)	25 (37.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
INITIALIZE	000001ED (493.)	26 (38.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
CLEANUP	00000057 (87.)	27 (39.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
MACH_CHK_SRV	00000011 (17.)	28 (40.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
RH_ISR	00000050 (80.)	29 (41.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ISSUE_MMREAD	00000020 (32.)	2A (42.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
MM_XFER	0000007B (123.)	2B (43.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
RH11TB_PRES	00000043 (67.)	2C (44.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD
DRIVE_PRES	0000003A (58.)	2D (45.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC WORD

Symbol Cross Reference

SYMBOL	VALUE	DEFINITION	REFERENCES...
$$ARGS	=0000000A	87 (1)	87 (1)
$$E	=00000001	1371 (3)	1368 (2) 1387 (3)
$$M	=00000004	1434 (4)	1434 (4)
$$N	=00000002	1436 (4)	1000 (2) 1008 (2) 1011 (2) 1016 (2)
			1024 (2) 1032 (2) 1037 (2) 1045 (2)
			1050 (2) 1058 (2) 1062 (2) #-1088 (2)
			1090 (2) 1112 (2) 1132 (2) 1147 (2)
			#-1161 (2) 1163 (2) #-1191 (2) 1193 (2)
			#-1198 (2) 1200 (2) #-1202 (2) #-1207 (2)
			1209 (2) #-1211 (2) #-1213 (2) #-1215 (2)
			#-1217 (2) #-1222 (2) 1224 (2) #-1240 (2)
			1242 (2) #-1258 (2) 1260 (2) #-1297 (2)
			1335 (2) 1434 (4) 1436 (4) 79 (1)
			80 (1)
$$S	=FFFFFFFF	76 (1)	
$$T1	=0000002C	87 (1)	87 (1)
$ENV	=00000001	76 (1)	78 (1)
$ER	=00000001	1387 (3)	1265 (2) 1371 (3)
$MO	=00000001	1598 (4)	1598 (4)
$ST	=00000000	1387 (3)	1368 (2) 1387 (3) 1434 (4)
$TN	=00000000	76 (1)	1265 (2) 1371 (3) 1598 (4) 923 (1)
ABORT_IF_FAIL	00000059-R	469 (1)	
ALL	=00000001	79 (1)	
ALL_ONES	=0000FFFF	396 (1)	
AL_DEVTYP	0000002B-R	80 (1)	78 (1)
ARG_LIST	00000062-R	473 (1)	
ARRAY_MSG1	000002EA-R	526 (1)	
ARRAY_MSG10	000003E1-R	535 (1)	
ARRAY_MSG11	000003F9-R	536 (1)	
ARRAY_MSG12	00000411-R	537 (1)	
ARRAY_MSG13	00000429-R	538 (1)	
ARRAY_MSG14	00000456-R	539 (1)	
ARRAY_MSG15	00000483-R	540 (1)	
ARRAY_MSG16	000004B0-R	541 (1)	
ARRAY_MSG17	000004D7-R	542 (1)	
ARRAY_MSG18	000004F4-R	543 (1)	
ARRAY_MSG19	00000513-R	544 (1)	
ARRAY_MSG2	0000031E-R	527 (1)	
ARRAY_MSG20	00000534-R	545 (1)	
ARRAY_MSG21	00000553-R	546 (1)	
ARRAY_MSG22	0000056E-R	547 (1)	
ARRAY_MSG23	000005A1-R	548 (1)	
ARRAY_MSG24	000005D4-R	549 (1)	
ARRAY_MSG26	00000607-R	550 (1)	
ARRAY_MSG27	0000062E-R	551 (1)	
ARRAY_MSG28	00000653-R	552 (1)	
ARRAY_MSG29	00000686-R	553 (1)	
ARRAY_MSG3	00000345-R	528 (1)	
ARRAY_MSG30	000006B9-R	554 (1)	
ARRAY_MSG31	000006DE-R	555 (1)	

```
ARRAY_MSG32                     000006FD-R   556    (1)
ARRAY_MSG33                     0000071C-R   557    (1)
ARRAY_MSG34                     0000073B-R   558    (1)
ARRAY_MSG35                     0000075A-R   559    (1)
ARRAY_MSG36                     00000779-R   560    (1)
ARRAY_MSG37                     00000798-R   561    (1)
ARRAY_MSG38                     000007B7-R   562    (1)
ARRAY_MSG39                     000007D6-R   563    (1)
ARRAY_MSG4                      0000035A-R   529    (1)
ARRAY_MSG40                     000007F5-R   564    (1)
ARRAY_MSG41                     00000814-R   565    (1)
ARRAY_MSG42                     00000833-R   566    (1)
ARRAY_MSG43                     00000852-R   567    (1)
ARRAY_MSG44                     00000871-R   568    (1)
ARRAY_MSG45                     00000890-R   569    (1)
ARRAY_MSG46                     000008AF-R   570    (1)
ARRAY_MSG47                     000008D0-R   571    (1)
ARRAY_MSG48                     000008F1-R   572    (1)
ARRAY_MSG49                     00000916-R   573    (1)
ARRAY_MSG5                      0000036F-R   530    (1)
ARRAY_MSG50                     0000093B-R   574    (1)
ARRAY_MSG51                     00000960-R   575    (1)
ARRAY_MSG52                     0000098B-R   576    (1)
ARRAY_MSG53                     000009B6-R   577    (1)
ARRAY_MSG54                     000009E1-R   578    (1)
ARRAY_MSG55                     00000A03-R   579    (1)
ARRAY_MSG56                     00000A25-R   580    (1)
ARRAY_MSG57                     00000A47-R   581    (1)
ARRAY_MSG58                     00000A63-R   582    (1)
ARRAY_MSG59                     00000A7F-R   583    (1)
ARRAY_MSG6                      00000384-R   531    (1)
ARRAY_MSG60                     00000A9B-R   584    (1)
ARRAY_MSG61                     00000AB7-R   585    (1)
ARRAY_MSG62                     00000AD3-R   586    (1)
ARRAY_MSG7                      00000399-R   532    (1)
ARRAY_MSG8                      000003B1-R   533    (1)
ARRAY_MSG9                      000003C9-R   534    (1)
ASCII_MBE                       0000197F-R   729    (1)    #-1281   (2)    #-1356   (2)
ASR_OFFSET                     =00000410     178    (1)
ATA                            =00008000     302    (1)
ATTENTION                      =00010000     211    (1)    #-1549   (4)    #-1551   (4)
ATTN                           =00000018     277    (1)
A_HEADEND                       00000058-R   78     (1)      78     (1)
BCR                            =00000010     154    (1)
BIT...                         =00660130     86     (1)      76     (1)      86     (1)
BIT0                           =00000001     131    (1)     233     (1)     322     (1)     327     (1)
BIT1                           =00000002     131    (1)     224     (1)     232     (1)     310     (1)     328     (1)
BIT10                          =00000400     131    (1)     216     (1)
BIT11                          =00000800     131    (1)     215     (1)     297     (1)     332     (1)
BIT12                          =00001000     131    (1)     214     (1)     305     (1)     316     (1)
BIT13                          =00002000     131    (1)     213     (1)     304     (1)     315     (1)
BIT14                          =00004000     131    (1)     212     (1)     303     (1)
BIT15                          =00008000     131    (1)     302     (1)
BIT16                          =00010000     131    (1)     211     (1)     256     (1)
BIT17                          =00020000     131    (1)     210     (1)     255     (1)
BIT18                          =00040000     131    (1)     209     (1)     254     (1)
BIT19                          =00080000     131    (1)     208     (1)     253     (1)
```

```
BIT2                    =00000004     131  (1)     231  (1)     321  (1)     329  (1)
BIT20                   =00100000     131  (1)     252  (1)
BIT21                   =00200000     131  (1)     251  (1)
BIT22                   =00400000     131  (1)     250  (1)
BIT23                   =00800000     131  (1)     207  (1)     246  (1)
BIT24                   =01000000     131  (1)     245  (1)
BIT25                   =02000000     131  (1)     244  (1)     264  (1)
BIT26                   =04000000     131  (1)     243  (1)     263  (1)
BIT27                   =08000000     131  (1)     242  (1)     262  (1)
BIT28                   =10000000     131  (1)     241  (1)     261  (1)
BIT29                   =20000000     131  (1)     206  (1)     240  (1)
BIT3                    =00000008     131  (1)     223  (1)     230  (1)     309  (1)     320  (1)
BIT30                   =40000000     131  (1)     239  (1)
BIT31                   =80000000     131  (1)     205  (1)     238  (1)     291  (1)
BIT4                    =00000010     131  (1)     222  (1)     229  (1)     308  (1)     319  (1)
BIT5                    =00000020     131  (1)     221  (1)     318  (1)
BIT6                    =00000040     131  (1)     220  (1)     317  (1)
BIT7                    =00000080     131  (1)     219  (1)     307  (1)
BIT8                    =00000100     131  (1)     218  (1)     306  (1)     331  (1)
BIT9                    =00000200     131  (1)     217  (1)
BLANK                    00001987-R   732  (1)
BLKSCSBI                =0000001C     273  (1)
BLKSND_COMD             =10000000     241  (1)
BR4                     =00000014     136  (1)
BR5                     =00000015     137  (1)
BR6                     =00000016     138  (1)
BR7                     =00000017     139  (1)
BYTE0                   =000000FF     337  (1)
BYTE1                   =0000FF00     338  (1)
BYTE2                   =00FF0000     339  (1)
BYTE3                   =FF000000     340  (1)
BYTE_COUNT_MSK          =0000FFFF     349  (1)
CAR                     =0000001C     156  (1)     #-1516  (4)
CAR_SNAP                 0000005E-R   471  (1)     #-1516  (4)
CB_HUNG                 =00800000     207  (1)
CEP_FUNCTIONAL          =00000000     76   (1)
CEP_REPAIR              =00000001     76   (1)       76    (1)
CLEAN_UP                 00000000-R   1371 (3)       78    (1)
CLEAN_UP_X               0000004F-R   1387 (3)
COMP_MSG1                00000AF5-R   592  (1)
COMP_MSG2                00000B42-R   594  (1)
CONT_PC                  00000055-R   467  (1)     #-1403  (4)
CPE                     =00000008     320  (1)
CR                      =00000004     151  (1)     #-1379  (3)     #-1442  (4)     #-1500  (4)     #-1540  (4)
                                                    #-1580  (4)     #-818   (1)     #-825   (1)     #-832   (1)
                                                    #-839   (1)     #-846   (1)     #-853   (1)     #-860   (1)
                                                    #-866   (1)     #-873   (1)     #-880   (1)     #-887   (1)
                                                    #-894   (1)     #-901   (1)     #-908   (1)     #-915   (1)
                                                    #-921   (1)
CRTED_READ_DATA         =20000000     206  (1)
CSR                     =00000000     150  (1)
DATA_IBUFFER             00000000-R   765  (1)
DATA_OBUFFER             00000200-R   766  (1)
DATA_XFER_ABRT          =00001000     214  (1)
DATA_XFER_DONE          =00002000     213  (1)
DATA_XFER_LATE          =00000800     215  (1)
DCM                     =04000000     263  (1)
```

```
DEFAULT                         =00000000     79      (1)
DEVICES                          0000000C-R   432     (1)     #-1299   (2)     #-1365   (2)
DEVICE_PTR                       000002E2-R   502     (1)     #-1329   (2)     #-1330   (2)     #-1331   (2)      1332   (2)
DEV_REG                          000002C2-R   488     (1)       78     (1)
DIF_MSG                          00001A74-R   742     (1)     #-1050   (2)     #-1058   (2)     #-1062   (2)
DISABLE_LOG                     =30000000     181     (1)
DISPATCH                         00000000-R   77      (1)       78     (1)
DMD                             =00000001     327     (1)
DOCC                            =00000100     331     (1)
DPE                             =00000010     319     (1)
DPR                             =00000100     306     (1)
DR                              =00000014     155     (1)     #-1503   (4)     #-1505   (4)      1508   (4)     #-1517   (4)
                                                              #-1518   (4)     #-1519   (4)
DRIVE0                           00000026-R   445     (1)     #-1344   (2)     #-1465   (4)     #-1578   (4)
DRIVE1                           0000002A-R   446     (1)
DRIVE2                           0000002E-R   447     (1)
DRIVE3                           00000032-R   448     (1)
DRIVE4                           00000036-R   449     (1)
DRIVE5                           0000003A-R   450     (1)
DRIVE6                           0000003E-R   451     (1)
DRIVE7                           00000042-R   452     (1)     #-1543   (4)
DRIVES_PRESENT                   00000000-R   1576    (4)
DRIVE_ADR_TBL                    00000026-R   444     (1)     #-1340   (2)
DRIVE_INUSE                      00000046-R   454     (1)     #-1344   (2)     #-1501   (4)
DRIVE_OFFSET                    =00000080     177     (1)     #-1341   (2)     #-1590   (4)
DRIVE_SEL_MSK                   =0000E000     354     (1)
DRV_ERRMASK                     =0000E000     391     (1)
DRV_INITMASK                    =00001180     392     (1)
DRY                             =00000080     307     (1)
DS$ABORT                         00010020                       1298   (2)      1437   (4)
DS$BREAK                         00010058                       1368   (2)      1387   (3)       924   (1)
DS$CANWAIT                       00010070                       1440   (4)
DS$CLRVEC                        00010168                       1386   (3)
DS$CVTREG                        000100B0                       1029   (2)      1042   (2)      1055   (2)      1130   (2)
DS$ENDPASS                       00010010                       1310   (2)
DS$ERRSYS                        000100C0                       1434   (4)
DS$GPHARD                        00010018                       1277   (2)      1314   (2)      1354   (2)      1376   (3)
DS$K_ERROR                      =00000002     86      (1)
DS$K_NORMAL                     =00000001     86      (1)
DS$K_SEVEPE                     =00000004     86      (1)
DS$K_SUBSYS                     =00000066     86      (1)       86     (1)
DS$K_WARNING                    =00000000     86      (1)
DS$PRINTB                        000100E0                       1000   (2)      1008   (2)      1011   (2)      1016   (2)
                                                               1088   (2)      1112   (2)      1132   (2)
DS$PRINTF                        000100F0                       1297   (2)      1335   (2)
DS$PRINTX                        000100E8                       1024   (2)      1032   (2)      1037   (2)      1045   (2)
                                                               1050   (2)      1058   (2)      1062   (2)      1090   (2)
                                                               1147   (2)      1161   (2)      1163   (2)      1191   (2)
                                                               1193   (2)      1198   (2)      1200   (2)      1202   (2)
                                                               1207   (2)      1209   (2)      1211   (2)      1213   (2)
                                                               1215   (2)      1217   (2)      1222   (2)      1224   (2)
                                                               1240   (2)      1242   (2)      1258   (2)      1260   (2)
                                                               1436   (4)
DS$SETVEC                        00010160                       1345   (2)
DS$WAITUS                        00010068                       1504   (4)
DS$_ARITH                       =006600D0     86      (1)
DS$_ASBE                        =00660118     86      (1)
```

```
DS$_BADLINK                     =006600F0      86      (1)
DS$_BADTYPE                     =006600E8      86      (1)
DS$_BIIC                        =00660120      86      (1)
DS$_CHME                        =006600A8      86      (1)
DS$_CHMK                        =006600E0      86      (1)
DS$_DEVNAME                     =00660108      86      (1)
DS$_ERROR                       =00660002      86      (1)
DS$_FHWE                        =00660068      86      (1)
DS$_FRAGBUF                     =00660080      86      (1)
DS$_ICBUSY                      =006600C8      86      (1)
DS$_ICERR                       =006600C0      86      (1)
DS$_IHWE                        =00660060      86      (1)
DS$_ILLCHAR                     =00660018      86      (1)
DS$_ILLPAGCNT                   =00660078      86      (1)
DS$_ILLUNIT                     =00660100      86      (1)
DS$_INSFMEM                     =00660050      86      (1)
DS$_IPL2HI                      =006600B8      86      (1)
DS$_IVADDR                      =00660040      86      (1)
DS$_IVVECT                      =00660038      86      (1)
DS$_KRNLSTK                     =00660090      86      (1)
DS$_LOGIC                       =00660070      86      (1)
DS$_MCHK                        =00660088      86      (1)
DS$_MMOFF                       =00660058      86      (1)
DS$_NEEDUNIT                    =006600F8      86      (1)
DS$_NODE                        =00660128      86      (1)
DS$_NOPCS                       =00660110      86      (1)
DS$_NORMAL                      =00660001      86      (1)
DS$_NOSUPPORT                   =006600B1      86      (1)
DS$_NOTDON                      =00660030      86      (1)
DS$_NOTIMP                      =006600B0      86      (1)     86      (1)
DS$_NULLSTR                     =00660010      86      (1)
DS$_OVERFLOW                    =00660008      86      (1)
DS$_POWER                       =00660098      86      (1)
DS$_PROGERR                     =00660020      86      (1)
DS$_SEVERE                      =00660004      86      (1)
DS$_TRANSL                      =006600A0      86      (1)
DS$_TRUNCATE                    =00660028      86      (1)
DS$_UNEXPINT                    =006600D8      86      (1)
DS$_VASFULL                     =00660048      86      (1)
DS$_WARNING                     =00660000      86      (1)       86      (1)
DSA$GL_FLAGS                     0000FE00                        1271    (2)      1304    (2)    #-1325    (2)
DSA$GL_PASSNO                    0000FE54                      #-1323    (2)
DSA$M_TRACE                     =00000400                      #-1325    (2)
DSA$V_PASSO                     =0000001D                      #-1271    (2)    #-1304    (2)
DS_MSK                          =0FFFF8FF      401     (1)
DTE                             =00001000      316     (1)
DT_ABORT                        =00000002      232     (1)
DT_BUSY                         =80000000      205     (1)
DVA                             =00000800      297     (1)
EBL                             =0000001A      275     (1)
EDM                             =02000000      264     (1)
ENABLE_PS                       =00000800      332     (1)
ENV$M_DOMAIN                    =00000002      76      (1)
ENV$M_LEVEL                     =00000001      76      (1)
ENV$M_SUPER                     =000003FC      76      (1)
ENV$S_DOMAIN                    =00000001      76      (1)
ENV$S_LEVEL                     =00000001      76      (1)
```

```
ENV$S_SUPER                     =00000008     76     (1)
ENV$V_DOMAIN                    =00000001     76     (1)       76     (1)
ENV$V_LEVEL                     =00000000     76     (1)       76     (1)
ENV$V_SUPER                     =00000002     76     (1)       78     (1)
ENV$_CPU                        =00000000     76     (1)       76     (1)
ENV$_FUNCTIONAL                 =00000000     76     (1)       76     (1)
ENV$_REPAIR                     =00000001     76     (1)       76     (1)
ENV$_SUPER                      =00000001     76     (1)       78     (1)
ENV$_SYSTEM                     =00000001     76     (1)       76     (1)
ERCE                            =10000000     261    (1)
ERR                             =00004000     303    (1)
ERR$_MSGADR                     =0000000C     87     (1)
ERR$_NARGS                      =0000000A     87     (1)
ERR$_NUM                        =00000004     87     (1)
ERR$_P1                         =00000014     87     (1)    #-1158  (2)  #-1188  (2)  #-1195  (2)  #-1204  (2)
                                                             #-1219  (2)  #-1237  (2)  #-1255  (2)
ERR$_P2                         =00000018     87     (1)    #-1159  (2)  #-1189  (2)  #-1196  (2)  #-1205  (2)
                                                             #-1220  (2)  #-1238  (2)  #-1256  (2)
ERR$_P3                         =0000001C     87     (1)
ERR$_P4                         =00000020     87     (1)
ERR$_P5                         =00000024     87     (1)
ERR$_P6                         =00000028     87     (1)
ERR$_POINTER                    =00000010     87     (1)
ERR$_UNIT                       =00000008     87     (1)
ERR_STAT                        =00000008     223    (1)
EXCEPTION                       =00000015     279    (1)
EXISTING_DRIVE                   0000004A-R   455    (1)
EXP_INTERRUPT                    00000052-R   463    (1)      1433  (4)  #-816  (1)  #-823  (1)  #-830  (1)
                                                             #-837  (1)  #-844  (1)  #-851  (1)  #-858  (1)
                                                             #-864  (1)  #-871  (1)  #-878  (1)  #-885  (1)
                                                             #-892  (1)  #-899  (1)  #-906  (1)  #-913  (1)
                                                             #-919  (1)
EXP_MSG                          00001A68-R   741    (1)    #-1024  (2)  #-1032  (2)  #-1062  (2)
EXT_REG_MSG                      00001985-R   731    (1)
EXT_REG_OFFSET                  =00000400     175    (1)    #-1339  (2)
FAIL                            =00000013     280    (1)
FERR                            =00000004     329    (1)
FMT_ALLDRVPRES                   000012D2-R   663    (1)
FMT_ANY_REG                      00000DC5-R   618    (1)
FMT_BADNEXUS                     00001301-R   666    (1)
FMT_BODY                         000014C5-R   681    (1)      1024  (2)      1032  (2)      1037  (2)      1045  (2)
                                                              1050  (2)      1058  (2)
FMT_BYTMSK_ER                    000017B8-R   711    (1)
FMT_CMICMD_ERR                   000017A4-R   708    (1)      1161  (2)
FMT_COMP_ERR                     000014D9-R   683    (1)      1062  (2)
FMT_CONTRL_REG                   00000B8F-R   603    (1)
FMT_DCMPERR                      00001709-R   703    (1)      1112  (2)
FMT_DIAG_REG                     00000D09-R   614    (1)
FMT_DRIVENORES                   000012BC-R   661    (1)
FMT_DRIVERES                     000012A5-R   660    (1)
FMT_DTERR                        000016A9-R   700    (1)
FMT_DUADR_ERR                    0000144C-R   677    (1)
FMT_DUALRS_ERR                   00001427-R   675    (1)
FMT_DUALRS_MSG                   000019FC-R   736    (1)
FMT_DUALWT_ERR                   00001407-R   673    (1)
FMT_DUALWT_MSG                   000019A5-R   734    (1)
FMT_ERRSUM                       00001782-R   707    (1)
```

```
FMT_EXCP_ERR                    000015A6-R   690    (1)
FMT_EXTREG_ERR                  00001654-R   696    (1)       1016   (2)
FMT_FATAL_MMM                   00001231-R   652    (1)
FMT_HEADER                      00001494-R   679    (1)       1011   (2)
FMT_IBCER1                      000017D0-R   714    (1)       1191   (2)
FMT_IBCER2                      000017E0-R   715    (1)       1198   (2)
FMT_IBCER3                      000017F0-R   716    (1)       1202   (2)
FMT_IBCER4                      00001809-R   717    (1)       1207   (2)
FMT_IBCER5                      00001819-R   718    (1)       1211   (2)
FMT_IBCER6                      0000184A-R   719    (1)       1213   (2)
FMT_IBCER7                      00001869-R   720    (1)       1215   (2)
FMT_IBCER8                      000018C3-R   722    (1)       1217   (2)
FMT_IBCER9                      000018E2-R   723    (1)       1222   (2)
FMT_IBUS0                       0000127C-R   656    (1)
FMT_IBUS1                       00001253-R   657    (1)
FMT_INIT1                       00001216-R   648    (1)       1335   (2)
FMT_INTFAILED                   00001B7F-R   759    (1)       1132   (2)
FMT_LOCATION                    0C0011F7-R   647    (1)
FMT_MAPFUN_ERR                  000015D7-R   691    (1)       1088   (2)
FMT_MAP_EXERR                   000015EE-R   693    (1)       1090   (2)
FMT_MAP_HEAD                    000014AA-R   680    (1)       1000   (2)
FMT_MAP_REG                     00000E7B-R   622    (1)
FMT_MBEREG_ERR                  C000167E-R   698    (1)       1008   (2)
FMT_MBE_ASR                     00001137-R   639    (1)
FMT_MBE_CR1                     00000F30-R   626    (1)
FMT_MBE_ER                      00001086-R   634    (1)
FMT_MBE_SR                      00000FD8-R   630    (1)
FMT_MISSEDXFER                  00001564-R   687    (1)
FMT_NOINTEST                    00001BB9-R   761    (1)
FMT_NOTHING                     000014D0-R   682    (1)     #-1024   (2)     #-1037   (2)     #-1050   (2)
FMT_NO_MBE                      0000193A-R   727    (1)
FMT_NO_MBE_SEL                  0000195C-R   728    (1)
FMT_NO_RH750                    00001B31-R   753    (1)       1297   (2)
FMT_NX_ER                       00001AB6-R   750    (1)       1147   (2)
FMT_PTR                         000002BA-R   485    (1)
FMT_RHS                         000011DF-R   646    (1)
FMT_SERR                        00001B0C-R   752    (1)       1258   (2)
FMT_SILOERROR                   000014F7-R   684    (1)
FMT_STATUS_REG                  00000C42-R   608    (1)       1130   (2)
FMT_UNEXP_INT                   00001B50-R   758    (1)       1436   (4)
FMT_VAL                         0000190F-R   725    (1)       1163   (2)      1193   (2)      1200   (2)      1209   (2)
                                                              1224   (2)      1242   (2)      1260   (2)
FMT_VARER                       00001399-R   670    (1)
FMT_VARINCERR                   000013E3-R   672    (1)
FMT_VECERR                      00001AE3-R   751    (1)       1240   (2)
FMT_XFERERR                     00001348-R   668    (1)
HP$A_DEVICE                     00000018                     #-1316   (2)     #-1362   (2)     #-1378   (3)
HP$A_LINK                       00000020                     #-1359   (2)
HP$Q_DEVICE                     00000000                     #-1328   (2)
HP$T_TYPE                       00000026                     #-1280   (2)     #-1356   (2)
HP$W_VECTOR                     00000024                     #-1318   (2)
IBC_MSG                          000002DA-R   497    (1)     #-1177   (2)
ILF                            =00000001     322    (1)
IMAPP                          =0000001D     272    (1)
IMBCP                          =0000001E     271    (1)
IMBDP                          =0000001F     270    (1)
INDEX_MSG                       00001AAE-R   749    (1)
```

```
INH_BYTE_CNT                    =00000010      229    (1)
INITIALIZE                       00000000-R    1265   (2)       78     (1)
INITIALIZE_X                     000001E5-R    1368   (2)
INTERRUPT_EN                    =00000004      231    (1)     #-1442   (4)     #-818    (1)     #-825    (1)     #-832    (1)
                                                              #-839    (1)     #-846    (1)     #-853    (1)     #-860    (1)
                                                              #-866    (1)     #-873    (1)     #-880    (1)     #-887    (1)
                                                              #-894    (1)     #-901    (1)     #-908    (1)     #-915    (1)
                                                              #-921    (1)
INT_OCCURRED                     00000053-R    464    (1)     #-1438   (4)     #-817    (1)     #-824    (1)     #-831    (1)
                                                              #-838    (1)     #-845    (1)     #-852    (1)     #-859    (1)
                                                              #-865    (1)     #-872    (1)     #-879    (1)     #-886    (1)
                                                              #-893    (1)     #-900    (1)     #-907    (1)     #-914    (1)
                                                              #-920    (1)
INVRT_MAP_PAR                   =20000000      240    (1)
INVRT_MB_CPAR                   =40000000      239    (1)
INVRT_MB_DPAR                   =80000000      238    (1)
INVRT_SILO_PAR                  =00400000      250    (1)
IPLR                            =00000012      144    (1)
ISP                             =00000016      278    (1)
ISR0                             00000000-R    815    (1)       793    (1)
ISR1                             00000000-R    822    (1)       794    (1)
ISR10                            00000000-R    884    (1)       803    (1)
ISR11                            00000000-R    891    (1)       804    (1)
ISR12                            00000000-R    898    (1)       805    (1)
ISR13                            00000000-R    905    (1)       806    (1)
ISR14                            00000000-R    912    (1)       807    (1)
ISR15                            00000000-R    918    (1)       808    (1)
ISR2                             00000000-R    829    (1)       795    (1)
ISR3                             00000000-R    836    (1)       796    (1)
ISR4                             00000000-R    843    (1)       797    (1)
ISR5                             00000000-R    850    (1)       798    (1)
ISR6                             00000000-R    857    (1)       799    (1)
ISR7                             00000000-R    863    (1)       800    (1)
ISR8                             00000000-R    870    (1)       801    (1)
ISR9                             00000000-R    877    (1)       802    (1)
ISR_TBL                          00001C3E-R    793    (1)
ISSUE_MMREAD                     00000000-R    1447   (4)
ISSUE_MMREADX                    0000001F-R    1469   (4)
L$A_CCP                          00000040-R    78     (1)
L$A_DEVP                         0000001C-R    78     (1)
L$A_DREG                         00000024-R    78     (1)
L$A_DTP                          00000018-R    78     (1)
L$A_ICP                          0000003C-R    78     (1)
L$A_LASTAD                       00000014-R    78     (1)
L$A_NAME                         00000008-R    78     (1)
L$A_REPP                         00000044-R    78     (1)
L$A_SECNAM                       00000050-R    78     (1)
L$A_STATAB                       00000048-R    78     (1)
L$A_TSTCNT                       00000054-R    78     (1)
L$L_ENVIRON                      00000004-R    78     (1)
L$L_ERRTYP                       0000004C-R    78     (1)
L$L_HEADLENGTH                   00000000-R    78     (1)
L$L_REV                          0000000C-R    78     (1)
L$L_UNIT                         00000020-R    78     (1)
L$L_UPDATE                       00000010-R    78     (1)
LASTAD                           00000000-R    78     (1)       78     (1)
LN                               00000008-R    431    (1)     #-1273   (2)     #-1277   (2)     #-1287   (2)     #-1290   (2)
```

```
                                                        #-1299   (2)    #-1352   (2)    #-1354   (2)    #-1365   (2)
LOWBITS_MSK                     =0000003F     394   (1)
LUN                              0000001C-R   436   (1)    #-1303   (2)    #-1305   (2)    #-1307   (2)    #-1309   (2)
                                                        #-1312   (2)
L_TSTCNT                         00000000-R   78    (1)      78     (1)
MACH_CHK_SRV                     00000000-R   1392  (4)
MAINT_MODE                      =00000008     230   (1)    #-1466   (4)    #-1500   (4)
MAP_1_PATTRN                    =80007FFF     358   (1)
MAP_INVALID                     =00000010     222   (1)
MAP_OFFSET                      =00000800     179   (1)
MAP_PE                          =00000020     221   (1)
MAP_PTR_MSK                     =0001FE00     352   (1)
MASS_CNTRL_PE                   =00020000     210   (1)
MASS_CTOD                       =00010000     256   (1)
MASS_DATA_PE                    =00000040     220   (1)
MASS_ECP                        =00000080     219   (1)
MASS_EXCP                       =00020000     255   (1)
MASS_FAIL                       =00100000     252   (1)
MASS_RUN                        =00080000     253   (1)
MASS_WCLK                       =00040000     254   (1)
MBA_LN                           00000014-R   434   (1)     1270    (2)      1373    (3)
MBDIB_SEL                       =00800000     246   (1)
MBE                             =00000002     80    (1)
MBE_ASR                         =00000010     167   (1)
MBE_CR1                         =00000000     163   (1)
MBE_CR2                         =00000014     168   (1)
MBE_CUR_ADR                      00000004-R   429   (1)    #-1349   (2)    #-1363   (2)    #-1541   (4)
MBE_DBR                         =0000001C     170   (1)
MBE_DTR                         =00000018     169   (1)    #-1545   (4)    #-1587   (4)
MBE_ER                          =00000008     165   (1)
MBE_MR                          =0000000C     166   (1)
MBE_PARAM                       =00000087     397   (1)
MBE_REG_MSG                      00001983-R   730   (1)
MBE_SR                          =00000004     164   (1)
MCLK                            =00000002     328   (1)
MEM_CSR0                        =40F20000     420   (1)
MEM_CSR1                        =40F20004     415   (1)
MIOBUFFER                        00000400-R   767   (1)
MISSED_XFER                     =00000100     218   (1)
MM_XFER                          00000000-R   1473  (4)
MM_XFERX                         0000007A-R   1521  (4)    #-1494   (4)
MOL                             =00001000     305   (1)
NIBBLE0                         =0000000F     341   (1)
NIBBLE1                         =000000F0     342   (1)
NIBBLE2                         =00000F00     343   (1)
NIBBLE3                         =0000F000     344   (1)
NIBBLE4                         =000F0000     345   (1)
NIBBLE5                         =00F00000     346   (1)
NIBBLE6                         =0F000000     347   (1)
NIBBLE7                         =F0000000     348   (1)
NODRIVE                          0000004E-R   458   (1)
NON_XIST_DRIVE                  =00040000     209   (1)
NOOP                            =00000000     186   (1)
NO_MBE                           00000021-R   438   (1)    #-1275   (2)    #-1283   (2)    #-1350   (2)
NO_MBE_MSG                       00001989-R   733   (1)
NO_RESPONSE                     =00000002     224   (1)
NO_RH750                         00000020-R   437   (1)    #-1274   (2)    #-1286   (2)    #-1288   (2)    #-1295   (2)
```

```
                                                        #-1307   (2)
NUMBER_BUFFER                 000002AA-R   475    (1)
NXM_ADR                       000002D6-R   495    (1)   #-1147   (2)
NXM_RD_FLAG                   00000054-R   465    (1)   #-1401   (4)
OCC                          =00000019     276    (1)
OPI                          =00002000     315    (1)
PAGE_BYTE_MSK                =000001FF     350    (1)
PF_FIELD                     =00000009     408    (1)
PF_WIDTH                     =00000015     410    (1)
PGM_INIT                     =00000001     233    (1)   #-1379   (3)   #-1540   (4)   #-1580   (4)
PIP                          =00002000     304    (1)
PM                           =08000000     262    (1)
PPFN_MASK                    =00007FFF     292    (1)
PRINT_CMDER                   00000000-R   1152   (2)
PRINT_CMDERX                  0000002E-R   1164   (2)
PRINT_DTERR                   00000000-R   1096   (2)
PRINT_DTERRX                  0000001F-R   1113   (2)
PRINT_IBCERR                  00000000-R   1170   (2)
PRINT_IBCERRX                 00000125-R   1225   (2)   #-1194   (2)   #-1201   (2)   #-1203   (2)   #-1210   (2)
                                                        #-1212   (2)   #-1214   (2)   #-1216   (2)   #-1218   (2)
PRINT_INTERR                  00000000-R   1118   (2)
PRINT_INTERRX                 00000000-R   1135   (2)
PRINT_MAPERR                  00000000-R   1068   (2)
PRINT_MAPERRX                 00000024-R   1091   (2)
PRINT_NULL                    00000000-R   964    (2)
PRINT_NULL_X                  00000002-R   966    (2)
PRINT_NXADR                   00000000-R   1140   (2)
PRINT_NXADRX                  00000015-R   1148   (2)
PRINT_SBE                     00000000-R   970    (2)
PRINT_SBEX                    000001CD-R   1063   (2)
PRINT_SILOER                  00000000-R   1248   (2)
PRINT_SILOERX                 0000002E-R   1261   (2)
PRINT_VECTERR                 00000000-R   1230   (2)
PRINT_VECTERRX                0000002E-R   1243   (2)
PROGRAM_ERROR                =00080000     208    (1)
PTBASE                        00000022-R   439    (1)     1277   (2)   #-1279   (2)     1314   (2)   #-1315   (2)
                                                          1354   (2)   #-1355   (2)     1376   (3)   #-1377   (3)
RANDOM                       =00008259     400    (1)
RCPA                         =00000008     309    (1)
RDPA                         =00000010     308    (1)
READ                         =00000039     194    (1)   #-1467   (4)
READ_HEADER                  =0000003B     195    (1)
READ_REV                     =0000003F     196    (1)
REC_MSG                       00001A5C-R   740    (1)   #-1037   (2)   #-1045   (2)   #-1062   (2)
REG_NAME                      000002AE-R   482    (1)   #-1000   (2)   #-1011   (2)
REG_NO                        000002B2-R   483    (1)   #-1002   (2)   #-1004   (2)   #-997    (2)
REG_SEL_MSK                  =00001F00     356    (1)
REG_STRING                    000002B6-R   484    (1)     1029   (2)     1042   (2)     1055   (2)
REPORT_BUFFER                 000000AA-R   474    (1)     1029   (2)   #-1032   (2)     1042   (2)   #-1045   (2)
                                                          1055   (2)   #-1058   (2)     1130   (2)   #-1132   (2)
RH11TB_MSK                    000002DE-R   499    (1)   #-1549   (4)   #-1551   (4)
RH11TB_PRES                   00000000-R   1526   (4)
RH750                        =00000001     80     (1)
RH750_LINK                    00000010-R   433    (1)   #-1319   (2)   #-1359   (2)
RHBCR_MSG                     00001A96-R   746    (1)
RHCR_MSG                      00001A80-R   743    (1)
RHDR_MSG                      00001A9E-R   747    (1)
```

```
RHMAPR_MSG                       00001AA5-R   748    (1)
RHSR_CHECK                       =000F5FFA    398    (1)
RHSR_MSG                         00001A87-R   744    (1)
RHVAR_MSG                        00001A8E-R   745    (1)
RH_CUR_ADR                       00000000-R   428    (1)    #-1317   (2)    #-1335   (2)    #-1338   (2)    #-1441   (4)
                                                             #-1464   (4)    #-1539   (4)
RH_ISR                           00000000-R   1407   (4)      1345   (2)
RH_ISRX                          0000003B-R   1439   (4)
RH_VECS                          00001BFA-R   770    (1)
RH_VECTOR                        00000018-R   435    (1)    #-1318   (2)    #-1345   (2)    #-1386   (3)
RMR                              =00000004    321    (1)
RRUN                             =00000002    310    (1)
RS_MSK                           =0FFF07FF    404    (1)
R_FAIL                           =00000040    317    (1)
R_MB_EXC                         =00000020    318    (1)
SA0_MSG                          00001A39-R   737    (1)
SA1_MSG                          00001A44-R   738    (1)
SCLK                             =0000001B    274    (1)
SEARCH                           =00000019    188    (1)
SECTION                          00000010-R   79     (1)      78     (1)
SEEK                             =00000002    187    (1)
SEP_FUNCTIONAL                   =00000002    76     (1)
SEP_REPAIR                       =00000003    76     (1)
SILO_PE                          =00004000    212    (1)
SIM_ATTN                         =01000000    245    (1)
SIM_EBL                          =04000000    243    (1)    #-1517   (4)    #-1518   (4)
SIM_EXCEPT                       =00200000    251    (1)
SIM_OCC                          =02000000    244    (1)    #-1468   (4)    #-1503   (4)    #-1519   (4)
SIM_SCLK                         =08000000    242    (1)    #-1507   (4)
SIZ...                           =00000001    76     (1)      76     (1)
SR                               =00000008    152    (1)    #-1499   (4)    #-1582   (4)
SUFFIX_PTR                       000002BE-R   486    (1)
SUMMARY                          00000000-R   923    (1)      78     (1)
SUMMARY_X                        00000002-R   924    (1)
S_ALL                            00000008-R   79     (1)      79     (1)
S_DEFAULT                        00000000-R   79     (1)      79     (1)
S_MBE                            0000000C-R   79     (1)      79     (1)
TEMP                             000002C6-R   490    (1)
TEMP1                            000002CA-R   491    (1)
TEMP2                            000002CE-R   492    (1)
TEMP3                            000002D2-R   493    (1)
TEST_VEC                         00001C3A-R   787    (1)    #-815    (1)    #-822    (1)    #-829    (1)    #-836    (1)
                                                             #-843    (1)    #-850    (1)    #-857    (1)    #-863    (1)
                                                             #-870    (1)    #-877    (1)    #-884    (1)    #-891    (1)
                                                             #-898    (1)    #-905    (1)    #-912    (1)    #-918    (1)
TYPE_MBE                         =00000020    393    (1)    #-1544   (4)    #-1588   (4)
T_MBE                            00000026-R   80     (1)      80     (1)
T_NAME                           00000058-R   78     (1)      78     (1)
T_RH750                          00000020-R   80     (1)      80     (1)
UNEXPECTED                       00001A4F-R   739    (1)
VALID_BIT                        =80000000    291    (1)
VAR                              =0000000C    153    (1)
VAR_PAT1                         =00000778    363    (1)
VAR_PAT10                        =00000800    372    (1)
VAR_PAT11                        =00006F80    373    (1)
VAR_PAT12                        =00001000    374    (1)
VAR_PAT13                        =00005F80    375    (1)
```

```
VAR_PAT14                       =00002000     376    (1)
VAR_PAT15                       =00003F80     377    (1)
VAR_PAT16                       =00004000     378    (1)
VAR_PAT17                       =00017800     379    (1)
VAR_PAT18                       =00008000     380    (1)
VAR_PAT19                       =0000F800     381    (1)
VAR_PAT2                        =00000080     364    (1)
VAR_PAT20                       =00010000     382    (1)
VAR_PAT21                       =00017FF8     383    (1)
VAR_PAT22                       =00008000     384    (1)
VAR_PAT23                       =0000FFF8     385    (1)
VAR_PAT24                       =00010000     386    (1)
VAR_PAT3                        =000006F8     365    (1)
VAR_PAT4                        =00000100     366    (1)
VAR_PAT5                        =000005F8     367    (1)
VAR_PAT6                        =00000200     368    (1)
VAR_PAT7                        =000003F8     369    (1)
VAR_PAT8                        =00000400     370    (1)
VAR_PAT9                        =00007780     371    (1)
WAIT_TIME                        0000005A-R   470    (1)
WCEL                            =00000009     286    (1)
WCEU                            =0000000A     285    (1)
WRITE                           =00000031     192    (1)
WRITE_CHECK                     =00000029     189    (1)
WRITE_CHECKHDR                  =0000002B     190    (1)
WRITE_CHECKREV                  =0000002F     191    (1)
WRITE_CHK_HIGH                  =00000400     216    (1)
WRITE_CHK_LOW                   =00000200     217    (1)
WRITE_HEADER                    =00000033     193    (1)
```

Macros Cross Reference

MACRO	SIZE	DEFINITION	REFERENCES...
$D1_ABPROGRAM	1	1298 (2)	1298 (2) 1437 (4)
$D1_ERR_S	1	1434 (4)	1434 (4)
$D1_PRINT_S	1	1000 (2)	1000 (2) 1008 (2) 1011 (2) 1016 (2) 1024 (2)
			1032 (2) 1037 (2) 1045 (2) 1050 (2) 1058 (2)
			1062 (2) 1088 (2) 1090 (2) 1112 (2) 1132 (2)
			1147 (2) 1161 (2) 1163 (2) 1191 (2) 1193 (2)
			1198 (2) 1200 (2) 1202 (2) 1207 (2) 1209 (2)
			1211 (2) 1213 (2) 1215 (2) 1217 (2) 1222 (2)
			1224 (2) 1240 (2) 1242 (2) 1258 (2) 1260 (2)
			1297 (2) 1335 (2) 1436 (4)
$D1_SEC	1	79 (1)	79 (1)
$DEF	1	131 (1)	
$DEFINI	1	82 (1)	82 (1) 83 (1) 84 (1) 85 (1)
$DS_ABORT	1	1298 (2)	1298 (2) 1437 (4)
$DS_BERROR	1	88 (1)	1278 (2)
$DS_BGNCLEAN	1	1371 (3)	1371 (3)
$DS_BGNINIT	1	1265 (2)	1265 (2)
$DS_BGNMOD	1	76 (1)	76 (1)
$DS_BGNSUMMARY	1	923 (1)	923 (1)
$DS_BITDEF	2	131 (1)	131 (1)
$DS_BNERROR	1	91 (1)	
$DS_BNPASS0	1	1304 (2)	1304 (2)
$DS_BPASS0	1	1271 (2)	1271 (2)
$DS_BREAK	1	924 (1)	1368 (2) 1387 (3) 924 (1)
$DS_CANWAIT_G	1	1440 (4)	1440 (4)
$DS_CLRVEC_S	1	1386 (3)	1386 (3)
$DS_CVTREG_S	1	1026 (2)	1026 (2) 1039 (2) 1052 (2) 1128 (2)
$DS_DEVTYP	1	80 (1)	80 (1)
$DS_DISPATCH	1	77 (1)	77 (1)
$DS_DSADEF	5	84 (1)	84 (1)
$DS_DSDEF	2	86 (1)	86 (1)
$DS_DSSDEF	8	85 (1)	85 (1)
$DS_ENDCLEAN	1	1387 (3)	1387 (3)
$DS_ENDINIT	1	1368 (2)	1368 (2)
$DS_ENDMOD	1	1598 (4)	1598 (4)
$DS_ENDPASS_S	1	1310 (2)	1310 (2)
$DS_ENDSUMMARY	1	924 (1)	924 (1)
$DS_ENVDEF	2	76 (1)	76 (1)
$DS_ERRDEF	1	87 (1)	87 (1)
$DS_ERRSYS_S	1	1434 (4)	1434 (4)
$DS_GPHARD_S	1	1277 (2)	1277 (2) 1314 (2) 1354 (2) 1376 (3)
$DS_HEADER	4	78 (1)	78 (1)
$DS_HPODEF	2	82 (1)	82 (1)
$DS_PRINTB_S	1	999 (2)	1008 (2) 1010 (2) 1016 (2) 1088 (2) 1111 (2)
			1131 (2) 999 (2)
$DS_PRINTF_S	1	1297 (2)	1297 (2) 1333 (2)
$DS_PRINTX_S	1	1022 (2)	1022 (2) 1030 (2) 1035 (2) 1043 (2) 1048 (2)
			1056 (2) 1059 (2) 1089 (2) 1147 (2) 1161 (2)
			1162 (2) 1191 (2) 1192 (2) 1198 (2) 1199 (2)
			1202 (2) 1207 (2) 1208 (2) 1211 (2) 1213 (2)

```
                                                1215   (2)      1217   (2)      1222   (2)      1223   (2)      1240   (2)
                                                1241   (2)      1258   (2)      1259   (2)      1435   (4)
$DS_RH750_DEF      1          83     (1)        83     (1)
$DS_SECTION        2          79     (1)        79     (1)
$DS_SETVEC_S       1          1345   (2)        1345   (2)
$DS_WAITUS_S       1          1504   (4)        1504   (4)
$EQU               1          131    (1)        131    (1)      76     (1)      86     (1)
$EQULS1            1          86     (1)        76     (1)      86     (1)
$EQULST            1          76     (1)        76     (1)      86     (1)
$GBLINI            2          76     (1)        131    (1)      76     (1)      86     (1)
$OFFDEF            1          87     (1)        87     (1)
$PUSHADR           1          1029   (2)        1029   (2)      1042   (2)      1055   (2)      1130   (2)      1434   (4)
                                                1504   (4)
$VIELD             1          76     (1)        76     (1)
$VIELD1            1          131    (1)        76     (1)
ERRPREP            2          96     (1)

                                        +-------------------------+
                                        ! Performance indicators !
                                        +-------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                    38    00:00:00.10    00:00:00.45
Command processing               122    00:00:00.29    00:00:00.70
Pass 1                           867    00:00:12.23    00:00:16.12
Symbol table sort                  0    00:00:00.71    00:00:00.72
Pass 2                           416    00:00:04.51    00:00:06.75
Symbol table output                3    00:00:00.33    00:00:00.50
Psect synopsis output              3    00:00:00.11    00:00:00.11
Cross-reference output            11    00:00:01.82    00:00:02.15
Assembler run totals            1461    00:00:20.13    00:00:27.52

The working set limit was 2048 pages.
238671 bytes (467 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 762 non-local and 52 local symbols.
1599 source lines were read in Pass 1, producing 142 object records in Pass 2.
101 pages of virtual memory were used to define 40 macros.

                                        +---------------------------+
                                        ! Macro library statistics !
                                        +---------------------------+

Macro library name                          Macros defined
------------------                          --------------
SYS$COMMON: SYSLIB DIAG.MLB;953                     39
SYS$COMMON: SYSLIB STARLET.MLB;2                     7
TOTALS (all libraries)                              46

730 GETS were required to define 46 macros.

There were no errors, warnings or information messages.

MACRO/CRO/LIS=ECCAA1.SEQ ECCAA1
```

Table of contents

(1) 76 DECLARATIONS
(2) 140 TEST 1: REGISTER AND INTERNAL BUS VALIDATION ROUTINE
(7) 331 TEST 2: INITIALIZATION TEST
(8) 415 TEST 3: CONTROL REGISTER SA0 TEST
(10) 498 TEST 4: DIAGNOSTIC REGISTER TEST
(14) 781 TEST 5: VIRTUAL ADDRESS REGISTER SA0 AND LOADING TEST
(16) 1052 TEST 6: MAP REGISTER SA0 TEST
(20) 1175 TEST 7: EXTERNAL REGISTER DATA PATH TEST
(21) 1248 TEST 8: CAR BUS FUNCTION TEST
(25) 1409 TEST 9: MBDIB DIAGNOSTIC REGISTER TEST
(32) 1889 TEST 10: STATUS REGISTER STATIC SA0 TEST
(45) 2348 TEST 11: MAPPING FUNCTION TEST
(47) 2458 TEST 12: STATUS REGISTER XFER BITS TEST
(61) 3054 TEST 13: BLOCK TRANSFER TEST
(63) 3182 TEST 14: BYTE COUNTER TEST
(67) 3339 TEST 15: FUNCTIONAL READ/WRITE TESTS

```
                     0000     1 .TITLE  ECCAA_2 RH750 REPAIR LEVEL MODULE 2
                     0000     2 .IDENT  /1.8/
                     0000     3
                     0000     4 ;
                     0000     5 ; COPYRIGHT (C) 1980,1983
                     0000     6 ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
                     0000     7 ;
                     0000     8 ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
                     0000     9 ; AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE
                     0000    10 ; AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
                     0000    11 ; SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR
                     0000    12 ; OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO
                     0000    13 ; AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERED.
                     0000    14 ;
                     0000    15 ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
                     0000    16 ; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
                     0000    17 ; EQUIPMENT CORPORATION.
                     0000    18 ;
                     0000    19 ; DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR  RELIABILITY
                     0000    20 ; OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
                     0000    21 ;
                     0000    22
                     0000    23 ;++
                     0000    24 ; FACILITY: STAR DIAGNOSTIC LIBRARY
                     0000    25 ;
                     0000    26 ; ABSTRACT:
                     0000    27 ;
                     0000    28 ; THIS PROGRAM VERIFIES THE MASS BUS ADAPTER FUNCTIONALITY
                     0000    29 ;
                     0000    30 ; ENVIRONMENT: DIAGNOSTIC SUPERVISOR
                     0000    31 ;
                     0000    32 ; REVISION HISTORY:
                     0000    33 ;
                     0000    34 ;       1.0     September 1980
                     0000    35 ;               Initial Release
                     0000    36 ;       1.1     January 1981 Don Monroe
                     0000    37 ;               Added a check of the status register to all subtests
                     0000    38 ;               of test 9 to make sure function completed correctly.
                     0000    39 ;               Modified 'NO DRIVE RESPONDED' message to only occur
                     0000    40 ;               on first pass.
                     0000    41 ;       1.2     Dom Andella     May 1981
                     0000    42 ;               Added two subtests to test 10 to test the DT Busy
                     0000    43 ;               One-Shot and timing components.
                     0000    44 ;       1.3     Dan Milleville  June 1981
                     0000    45 ;               Changed ASCIC messages for MDP numbers from 0 - 7 to 1 - 8,
                     0000    46 ;               and changed message printed when MBE selected but not present
                     0000    47 ;               to a hard error.
                     0000    48 ;       1.4     Dan Milleville  June 1981
                     0000    49 ;               Changed timing in Test 10 Subtest 12 to 1 MS., and Test 10
                     0000    50 ;               Subtest 13 to 300 US.  Also, with Memory Management on, Test
                     0000    51 ;               10 would fail, caused by the pound symbol (#) missing from the
                     0000    52 ;               four "$DS_SETIPL_S"'s parameter.
                     0000    53 ;       1.5     Dan Milleville  Jan 1982
                     0000    54 ;               Added Subtest 5 to Test 1 - checks that the Unit Under Test
                     0000    55 ;               (UUT) responds to only 1 address between 40F28800(X) and
                     0000    56 ;               40F2F800(X), in steps of 1000(X)
                     0000    57 ;               Added Subtest 1 to Test 6 - checks that there is no "bit cross-
```

```
                0000     58 ;                   talk" between the 100(X) Map Registers, i.e. writing to 1
                0000     59 ;                   register does not result in any other bits in any other regis-
                0000     60 ;                   ters becoming set.
                0000     61 ;           1.6     Dan Milleville  Nov 1982
                0000     62 ;                   Because of an ECO, a TSTL had to be added to T1S5 in the routine
                0000     63 ;                   that tried to clear all registers.
                0000     64 ;
                0000     65 ;
                0000     66 ;           1.7     Ricardo De Andrade May 1983
                0000     67 ;                   Fixed Test 1, used to fail with two MBA's because of a bad
                0000     68 ;                   call to the P_TABLE when the UUT was RH1 or RH2.
                0000     69 ;
                0000     70 ;
                0000     71 ;           1.8     Susan E. Hutcheson                      January 1985
                0000     72 ;                   Test 12 subtest 10 forces a double bit error but does not clean
                0000     73 ;                   up error status flags in CRS0. These bits must be cleared man-
                0000     74 ;                   ually. Search on  1.8 .
                0000     75 ;--
```

```
                        76 .SBTTL  DECLARATIONS
                        77
                        78 ;
                        79 ; LIBRARY FILES:
                        80 ;
                        81 .LIBRARY        /SYS$LIBRARY:DIAG/
                        82
                        83 $DS_SECDEF      <ALL,MBE>
                        84 ;++
                        85 ; MACROS UNIQUE TO THIS PROGRAM
                        86 ;--
                        87         .MACRO  ERRPREP, MREG_NAME,MREG_NO ,MREG_STRING ,SUFFIX = BLANK
                        88 ;++
                        89 ; THIS MACRO IS USED TO SET UP ERROR DISPLAY PARAMETERS
                        90 ; FOR THE PRINT ERROR ROUTINE
                        91 ;
                        92 ;       MREG_NAME IS A POINTER TO THE STRING DESCRIBING THE REGISTER
                        93 ;       WHICH PRECEEDS THE ERROR CONDITION
                        94 ;
                        95 ;       MREG_NO IS THE RH750 REGISTER NUMBER UNDER TEST.
                        96 ;
                        97 ;       MREG_STRING IS A POINTER TO THE STRING CONTAINING THE BIT
                        98 ;       MNEMONICS OF THE REGISTER UNDER TEST.
                        99 ;
                       100 ;       SUFFIX IS A POINTER TO THE SUFFIX APPENDED TO THE ERROR REPORT.
                       101 ;--
                       102         .IF     NB MREG_NAME
                       103                 MOVAL   MREG_NAME,REG_NAME
                       104         .ENDC
                       105         .IF     NB MREG_NO
                       106                 MOVZBL  #MREG_NO,REG_NO
                       107         .ENDC
                       108         .IF     NB MREG_STRING
                       109                 MOVAL   MREG_STRING,REG_STRING
                       110         .ENDC
                       111         .IF     DF SUFFIX
                       112                 MOVAL   SUFFIX,SUFFIX_PTR
                       113         .ENDC
                       114         .ENDM
                       115 ;++
                       116 ; THE FOLLOWING MACROS ARE USED IN THE TESTING OF RH750 INTERRUPTS
                       117 ;--
                       118 .MACRO  EXPECT_INTER
                       119                 MTPR    #20,#IPLR
                       120                 MOVB    #1,EXP_INTERRUPT
                       121                 CLRB    INT_OCCURRED
                       122                 BISL    #INTERRUPT_EN,CR(R2)
                       123 .ENDM
                       124
                       125 .MACRO  NO_INTER
                       126         CLRB    EXP_INTERRUPT
                       127         CLRB    INT_OCCURRED
                       128 .ENDM
                       129
                       130 ;
                       131 ; EQUATED SYMBOLS
                       132 ;
```

```
0000    133 $DS_BITDEF
0000    134 $DS_DSADEF
0000    135 $DS_HPODEF
0000    136 $DS_RH750_DEF
0000    137 $DS_BGNMOD      CEP_REPAIR,1
0000        ;;;     Macro-32 Diagnostic macro library Version 7.0 "DIAG.MLB (186)"
0000    138 $DS_PAGE
```

```
                        0000                    .SBTTL  TEST 1:  REGISTER AND INTERNAL BUS VALIDATION ROUTINE
                    00000000                    .PSECT  TEST_001, PAGE, NOWRT
                        0030
                        0030   141 $DS_BGNTEST      <DEFAULT,ALL>,,LONG
                        0030       DATA_001:
                    00000000    0030                .LONG   0                   ; TEST ARGUMENT TABLE TERMINATOR
                        0034       TEST_001::
                        0034                        .WORD    M<>                ; ENTRY MASK
                        0036   142 ;++
                        0036   143 ; FUNCTIONAL DESCRIPTION:
                        0036   144 ;
                        0036   145 ; THIS TEST VERIFIES THAT THE INTERNAL REGISTERS, MAP REGISTERS,
                        0036   146 ; AND MBA REGISTER SPACE EXIST, AND THAT THE INTERNAL BUS (IBUS)
                        0036   147 ; IS CAPABLE OF TRANSFERRING 0'S AND 1'S IN ALL BIT
                        0036   148 ; POSITIONS.
                        0036   149 ;--
                        0036   150
                        0036   151
                        0036   152          $DS_BGNSUB
                        0036       T1_S1::
    00000000'9F   00000000'EF   FA  0036                CALLG   $S$, a#DS$BGNSUB
                        0041   153 ;
                        0041   154 ;  Subtest one performs a read of each Internal Register to
                        0041   155 ;  verify its existence. The Map Selected Register is not
                        0041   156 ;  implemented in the RH750 and is therefore bypassed.
                        0041   157 ;
            52  00000000'EF   D0  0041   158          MOVL    RH_CUR_ADR,R2       ; R2 <-- RH750 BASE ADDRESS
                          54    D4  0048   159          CLRL    R4                  ; CLEAR INTERNAL REG INDEX COUNTER
00000000'EF   00000076'EF       DE  004A   160          MOVAL   20$,CONT_PC         ; STORE CONT PC
                        0055   161          $DS_SETVEC_S    #4,MACH_CHK_SRV ; GET MACHINE CHECK VECTOR
                          00    DD  0055                PUSHL   #0
                  00000000'EF   DF  0057                PUSHAL  MACH_CHK_SRV
                          04    DD  005D                PUSHL   #4
          00000000'9F     03    FB  005F                CALLS   #3, a#DS$SETVEC
          00000000'EF     01    88  0066   162 10$:     BISB2   #1,NXM_RD_FLAG      ; SET FLAG IN EVENT OF RD NXM EXC
          00000000'EF     52    D0  006D   163          MOVL    R2,NXM_ADR          ; STORE INT REGISTER ADDRESS
                          62    D5  0074   164          TSTL    (R2)                ; READ INTERNAL REGISTER
       1B 00000000'EF     00    E0  0076   165 20$:     BBS     #0,NXM_RD_FLAG,30$  ; DID RD NXM EXCEPTION OCCUR?
                        007E   166          $DS_ERRHARD_S   #1,LUN,-         ; REPORT ERROR
                        007E   167                          ARRAY_MSG22,-
                        007E   168                          PRINT_NXADR
                  00000000'EF   DF  007E                PUSHAL  PRINT_NXADR
                  00000000'EF   DF  0084                PUSHAL  ARRAY_MSG22
                  00000000'EF   DD  008A                PUSHL   LUN
                          01    DD  0090                PUSHL   #1
          00000000'9F     04    FB  0092                CALLS   #$$M, a#DS$ERRHARD
                        0099   169
                        0099   170
                        0099   171 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   CA    AF      FA  0099                CALLG   10$, a#DS$CKLOOP
                52      04      C0  00A1   172          ADDL2   #4,R2               ; GENERATE NEXT INT REG ADRESS
 55   00000000'EF       18      C1  00A4   173          ADDL3   # X18,RH_CUR_ADR,R5 ; FORM MSR ADDRESS
                52      55      D1  00AC   174          CMPL    R5,R2               ; WATCH FOR MSR ADDRESS SPACE
                          03    12  00AF   175          BNEQ    40$                 ; IF NOT MSR SPACE, CONTINUE
                52      04      C0  00B1   176          ADDL2   #4,R2               ; GENERATE LAST INTERNAL REG ADDR
              AE 54     06      F3  00B4   177 40$:     AOBLEQ  #6,R4,10$           ; CONTINUE FOR ALL INT REGISTERS
                        00B8   178          $DS_CLRVEC_S    #4              ; RETURN VECTOR TO SUPERVISOR
```

```
                        04   DD  00B8                          PUSHL   #4
          00000000'9F   01   FB  00BA                          CALLS   #1, a#DS$CLRVEC
                                 00C1   179          $DS_ENDSUB
                                 00C1        T1_S1_X:
   00000000'9F   00000000'EF   FA  00C1                          CALLG   $S$, a#DS$ENDSUB
```

```
                                  00CC   181 ;
                                  00CC   182 ;  Subtest two performs a read of each Map Register to verify its existence.
                                  00CC   183 ;
                                  00CC   184
                                  00CC   185          $DS_BGNSUB
                                  00CC       T1_S2::
00000000'9F   0000000C'EF   FA    00CC                       CALLG   $S$, a#DS$BGNSUB
          52   00000000'EF   D0    00D7   186          MOVL    RH_CUR_ADR,R2           ; LOAD RH BASE ADDRESS
     0000'C2   00000000'8F   C8    00DE   187          BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
                         5A   D4    00E7   188          CLRL    R10                     ; CLEAR MAP SELECTION INDEX COUNTER
00000000'EF   0000011E'EF   DE    00E9   189          MOVAL   20$,CONT_PC             ; STORE CONT PC
          52   00000000'8F   C0    00F4   190          ADDL2   #MAP_OFFSET,R2          ; R2 POINTS TO FIRST MAP REGISTER
                                  00FB   191          $DS_SETVEC_S    #4,MACH_CHK_SRV ; GET MACHINE CHECK VECTOR
                         00   DD    00FB                       PUSHL   #0
                00000000'EF   DF    00FD                       PUSHAL  MACH_CHK_SRV
                         04   DD    0103                       PUSHL   #4
          00000000'9F   03   FB    0105                       CALLS   #3, a#DS$SETVEC
          00000000'EF   01   88    010C   192 10$:     BISB2   #1,NXM_RD_FLAG          ; SET FLAG IN EVENT OF NXM MAP REG
       00000000'EF    624A   DE    0113   193          MOVAL   (R2) R10 ,NXM_ADR       ; STORE MAP REGISTER ADDRESS
                      624A   D5    011B   194          TSTL    (R2) R10                ; READ MAP REGISTER
     1B 00000000'EF   00   E0    C11E   195 20$:     BBS     #0,NXM_RD_FLAG,30$      ; DID READ NXM EXC OCCUR?
                                  0126   196          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                  0126   197                          ARRAY_MSG60,-
                                  0126   198                          PRINT_NXADR
                00000000'EF   DF    0126                       PUSHAL  PRINT_NXADR
                00000000'EF   DF    012C                       PUSHAL  ARRAY_MSG60
                00000000'EF   DD    0132                       PUSHL   LUN
                         01   DD    0138                       PUSHL   #1
          00000000'9F   04   FB    013A                       CALLS   #$$M, a#DS$ERRHARD
                                  0141   199
                                  0141   200
                                  0141   201 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
     00000000'9F   C8 AF   FA    0141                       CALLG   10$, a#DS$CKLOOP
     BB 5A   000000FF 8F   F3    0149   202          AOBLEQ  #255,R10,10$            ; CONTINUE FOR 256 MAP REGISTERS
                                  0151   203          $DS_CLRVEC_S    #4              ; RETURN VECTOR TO SUPERVISOR
                         04   DD    0151                       PUSHL   #4
          00000000'9F   01   FB    0153                       CALLS   #1, a#DS$CLRVEC
                                  015A   204          $DS_ENDSUB
                                  015A       T1_S2_X:
00000000'9F   0000000C'EF   FA    015A                       CALLG   $S$, a#DS$ENDSUB
```

```
                                0165   206 ;
                                0165   207 ;  Subtest three performs a read of all External Registers in the
                                0165   208 ;  MBA address space to verify their existence.
                                0165   209 ;
                                0165   210
                                0165   211          $DS_BGNSUB
                                0165          T1_S3::
00000000'9F   00000018'EF   FA  0165                       CALLG   $S$, a#DS$BGNSUB
        52    00000000'EF   D0  0170   212          MOVL    RH_CUR_ADR,R2            ; LOAD RH BASE ADDRESS
    0000'C2   00000000'8F   C8  0177   213          BISL    #PGM_INIT,CR(R2)         ; INIT RH750
                        54  D4  0180   214          CLRL R4                          ; CLEAR EXTERNAL REGISTER INDEX COUNTER
00000000'EF   000001B5'EF   DE  0182   215          MOVAL   20$,CONT_PC              ; STORE CONT PC
                            18D   216          $DS_SETVEC_S    #4,MACH_CHK_SRV ; GET MACHINE CHECK VECTOR
                        00  DD  018D                       PUSHL   #0
                00000000'EF  DF  018F                       PUSHAL  MACH_CHK_SRV
                        04  DD  0195                       PUSHL   #4
          00000000'9F   03  FB  0197                       CALLS   #3, a#DS$SETVEC
          58    00000000'EF   D0  019E   217          MOVL    DRIVE0,R8                ; STORE ADDRESS OF DRIVE 0
          00000000'EF   01  88  01A5   218 10$:     BISB2   #1,NXM_RD_FLAG           ; SET FLAG IN EVENT OF NXM READ
          00000000'EF   58  D0  01AC   219          MOVL    R8,NXM_ADR               ; STORE EXTERNAL REGISTER ADDRESS
                        68  D5  01B3   220          TSTL    (R8)                     ; READ DRIVE REGISTER
       1B 00000000'EF   00  E0  01B5   221 20$:     BBS     #0,NXM_RD_FLAG,30$       ; DID READ NXM DRIVE REG OCCUR?
                            01BD   222          $DS_ERRHARD_S   #1,LUN,-         ; REPORT ERROR
                            01BD   223                          ARRAY_MSG14,-
                            01BD   224                          PRINT_NXADR
                00000000'EF  DF  01BD                       PUSHAL  PRINT_NXADR
                00000000'EF  DF  01C3                       PUSHAL  ARRAY_MSG14
                00000000'EF  DD  01C9                       PUSHL   LUN
                        01  DD  01CF                       PUSHL   #1
          00000000'9F   04  FB  01D1                       CALLS   #$$M, a#DS$ERRHARD
                            01D8   225
                            01D8   226
                            01D8   227 30$:     $DS_CKLOOP      10$              ; SCOPE LOOP?
       00000000'9F   CA  AF  FA  01D8                       CALLG   10$, a#DS$CKLOOP
                    58  04  C0  01E0   228          ADDL2   #4,R8                    ; GENERATE NEXT EXTERNAL ADDRESS
       BA 54    000000FF  8F  F3  01E3   229          AOBLEQ  #255,R4,10$              ; CONTINUE FOR 256 EXT REGISTERS
                            01EB   230          $DS_CLRVEC_S    #4               ; RETURN VECTOR TO SUPERVISOR
                        04  DD  01EB                       PUSHL   #4
          00000000'9F   01  FB  01ED                       CALLS   #1, a#DS$CLRVEC
                            01F4   231          $DS_ENDSUB
                            01F4          T1_S3_X:
00000000'9F   00000018'EF   FA  01F4                       CALLG   $S$, a#DS$ENDSUB
```

```
                                01FF    233 ;
                                01FF    234 ; Subtest four verifies that the Internal Bus is capable of transferring
                                01FF    235 ; zeroes and ones in all bit positions.  This is accomplished  by reading
                                01FF    236 ; the Configuration Register after performing an Init. This checks for any
                                01FF    237 ; SA1 conditions. All ones are then written into the VAR and DR to check
                                01FF    238 ; that the IBUS has no SA0 faults.
                                01FF    239 ;
                                01FF    240
                                01FF    241         $DS_BGNSUB
                                01FF            T1_S4::
    00000000'9F   00000024'EF   FA  01FF                    CALLG   $S$, a#DS$BGNSUB
        0000'C2   00000000'8F   C8  020A    242 10$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE
                53   0000'C2    D0  0213    243         MOVL    CSR(R2),R3              ; READ CONFIGURATION REGISTER
                          53    13  0218    244         BEQL    30$                     ; SKIP IF NO STUCK AT ONE BITS
                                021A    245         $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                021A    246                         ARRAY_MSG14     ;
                          00    DD  021A                    PUSHL   #0
                00000000'EF     DF  021C                    PUSHAL  ARRAY_MSG14
                00000000'EF     DD  0222                    PUSHL   LUN
                          01    DD  0228                    PUSHL   #1
            00000000'9F   04    FB  022A                    CALLS   #$$M, a#DS$ERRHARD
                                0231    247         $DS_CVTREG_S    #31,R3,-        ; CONVERT REGISTER BITS TO MNEMONICS
                                0231    248                         FMT_ANY_REG,-
                                0231    249                         REPORT_BUFFER,-
                                0231    250                         #512
                          00    DD  0231                    PUSHL   #0
                          00    DD  0233                    PUSHL   #0
                          00    DD  0235                    PUSHL   #0
                          00    DD  0237                    PUSHL   #0
                          00    DD  0239                    PUSHL   #0
                          00    DD  023B                    PUSHL   #0
                00000200  8F    DD  023D                    PUSHL   #512
                00000000'EF     9F  0243                    PUSHAB  REPORT_BUFFER
                00000000'EF     9F  0249                    PUSHAB  FMT_ANY_REG
                          53    DD  024F                    PUSHL   R3
                          1F    DD  0251                    PUSHL   #31
        00000000'9F     0B    FB  0253                    CALLS   #11, a#DS$CVTREG
                                025A    251         $DS_PRINTX_S    FMT_IBUS1,-     ; PRINT IBUS ERROR
                                025A    252                         #REPORT_BUFFER
                00000000'8F     DD  025A                    PUSHL   #REPORT_BUFFER
                00000000'EF     9F  0260                    PUSHAB  FMT_IBUS1
        00000000'9F     02      FB  0266                    CALLS   #$$N, a#DS$PRINTX
                                026D    253 30$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
      00000000'9F     9A AF       FA  026D                    CALLG   10$, a#DS$CKLOOP
0000'C2     00000000'8F     C8  0275    254 40$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE THE RH
          0000'C2     00      D2  027E    255         MCOML   #0,VAR(R2)              ; WRITE 1'S INTO VAR REGISTER
          53    0000'C2       D0  0283    256         MOVL    VAR(R2),R3              ; STORE VAR
0000'C2     00000000'8F     C8  0288    257         BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
          0000'C2     00      D2  0291    258         MCOML   #0,DR(R2)               ; WRITE 1'S INTO DIAGNOSTIC REGISTER
          54    0000'C2       D0  0296    259         MOVL    DR(R2),R4               ; STORE DIAGNOSTIC REGISTER
                53    54      C8  029B    260         BISL    R4,R3                   ; MERGE ALL BITS READ FROM VAR AND DR
                53    53      D2  029E    261         MCOML   R3,R3                   ; COMPLEMENT RESULTS
                                02A1    262                                         ; IF THE IBUS CAN SUPPORT A ONE IN
                                02A1    263                                         ; EVERY BIT POSITION R3 WILL EQUAL 0
                          53    13  02A1    264         BEQL    50$                 ; SKIP IF NO STUCK AT ZEROES
                                02A3    265         $DS_ERRHARD_S   #2,LUN,-        ; REPORT ERROR
                                02A3    266                         ARRAY_MSG1      ;
```

```
                      00    DD  02A3                        PUSHL   #0
            00000000'EF     DF  02A5                        PUSHAL  ARRAY_MSG1
            00000000'EF     DD  02AB                        PUSHL   LUN
                      02    DD  02B1                        PUSHL   #2
      00000000'9F     04    FB  02B3                        CALLS   #$$M, a#DS$ERRHARD
                            02BA  267          $DS_CVTREG_S    #31,R3,-        ; CONVERT REGISTER BITS TO MNEMONICS
                            02BA  266                          FMT_ANY_REG,-
                            02BA  269                          REPORT_BUFFER,-
                            02BA  270                          #512
                      00    DD  02BA                        PUSHL   #0
                      00    DD  02BC                        PUSHL   #0
                      00    DD  02BE                        PUSHL   #0
                      00    DD  02C0                        PUSHL   #0
                      00    DD  02C2                        PUSHL   #0
                      00    DD  02C4                        PUSHL   #0
            00000200  8F    DD  02C6                        PUSHL   #512
            00000000'EF     9F  02CC                        PUSHAB  REPORT_BUFFER
            00000000'EF     9F  02D2                        PUSHAB  FMT_ANY_REG
                      53    DD  02D8                        PUSHL   R3
                      1F    DD  02DA                        PUSHL   #31
      00000000'9F     0B    FB  02DC                        CALLS   #11, a#DS$CVTREG
                            02E3  271          $DS_PRINTX_S    FMT_IBUS0,-     ; AND PRINT THEM
                            02E3  272                          #REPORT_BUFFER
            00000000'8F     DD  02E3                        PUSHL   #REPORT_BUFFER
            00000000'EF     9F  02E9                        PUSHAB  FMT_IBUS0
      00000000'9F     02    FB  02EF                        CALLS   #$$N, a#DS$PRINTX
                            02F6  273 50$:     $DS_CKLOOP      40$             ; SCOPE LOOP?
00000000'9F     FF7B CF       FA  02F6                        CALLG   40$, a#DS$CKLOOP
000J'C2   00000000'8F         D0  02FF  274          MOVL    #PGM_INIT,CR(R2)        ; INIT THE RH750
                            0308  275          $DS_ENDSUB
                            0308       T1_S4_X:
00000000'9F   00000024'EF     FA  0308                        CALLG   $S$, a#DS$ENDSUB
                            0313  276

                            0313  277          $DS_BGNSUB
                            0313       T1_S5::
00000000'9F   00000030'EF     FA  0313                        CALLG   $S$, a#DS$BGNSUB
                            031E  278
                            031E  279          $DS_SETVEC_S    # X60,5$        ; Set vector 60 (write timeout) to 5$
                      00    DD  031E                        PUSHL   #0
            000003B0'EF     DF  0320                        PUSHAL  5$
            00000060  8F    DD  0326                        PUSHL   # X60
      00000000'9F     03    FB  032C                        CALLS   #3, a#DS$SETVEC
                            0333  280          $DS_SETVEC_S    # X20,5$        ; Set vector 20 (mm write timeout) to 5$
                      00    DD  0333                        PUSHL   #0
            000003B0'EF     DF  0335                        PUSHAL  5$
                      20    DD  033B                        PUSHL   # X20
      00000000'9F     03    FB  033D                        CALLS   #3, a#DS$SETVEC
                            0344  281          $DS_SETVEC_S    # X04,5$        ; Set vector 04 (machine check) to 5$
                      00    DD  0344                        PUSHL   #0
            000003B0'EF     DF  0346                        PUSHAL  5$
                      04    DD  034C                        PUSHL   # X04
      00000000'9F     03    FB  034E                        CALLS   #3, a#DS$SETVEC
      53    00000000'EF     D0  0355  282          MOVL    RH_CUR_ADR,R3           ; GET BASE ADRRESS FOR UUT
      55    40F28800  8F     D0  035C  283          MOVL    # X40F28800,R5         ; Move 1st address to check to R5
      000003A8'EF     5E    D0  0363  284          MOVL    SP,3$                   ; Move stack pointer value to 3$
000003AC'EF   00000379'EF     DE  036A  285          MOVAL   2$,4$                   ; Return to 2$
```

```
ZZ-ECCAA-1.8    1.8                 TEST 1:  REGISTE20-FEB-1985          Fiche 1  Frame L9          Sequence 115
                      65   D5  0375   286 1$:     TSTL    (R5)              ; See if address exists
```

```
                    65   D4  0377   287          CLRL    (R5)                      ; Clear it if it does
        55   00002000 8F   C0  0379   288 2$:     ADDL2   # X2000,R5                ; Move to next test cell
        55   40F30800 8F   D1  0380   289          CMPL    # X40F30800,R5            ; Have we checked them all?
                    EC   12  0387   290          BNEQ    1$                        ; Branch back if not
                             0389   291          $DS_CLRVEC_S    # X60             ; Return timeout vector to the DS>
             00000060 8F   DD  0389                      PUSHL   # X60
        00000000'9F   01   FB  038F                      CALLS   #1, a#DS$CLRVEC
        55   40F28800 8F   D0  0396   292          MOVL    # X40F28800,R5            ; Restore beginning of window to R5
   0800 C3   000000FF 8F   C8  039D   293          BISL2   # XFF, X800(R3)           ; Set some non-critical bits in uut register
                    12   11  03A6   294          BRB     6$                        ; Branch over timeout routines
                             03A8   295 ;        SUBL3   # X40F28000,R3,R7
                             03A8   296 ;20$:    TSTL    R7                        ; See which rh this is
                             03A8   297 ;        BEQL    6$                        ; Branch over timeout routines
                             03A8   298 ;        SUBL2   # X2000,R7                ; Clear the initial address
                             03A8   299 ;        CLRL    (R5) R7                   ;  for the map registers to
                             03A8   300 ;        BRB     20$                       ;  all the RHs
                             03A8   301
                             03A8   302
                  00000000   03A8   303 3$:     .LONG   0                         ; Location for stack pointer
                  00000000   03AC   304 4$:     .LONG   0                         ; Location for return address
                             03B0   305
                             03B0   306          .ALIGN  LONG                      ; Align timeout routine to longword
             5E   F5 AF   D0  03B0   307 5$:     MOVL    3$,SP                     ; Clean the stack
                26   0F   DA  03B4   308          MTPR    # XF,# X26                ; Clear error register
                   F2 BF   17  03B7   309          JMP     a4$                       ; Go back to routine
                             03BA   310
     EA AF   00000401'EF   DE  03BA   311 6$:     MOVAL   8$,4$                     ; Returns to 8$
                    65   D5  03C2   312 7$:     TSTL    (R5)                      ; See if location is non-zero
                    3B   13  03C4   313          BEQL    8$                        ; Branch if zero
   54   53   00000800 8F   C1  03C6   314          ADDL3   # X800,R3,R4              ; Form UUT address
                 54   55   D1  03CE   315          CMPL    R5,R4                     ; See if address is the UUT
                    2E   13  03D1   316          BEQL    8$                        ; Branch if so
           56   55   54   CD  03D3   317          XORL3   R4,R5,R6                  ; Form XOR data
                             03D7   318          $DS_ERRHARD_S   #1,LUN,-
                             03D7   319                          FMT_DUALRS_ERR    ; Print error
                    00   DD  03D7                      PUSHL   #0
             00000000'EF   DF  03D9                      PUSHAL  FMT_DUALRS_ERR
             00000000'EF   DD  03DF                      PUSHL   LUN
                    01   DD  03E5                      PUSHL   #1
        00000000'9F   04   FB  03E7                      CALLS   #$$M, a#DS$ERRHARD
                             03EE   320          $DS_PRINTX_S    FMT_DUALRS_MSG,-
                             03EE   321                          R4,R5,R6          ; Print error parameters
                    56   DD  03EE                      PUSHL   R6
                    55   DD  03F0                      PUSHL   R5
                    54   DD  03F2                      PUSHL   R4
             00000000'EF   9F  03F4                      PUSHAB  FMT_DUALRS_MSG
        00000000'9F   04   FB  03FA                      CALLS   #$$N, a#DS$PRINTX
        55   00002000 8F   C0  0401   322 8$:     ADDL2   # X2000,R5                ; Advance to next location and
        55   40F30800 8F   D1  0408   323          CMPL    # X40F30800,R5            ; Are we done yet?
                    B1   12  040F   324          BNEQ    7$                        ; Branch if another address to check
                             0411   325          $DS_CLRVEC_S    # X04             ; Return timeout vector to the DS>
                    04   DD  0411                      PUSHL   # X04
        00000000'9F   01   FB  0413                      CALLS   #1, a#DS$CLRVEC
                             041A   326          $DS_CLRVEC_S    # X20             ; Return MM timeout vector to the DS>
                    20   DD  041A                      PUSHL   # X20
        00000000'9F   01   FB  041C                      CALLS   #1, a#DS$CLRVEC
                             0423   327          $DS_ENDSUB
```

```
                                0423        T1_S5_X:
    00000000'9F   00000030'EF   FA  0423                    CALLG   $S$, a#DS$ENDSUB
```

```
                                  042E   329 $DS_ENDTEST
                  50   01   D0    042E                       MOVL    #1, R0          ; NORMAL EXIT
                                  0431       TEST_001_X::
        00000000'9F   6E   FA     0431                       CALLG   (SP), a#DS$BREAK
                             04   0438                       RET                     ; RETURN TO TEST SEQUENCER
                                  0439   330 $DS_PAGE
                                  0439                .SBTTL  TEST 2:  INITIALIZATION TEST
                             00000000                       .PSECT  TEST_002, PAGE, NOWRT
                                  0018
                                  0018   332 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  0018       DATA_002:
                       00000000   0018                       .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  001C       TEST_002::
                           0000   001C                       .WORD    M<>            ; ENTRY MASK
                                  001E   333 ;++
                                  001E   334 ; TEST DESCRIPTION:
                                  001E   335 ;
                                  001E   336 ; THIS TEST INITS THE RH, CLEARS ALL INTERNAL REGISTERS AND THEN
                                  001E   337 ; READS EACH REGISTER CHECKING FOR STUCK AT ONE BIT FAULTS.
                                  001E   338 ;
                                  001E   339 ; ASSUMPTIONS:
                                  001E   340 ;
                                  001E   341 ;       THIS TEST AND ALL SUBSEQUENT TESTS ASSUMES RH_CUR_ADR HAS
                                  001E   342 ;       BEEN INITIALIZED; I.E. CONTAINS THE ADDRESS OF
                                  001E   343 ;       THE RH750 UNDER TEST (MUT).
                                  001E   344 ;
                                  001E   345 ; TEST ALGORITHM:
                                  001E   346 ;
                                  001E   347 ; CLEAR THE STATUS REGISTER AND CHECK FOR ANY STUCK AT ONE BITS.
                                  001E   348 ; READ THE CONTROL REGISTER AND CHECK FOR ANY STUCK AT ONE BITS.
                                  001E   349 ; READ THE DIAGNOSTIC REGISTER AND CHECK FOR ANY STUCK AT ONE BITS.
                                  001E   350 ; CLEAR THE VIRTUAL ADDRESS REGISTER AND CHECK FOR ANY STUCK AT ONE BITS.
                                  001E   351 ; CLEAR THE BYTE COUNT REGISTER AND CHECK FOR ANY STUCK AT ONE BITS.
                                  001E   352 ; CLEAR EACH MAP REGISTER AND READ EACH ONE BACK, REPORT ANY NON ZERO
                                  001E   353 ; MAP ENTRIES.
                                  001E   354 ;--
          52   00000000'EF   D0   001E   355         MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS TO R2
                         54   D4   0025   356         CLRL    R4                      ; SET EXPECTED DATA
     0000'C2   00000000'8F   C8   0027   357 10$:    BISL2   #PGM_INIT,CR(R2)        ; INIT THE RH750 UNDER TEST (MUT)
                                  0030   358 20$:    ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                  0030   359                 FMT_STATUS_REG
 00000000'EF   00000000'EF   DE   0030                       MOVAL   RHSR_MSG,REG_NAME
          00000000'EF   02   9A   003B                       MOVZBL  #2,REG_NO
 00000000'EF   00000000'EF   DE   0042                       MOVAL   FMT_STATUS_REG,REG_STRING
                    0000'C2   D4   004D   360         CLRL    SR(R2)                  ; CLEAR STATUS REGISTER
              53    0000'C2   D0   0051   361         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                         1B   13   0056   362         BEQL    50$                     ; BRANCH IF NO ERRORS
                                  0058   363         $DS_ERRHARD_S   #1,LUN,-        ; PRINT STATUS REGISTER ERROR
                                  0058   364                         ARRAY_MSG1,-
                                  0058   365                         PRINT_SBE
              00000000'EF   DF   0058                       PUSHAL  PRINT_SBE
              00000000'EF   DF   005E                       PUSHAL  ARRAY_MSG1
              00000000'EF   DD   0064                       PUSHL   LUN
                         01   DD   006A                       PUSHL   #1
          00000000'9F   04   FB   006C                       CALLS   #$$M, a#DS$ERRHARD
                                  0073   366 50$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
       00000000'9F   B1 AF   FA   0073                       CALLG   10$, a#DS$CKLOOP
```

```
                                    007B   367          ERRPREP RHCR_MSG,1,-            ; PREPARE TO PRINT ERROR
                                    007B   368                  FMT_CONTRL_REG
    00000000'EF   00000000'EF   DE  007B                        MOVAL   RHCR_MSG,REG_NAME
          00000000'EF   01      9A  0086                        MOVZBL  #1,REG_NO
    00000000'EF   00000000'EF   DE  008D                        MOVAL   FMT_CONTRL_REG,REG_STRING
              53   0000'C2      D0  0098   369          MOVL    CR(R2),R3               ; READ CONTROL REGISTER
                        1B      13  009D   370          BEQL    60$                     ; BRANCH IF OK
                                    009F   371          $DS_ERRHARD_S   #2,LUN,-        ; REPORT CONTROL REGISTER ERROR
                                    009F   372                  ARRAY_MSG56,-   ;
                                    009F   373                  PRINT_SBE
                00000000'EF    DF   009F                        PUSHAL  PRINT_SBE
                00000000'EF    DF   00A5                        PUSHAL  ARRAY_MSG56
                00000000'EF    DD   00AB                        PUSHL   LUN
                          02    DD   00B1                        PUSHL   #2
             00000000'9F      04   FB  00B3                        CALLS   #$$M, a#DS$ERRHARD
                                    00BA   374 60$:      $DS_CKLOOP      10$             ; SCOPE LOOP?
        00000000'9F    FF69 CF   FA  00BA                        CALLG   10$, a#DS$CKLOOP
                                    00C3   375          ERRPREP RHDR_MSG,5,FMT_DIAG_REG ; PREPARE TO HANDLE ERROR
    00000000'EF   00000000'EF   DE  00C3                        MOVAL   RHDR_MSG,REG_NAME
          00000000'EF   05      9A  00CE                        MOVZBL  #5,REG_NO
    00000000'EF   00000000'EF   DE  00D5                        MOVAL   FMT_DIAG_REG,REG_STRING
   53   0000'C2   00000000'8F   CB  00E0   376          BICL3   #MASS_CTOD!BYTE1!BYTE0,DR(R2),R3 ; READ THE REGISTER
                        1B      13  00EA   377          BEQL    70$                     ; REPORT ELSE CONTINUE
                                    00EC   378          $DS_ERRHARD_S   #3,LUN,-        ; REPORT ERROR
                                    00EC   379                  ARRAY_MSG1,-
                                    00EC   380                  PRINT_SBE
                00000000'EF    DF   00EC                        PUSHAL  PRINT_SBE
                00000000'EF    DF   00F2                        PUSHAL  ARRAY_MSG1
                00000000'EF    DD   00F8                        PUSHL   LUN
                          03    DD   00FE                        PUSHL   #3
             00000000'9F      04   FB  0100                        CALLS   #$$M, a#DS$ERRHARD
                                    0107   381 70$:      $DS_CKLOOP      10$             ; SCOPE LOOP?
        00000000'9F    FF1C CF   FA  0107                        CALLG   10$, a#DS$CKLOOP
                                    0110   382          ERRPREP RHVAR_MSG,3,FMT_ANY_REG ; PREPARE TO HANDLE ERROR
    00000000'EF   00000000'EF   DE  0110                        MOVAL   RHVAR_MSG,REG_NAME
          00000000'EF   03      9A  011B                        MOVZBL  #3,REG_NO
    00000000'EF   00000000'EF   DE  0122                        MOVAL   FMT_ANY_REG,REG_STRING
              0000'C2   00      D2  012D   383 75$:      MCOML   #0,VAR(R2)              ; WRITE ONE'S INTO VAR
                   0000'C2      D4  0132   384          CLRL    VAR(R2)                 ; CLEAR VIRTUAL ADDRESS REGISTER
              53   0000'C2      D0  0136   385          MOVL    VAR(R2),R3              ; READ VIRTUAL ADDRESS REGISTER
                        1B      13  013B   386          BEQL    80$                     ; BRANCH IF NO ERRORS
                                    013D   387          $DS_ERRHARD_S   #4,LUN,-        ; REPORT ERROR
                                    013D   388                  ARRAY_MSG23,-   ;
                                    013D   389                  PRINT_SBE
                00000000'EF    DF   013D                        PUSHAL  PRINT_SBE
                00000000'EF    DF   0143                        PUSHAL  ARRAY_MSG23
                00000000'EF    DD   0149                        PUSHL   LUN
                          04    DD   014F                        PUSHL   #4
             00000000'9F      04   FB  0151                        CALLS   #$$M, a#DS$ERRHARD
                                    0158   390 80$:      $DS_CKLOOP      10$             ; SCOPE LOOP?
        00000000'9F    FECB CF   FA  0158                        CALLG   10$, a#DS$CKLOOP
                                    0161   391          ERRPREP RHBCR_MSG,4,FMT_ANY_REG ; PREPARE TO HANDLE ERROR
    00000000'EF   00000000'EF   DE  0161                        MOVAL   RHBCR_MSG,REG_NAME
          00000000'EF   04      9A  016C                        MOVZBL  #4,REG_NO
    00000000'EF   00000000'EF   DE  0173                        MOVAL   FMT_ANY_REG,REG_STRING
              0000'C2   00      D2  017E   392 85$:      MCOML   #0,BCR(R2)              ; WRITE ONE'S TO BYTE COUNTER
                   0000'C2      D4  0183   393          CLRL    BCR(R2)                 ; CLEAR BYTE COUNTER REGISTER
```

```
                53   0000'C2   D0  0187   394          MOVL    BCR(R2),R3              ; READ BYTE COUNTER
                          1B   13  018C   395          BEQL    90$                     ; BRANCH IF ALL ZEROES
                               018E   396          $DS_ERRHARD_S   #5,LUN,-         ; REPORT ERROR
                               018E   397                          ARRAY_MSG14,-    ;
                               018E   398                          PRINT_SBE
                00000000'EF    DF  018E                          PUSHAL  PRINT_SBE
                00000000'EF    DF  0194                          PUSHAL  ARRAY_MSG14
                00000000'EF    DD  019A                          PUSHL   LUN
                          05   DD  01A0                          PUSHL   #5
             00000000'9F   04   FB  01A2                          CALLS   #$$M, a#DS$ERRHARD
                               01A9   399 90$:      $DS_CKLOOP      10$             ; SCOPE LOOP?
          00000000'9F   FE7A CF   FA  01A9                          CALLG   10$, a#DS$CKLOOP
                               01B2   400          ERRPREP RHMAPR_MSG,10,-          ; PREPARE TO PRINT ERROR
                               01B2   401                          FMT_MAP_REG
     00000000'EF   00000000'EF   DE  01B2                          MOVAL   RHMAPR_MSG,REG_NAME
             00000000'EF   0A   9A  01BD                          MOVZBL  #10,REG_NO
     00000000'EF   00000000'EF   DE  01C4                          MOVAL   FMT_MAP_REG,REG_STRING
                          5A   D4  01CF   402          CLRL    R10                     ; CLEAR MAP SELECTION INDEX
             52   00000000'8F   C0  01D1   403          ADDL2   #MAP_OFFSET,R2          ; R2 NOW POINTS TO THE FIRST MAP REG
                        624A   D4  01D8   404 100$:     CLRL    (R2) R10                ; CLEAR MAP REGISTER R10
                   53   624A   D0  01DB   405          MOVL    (R2) R10 ,R3            ; READ MAP REGISTER R10
                          1B   13  01DF   406          BEQL    110$                    ; BRANCH IF NO BITS SET
                               01E1   407          $DS_ERRHARD_S   #6,LUN,-         ; PRINT ERROR
                               01E1   408                          ARRAY_MSG14,-    ;
                               01E1   409                          PRINT_SBE
                00000000'EF    DF  01E1                          PUSHAL  PRINT_SBE
                00000000'EF    DF  01E7                          PUSHAL  ARRAY_MSG14
                00000000'EF    DD  01ED                          PUSHL   LUN
                          06   DD  01F3                          PUSHL   #6
             00000000'9F   04   FB  01F5                          CALLS   #$$M, a#DS$ERRHARD
                               01FC   410 110$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
     00000000'9F   FE27 CF   FA  01FC                          CALLG   10$, a#DS$CKLOOP
       CB 5A   000000FF 8F   F3  0205   411          AOBLEQ  #255,R10,100$           ; TEST ALL MAP REGISTERS
                               020D   412 $DS_ENDTEST
                     50   01   D0  020D                          MOVL    #1, R0          ; NORMAL EXIT
                               0210       TEST_002_X::
             00000000'9F   6E   FA  0210                          CALLG   (SP), a#DS$BREAK
                               0217                          RET                     ; RETURN TO TEST SEQUENCER
                               0218   413 $DS_PAGE
```

```
                                0218                .SBTTL  TEST 3:  CONTROL REGISTER SA0 TEST
                         00000000                   .PSECT  TEST_003, PAGE, NOWRT
                                001C
                                001C    416 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                001C        DATA_003:
                         00000000       001C            .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                0020        TEST_003::
                             0000       0020            .WORD    M<>            ; ENTRY MASK
                                0022    417 ;++
                                0022    418 ; TEST DESCRIPTION:
                                0022    419 ;
                                0022    420 ; THIS TEST CHECKS FOR STUCK AT ZERO BITS IN THE RH750 CONTROL REGISTER.
                                0022    421 ;
                                0022    422 ;
                                0022    423 ;
                                0022    424 ; TEST ALGORITHM:
                                0022    425 ;
                                0022    426 ;       WRITE ONE'S INTO EACH BIT, THEN CLEAR.
                                0022    427 ;       WRITE EACH BIT IN THE CONTROL REGISTER
                                0022    428 ;       REPORT ERROR IF SELECTED BIT IS NOT SET
                                0022    429 ;--
                52   00000000'EF   D0    0022    430         MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS TO R2
                                0029    431         ERRPREP RHCR_MSG,1,-            ; PREPARE TO PRINT ERROR
                                0029    432                 FMT_CONTRL_REG,SA0_MSG
    00000000'EF   00000000'EF   DE      0029                 MOVAL   RHCR_MSG,REG_NAME
          00000000'EF   01      9A      0034                 MOVZBL  #1,REG_NO
    00000000'EF   00000000'EF   DE      003B                 MOVAL   FMT_CONTRL_REG,REG_STRING
                                0046    433 ;++
                                0046    434 ; TEST CLEARING
                                0046    435 ;--
                                0046    436         $DS_BGNSUB
                                0046        T3_S1::
00000000'9F   0000003C'EF   FA      0046                 CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8      0051    437 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT RH UNDER TEST
              0000'C2   1E      9A      005A    438         MOVZBL  # X1E,CR(R2)            ; WRITE ONE'S VIA D INPUTS TO CR
                  0000'C2   D4      005F    439         CLRL    CR(R2)                  ; CLEAR
              53  0000'C2   D0      0063    440         MOVL    CR(R2),R3               ; READ 0'S FROM CONTROL REGISTER
                        1D  13      0068    441         BEQL    20$                     ; SKIP IF NO ERRORS
                        54  D4      006A    442         CLRL    R4                      ; CLEAR EXPECTED RESULTS REGISTER
                                006C    443         $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                006C    444                         ARRAY_MSG57,-
                                006C    445                         PRINT_SBE
                   00000000'EF  DF      006C                 PUSHAL  PRINT_SBE
                   00000000'EF  DF      0072                 PUSHAL  ARRAY_MSG57
                   00000000'EF  DD      0078                 PUSHL   LUN
                             01 DD      007E                 PUSHL   #1
              00000000'9F   04  FB      0080                 CALLS   #$$M, a#DS$ERRHARD
                                0087    446 20$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
          00000000'9F   C7 AF   FA      0087                 CALLG   10$, a#DS$CKLOOP
                                008F    447         $DS_ENDSUB
                                008F        T3_S1_X:
00000000'9F   0000003C'EF   FA      008F                 CALLG   $S$, a#DS$ENDSUB
                                009A    448

                                009A    449 ;++
                                009A    450 ; TEST CONTROL REGISTER FOR STUCK AT ZEROES
                                009A    451 ;--
```

```
                                  009A        T3_S2::
00000000'9F   00000048'EF   FA  009A                        CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  00A5   453 10$:    BISL    #PGM_INIT,CR(R2)          ; INIT THE MUT
         54   00000000'8F   D0  00AE   454         MOVL    #MAINT_MODE,R4            ; INDICATE EXPECTED VALUE
    0000'C2   00000000'8F   C8  00B5   455         BISL    #MAINT_MODE,CR(R2)        ; WRITE MAINT MODE BIT IN CONTROL REG
            53   0000'C2    D0  00BE   456         MOVL    CR(R2),R3                 ; READ CONTROL REGISTER
    0000'C2   00000000'8F   CA  00C3   457         BICL    #MAINT_MODE,CR(R2)        ; CLEAR MAINT MODE
                 54   53    D1  00CC   458         CMPL    R3,R4                     ; RECEIVED = EXPECTED?
                      1B    13  00CF   459         BEQL    20$                       ; IF NO ERRORS BRANCH
                                  00D1   460         $DS_ERRHARD_S   #1,LUN,-          ; REPORT ERROR
                                  00D1   461                         ARRAY_MSG35,-
                                  00D1   462                         PRINT_SBE
            00000000'EF    DF  00D1                        PUSHAL  PRINT_SBE
            00000000'EF    DF  00D7                        PUSHAL  ARRAY_MSG35
            00000000'EF    DD  00DD                        PUSHL   LUN
                      01    DD  00E3                        PUSHL   #1
        00000000'9F   04    FB  00E5                        CALLS   #$$M, a#DS$ERRHARD
                                  00EC   463 20$:    $DS_CKLOOP      10$               ; SCOPE LOOP?
     00000000'9F   B6 AF    FA  00EC                        CALLG   10$, a#DS$CKLOOP
    0000'C2   00000000'8F   C8  00F4   464 30$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH750 UNDER TEST
         54   00000000'8F   D0  00FD   465         MOVL    #INTERRUPT_EN,R4          ; SET UP EXPECTED VALUE
            0000'C2   54    D0  0104   466         MOVL    R4,CR(R2)                 ; WRITE CONTROL REGISTER
            53   0000'C2    D0  0109   467         MOVL    CR(R2),R3                 ; READ CONTROL REGISTER
                 54   53    D1  010E   468         CMPL    R3,R4                     ; ANY DEVIATIONS FROM EXPECTED VALUE?
                      1B    13  0111   469         BEQL    40$                       ; SKIP ERROR REPORT IF NO ERRORS
                                  0113   470         $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                  0113   471                         ARRAY_MSG35,-
                                  0113   472                         PRINT_SBE
            00000000'EF    DF  0113                        PUSHAL  PRINT_SBE
            00000000'EF    DF  0119                        PUSHAL  ARRAY_MSG35
            00000000'EF    DD  011F                        PUSHL   LUN
                      02    DD  0125                        PUSHL   #2
        00000000'9F   04    FB  0127                        CALLS   #$$M, a#DS$ERRHARD
                                  012E   473 40$:    $DS_CKLOOP      30$               ; SCOPE LOOP?
     00000000'9F   C3 AF    FA  012E                        CALLG   30$, a#DS$CKLOOP
    0000'C2   00000000'8F   C8  0136   474 50$:    BISL    #PGM_INIT,CR(R2)          ; RESET RH750 UNDER TEST
         54   00000000'8F   D0  013F   475         MOVL    #DT_ABORT,R4              ; SET UP EXPECTED VALUE
            0000'C2   54    D0  0146   476         MOVL    R4,CR(R2)                 ; WRITE CONTROL
            53   0000'C2    D0  014B   477         MOVL    CR(R2),R3                 ; READ CONTROL REGISTER INTO R3
                 54   53    D1  0150   478         CMPL    R3,R4                     ; RECEIVED = EXPECTED?
                      1B    13  0153   479         BEQL    60$                       ; IF NO ERRORS SKIP REPORT
                                  0155   480         $DS_ERRHARD_S   #3,LUN,-          ; REPORT ERROR
                                  0155   481                         ARRAY_MSG31,-
                                  0155   482                         PRINT_SBE
            00000000'EF    DF  0155                        PUSHAL  PRINT_SBE
            00000000'EF    DF  015B                        PUSHAL  ARRAY_MSG31
            00000000'EF    DD  0161                        PUSHL   LUN
                      03    DD  0167                        PUSHL   #3
        00000000'9F   04    FB  0169                        CALLS   #$$M, a#DS$ERRHARD
                                  0170   483 60$:    $DS_CKLOOP      50$               ; SCOPE LOOP?
     00000000'9F   C3 AF    FA  0170                        CALLG   50$, a#DS$CKLOOP
    0000'C2   00000000'8F   C8  0178   484 70$:    BISL    #PGM_INIT,CR(R2)          ; RESET RH750 UNDER TEST
         54   00000000'8F   D0  0181   485         MOVL    #INH_BYTE_CNT,R4          ; SETUP EXPECTED VALUE
            0000'C2   54    D0  0188   486         MOVL    R4,CR(R2)                 ; WRITE CONTROL
            53   0000'C2    D0  018D   487         MOVL    CR(R2),R3                 ; READ CONTROL REGISTER
                 54   53    D1  0192   488         CMPL    R3,R4                     ; RECEIVED = EXPECTED?
                      1B    13  0195   489         BEQL    80$                       ; BRANCH IF YES
```

```
                                  0197   490          $DS_ERRHARD_S   #4,LUN,-          ; REPORT THE ERROR
                                  0197   491                          ARRAY_MSG37,-
                                  0197   492                          PRINT_SBE
              00000000'EF   DF    0197                       PUSHAL  PRINT_SBE
              00000000'EF   DF    019D                       PUSHAL  ARRAY_MSG37
              00000000'EF   DD    01A3                       PUSHL   LUN
                        04  DD    01A9                       PUSHL   #4
         00000000'9F    04  FB    01AB                       CALLS   #$$M, a#DS$ERRHARD
                                  01B2   493 80$:     $DS_CKLOOP      70$               ; SCOPE LOOP?
      00000000'9F   C3 AF   FA    01B2                       CALLG   70$, a#DS$CKLOOP
                                  01BA   494          $DS_ENDSUB
                                  01BA       T3_S2_X:
00000000'9F   00000048'EF   FA    01BA                       CALLG   $S$, a#DS$ENDSUB
                                  01C5   495

                                  01C5   496 $DS_ENDTEST
                  50    01  D0    01C5                       MOVL    #1, R0            ; NORMAL EXIT
                                  01C8       TEST_003_X::
         00000000'9F    6E  FA    01C8                       CALLG   (SP), a#DS$BREAK
                            04    01CF                       RET                       ; RETURN TO TEST SEQUENCER
                                  01D0   497 $DS_PAGE
                                  01D0               .SBTTL  TEST 4:  DIAGNOSTIC REGISTER TEST
                            00000000                       .PSECT  TEST_004, PAGE, NOWRT
                                  001C
                                  001C   499 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  001C       DATA_004:
                      00000000    001C                       .LONG   0                 ; TEST ARGUMENT TABLE TERMINATOR
                                  0020       TEST_004::
                          0000    0020                       .WORD    M<>              ; ENTRY MASK
                                  0022   500 ;++
                                  0022   501 ; TEST DESCRIPTION:
                                  0022   502 ;
                                  0022   503 ;       THIS TEST CHECKS THE DIAGNOSTIC REGISTER FOR ANY STUCK
                                  0022   504 ;       AT ONE BITS WHEN NO RH750 ACTIVITY AND IN THE MAINTENANCE MODE.
                                  0022   505 ;       IT THEN CHECKS FOR STUCK AT 0 CONDITIONS IN THE DIAGNOSTIC
                                  0022   506 ;       REGISTER.
                                  0022   507 ;
                                  0022   508 ; TEST ALGORITHM:
                                  0022   509 ;
                                  0022   510 ;       PUT RH750 INTO MAINTENANCE MODE
                                  0022   511 ;       CHECK FOR STUCK AT 1 BITS IN DIAGNOSTIC REGISTER
                                  0022   512 ;
                                  0022   513 ;       SET EACH WRITEABLE BIT
                                  0022   514 ;       READ IT BACK AND TEST FOR ERRORS
                                  0022   515 ;
                                  0022   516 ;*******************************************************************
                                  0022   517 ;
                                  0022   518 ; NOTE:
                                  0022   519 ;
                                  0022   520 ; IF MASS FAIL IS NOT ASSERTED THIS TEST WILL $DS_ESCAPE TO ADDRESS 0 IF AND
                                  0022   521 ; ONLY IF THE 'ABORT_IF_FAIL' IS SET TO A ONE. 'ABORT_IF_FAIL IS
                                  0022   522 ; SET TO A ONE IF THE OPERATOR INDICATES THAT THE PROGRAM IS TO ABORT
                                  0022   523 ; IF MASS FAIL IS NOT ASSERTED.
                                  0022   524 ;--
                                  0022   525          ERRPREP RHDR_MSG,5,-              ; PREPARE TO PRINT ERROR
                                  0022   526                  FMT_DIAG_REG,SA1_MSG
00000000'EF   00000000'EF   DE    0022                       MOVAL   RHDR_MSG,REG_NAME
```

ZZ-ECCAA-1.8 1.8 TEST 3: CONTROL 20-FEB-1985 Fiche 1 Frame I10 Sequence 125

00000000'EF 05 9A 002D MOVZBL #5,REG_NO

```
00000000'EF   00000000'EF   DE  0034                          MOVAL   FMT_DIAG_REG,REG_STRING
          52   00000000'EF   D0  003F   527          MOVL    RH_CUR_ADR,R2              ; MOVE MUT'S ADDRESS TO R2
                                0046   528          $DS_BGNSUB
                                0046       T4_S1::
00000000'9F   00000054'EF   FA  0046                          CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0051   529 10$:     BISL    #PGM_INIT,CR(R2)           ; INIT THE MUT
              0000'C2   00   D2  005A   530          MCOML   #0,DR(R2)                  ; TRY TO WRITE ONE'S INTO DIAG REG
              00000000'8F   CB  005F   531          BICL3   #MASS_CTOD!BYTE1!BYTE0,-
              53   0000'C2      0065   532                  DR(R2),R3                  ; READ DIAG REGISTER--CHECK FOR ZEROES
                        1D   13  0069   533          BEQL    15$                        ; IF ALL BITS IN DIAG REGISTER CLEAR
                                006B   534                                             ; THEN NO ERRORS
                        54   D4  006B   535          CLRL    R4                         ; CLEARED EXPECTED VALUE REGISTER
                                006D   536          $DS_ERRHARD_S   #1,LUN,-           ; PRINT ERROR
                                006D   537                          ARRAY_MSG1,-
                                006D   538                          PRINT_SBE
              00000000'EF   DF  006D                          PUSHAL  PRINT_SBE
              00000000'EF   DF  0073                          PUSHAL  ARRAY_MSG1
              00000000'EF   DD  0079                          PUSHL   LUN
                        01   DD  007F                          PUSHL   #1
         00000000'9F   04   FB  0081                          CALLS   #$$M, a#DS$ERRHARD
                                0088   539 15$:     $DS_CKLOOP      10$                ; SCOPE LOOP?
      00000000'9F   C6 AF   FA  0088                          CALLG   10$, a#DS$CKLOOP
                                0090   540          $DS_ENDSUB
                                0090       T4_S1_X:
00000000'9F   00000054'EF   FA  0090                          CALLG   $S$, a#DS$ENDSUB
                                009B   541

                                009B   542          $DS_BGNSUB
                                009B       T4_S2::
00000000'9F   00000060'EF   FA  009B                          CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  00A6   543 20$:     BISL    #PGM_INIT,CR(R2)           ; INIT THE RH
    0000'C2   00000000'8F   C8  00AF   544          BISL    #MAINT_MODE,CR(R2)         ; PUT INTO MAINTENANCE MODE
          54   00000000'8F   D0  00B8   545          MOVL    #MASS_FAIL,R4              ; EXPECT MASS FAIL ASSERTED
              00000000'8F   CB  00BF   546          BICL3   #MASS_CTOD!BYTE1!BYTE0,-
              53   0000'C2      00C5   547                  DR(R2),R3                  ; READ DIAG REGISTER W/O MBDIB FIELD
                   54   53   D1  00C9   548          CMPL    R3,R4                      ;
                        2E   13  00CC   549          BEQL    30$                        ; BRANCH IF OK
                                00CE   550          $DS_ERRHARD_S   #1,LUN,-           ; PRINT ERROR
                                00CE   551                          ARRAY_MSG8,-
                                00CE   552                          PRINT_SBE
              00000000'EF   DF  00CE                          PUSHAL  PRINT_SBE
              00000000'EF   DF  00D4                          PUSHAL  ARRAY_MSG8
              00000000'EF   DD  00DA                          PUSHL   LUN
                        01   DD  00E0                          PUSHL   #1
         00000000'9F   04   FB  00E2                          CALLS   #$$M, a#DS$ERRHARD
      0B 00000000'EF   00   E1  00E9   553          BBC     #0,ABORT_IF_FAIL,30$       ; IF FAIL CLEAR ABORT IFF FLAG IS SET
                                00F1   554          $DS_ESCAPE      SUB                ; ABORT TEST
00000000'9F   00000104'EF   FA  00F1                          CALLG   T4_S2_X, a#DS$ESCAPE
                                00FC   555 30$:     $DS_CKLOOP      20$                ; SCOPE LOOP?
      00000000'9F   A7 AF   FA  00FC                          CALLG   20$, a#DS$CKLOOP
                                0104   556          $DS_ENDSUB
                                0104       T4_S2_X:
00000000'9F   00000060'EF   FA  0104                          CALLG   $S$, a#DS$ENDSUB
                                010F   557

                                010F   558          $DS_BGNSUB
                                010F       T4_S3::
```

```
    00000000'9F   0000006C'EF   FA  010F                    CALLG   $S$, a#DS$BGNSUB
        0000'C2   00000000'8F   C8  011A   559 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH
```

```
                0000'C2   00000000'8F   C8  0123    560          BISL    #MAINT_MODE,CR(R2)       ; PUT RH750 INTO MAINTENANCE MODE
                      0000'C2   00      D2  012C    561          MCOML   #0,DR(R2)                ; SET ALL WRITEABLE BITS
                     54   00000000'8F   D0  0131    562          MOVL    #MASS_FAIL,R4            ; SET UP EXPECTED VALUE
                          0000'C2      D4  0138    563          CLRL    DR(R2)                   ; CLEAR ALL WRITEABLE BITS
                      00000000'8F      CB  013C    564          BICL3   #NIBBLE4!BYTE1!BYTE0,-
                     53   0000'C2              0142    565                  DR(R2),R3                ; READ DIAGNOSTIC REGISTIER
                          54   53     D1  0146    566          CMPL    R3,R4                    ; CHECK
                               1B     13  0149    567          BEQL    20$                      ; IF R/W FIELD IS CLEAR BRANCH
                                          014B    568          $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                          014B    569                          ARRAY_MSG2,-
                                          014B    570                          PRINT_SBE
                      00000000'EF      DF  014B                         PUSHAL  PRINT_SBE
                      00000000'EF      DF  0151                         PUSHAL  ARRAY_MSG2
                      00000000'EF      DD  0157                         PUSHL   LUN
                                01     DD  015D                         PUSHL   #1
                00000000'9F   04      FB  015F                         CALLS   #$$M, a#DS$ERRHARD
                                          0166    571 20$:     $DS_CKLOOP       10$              ; SCOPE LOOP?
                  00000000'9F   B1 AF     FA  0166                         CALLG   10$, a#DS$CKLOOP
                                          016E    572          $DS_ENDSUB
                                          016E         T4_S3_X:
            00000000'9F   0000006C'EF      FA  016E                         CALLG   $S$, a#DS$ENDSUB
                                          0179    573

                                          0179    574          $DS_BGNSUB
                                          0179         T4_S4::
            00000000'9F   00000078'EF      FA  0179                         CALLG   $S$, a#DS$BGNSUB
                0000'C2   00000000'8F      C8  0184    575 20$:     BISL    #PGM_INIT,CR(R2)         ; INIT FOR SCOPE LOOP
                0000'C2   00000000'8F      C8  018D    576          BISL    #MAINT_MODE,CR(R2)       ; RETURN TO MAINTENANCE MODE
                     54   00000000'8F      D0  0196    577          MOVL    #SIM_SCLK,R4             ; PREPARE TO WRITE DIAGNOSTIC REGISTER
                          0000'C2   54      D0  019D    578          MOVL    R4,DR(R2)                ; WRITE DIAGNOSTIC REGISTER
                     54   00000000'8F      C8  01A2    579          BISL    #MASS_FAIL!MASS_WCLK,R4  ; EXPECT MASS FAIL AND W CLOCK
                          00000000'8F      CB  01A9    580          BICL3   #MASS_CTOD!BYTE1!BYTE0,-
                     53   0000'C2              01AF    581                  DR(R2),R3                ; READ DIAG REGISTER W/O MBDIB FIELD
                          54   53     D1  01B3    582          CMPL    R3,R4                    ; CHECK
                               1B     13  01B6    583          BEQL    30$                      ; BRANCH IF SIM_SCLK IS SET
                                          01B8    584          $DS_ERRHARD_S   #1,LUN,-         ; PRINT SIM SCLK ERROR
                                          01B8    585                          ARRAY_MSG21,-
                                          01B8    586                          PRINT_SBE
                      00000000'EF      DF  01B8                         PUSHAL  PRINT_SBE
                      00000000'EF      DF  01BE                         PUSHAL  ARRAY_MSG21
                      00000000'EF      DD  01C4                         PUSHL   LUN
                                01     DD  01CA                         PUSHL   #1
                00000000'9F   04      FB  01CC                         CALLS   #$$M, a#DS$ERRHARD
                                          01D3    587 30$:     $DS_CKLOOP       20$              ; SCOPE LOOP?
                  00000000'9F   AE AF     FA  01D3                         CALLG   20$, a#DS$CKLOOP
                0000'C2   00000000'8F      C8  01DB    588 50$:     BISL    #PGM_INIT,CR(R2)         ; INIT FOR SCOPE LOOP
                0000'C2   00000000'8F      C8  01E4    589          BISL    #MAINT_MODE,CR(R2)       ; RETURN TO MAINTENANCE MODE
                     54   00000000'8F      D0  01ED    590          MOVL    #INVRT_MB_DPAR,R4        ; SET UP EXPECTED VALUE REGISTER
                          0000'C2   54      D0  01F4    591          MOVL    R4,DR(R2)                ; WRITE DIAGNOSTIC REGISTER
                     54   00000000'8F      C8  01F9    592          BISL    #MASS_FAIL,R4            ; EXPECT MASS FAIL
                          00000000'8F      CB  0200    593          BICL3   #MASS_CTOD!BYTE1!BYTE0,-
                     53   0000'C2              0206    594                  DR(R2),R3                ; READ DIAGNOSTIC REGISTER
                          54   53     D1  020A    595          CMPL    R3,R4                    ; CHECK
                               1B     13  020D    596          BEQL    60$                      ; BRANCH IF INVERT MASS BUS DATA
                                          020F    597                                           ; PARITY BIT IS SET
                                          020F    598          $DS_ERRHARD_S   #2,LUN,-         ; PRINT ERROR
                                          020F    599                          ARRAY_MSG45,-
```

```
                00000000'EF   DF  020F                        PUSHAL  PRINT_SBE
                00000000'EF   DF  0215                        PUSHAL  ARRAY_MSG45
                00000000'EF   DD  021B                        PUSHL   LUN
                          02  DD  0221                        PUSHL   #2
           00000000'9F    04  FB  0223                        CALLS   #$$M, a#DS$ERRHARD
                              022A  601 60$:    $DS_CKLOOP      50$                     ; SCOPE LOOP?
     00000000'9F    AE AF   FA  022A                        CALLG   50$, a#DS$CKLOOP
0000'C2    00000000'8F   C8  0232  602 70$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F   C8  023B  603         BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
     54    00000000'8F   D0  0244  604         MOVL    #INVRT_MB_CPAR,R4       ; SET UP EXPECTED VALUE REGISTER
          0000'C2    54   D0  024B  605         MOVL    R4,DR(R2)               ; WRITE INVERT CONTROL PARITY IN DR
     54    00000000'8F   C8  0250  606         BISL    #MASS_FAIL,R4           ; EXPECT MASS FAIL SET
          00000000'8F   CB  0257  607         BICL3   #MASS_CTOD!BYTE1!BYTE0,-
         53    0000'C2         025D  608                 DR(R2),R3               ; READ DIAGNOSTIC REGISTER
              54    53   D1  0261  609         CMPL    R3,R4                   ; CHECK
                    1B   13  0264  610         BEQL    80$                     ; BRANCH IF INVERT MASS BUS CONTROL
                              0266  611                                         ; PARITY IS SET
                              0266  612         $DS_ERRHARD_S   #3,LUN,-         ; PRINT ERROR
                              0266  613                         ARRAY_MSG44,-
                              0266  614                         PRINT_SBE
                00000000'EF   DF  0266                        PUSHAL  PRINT_SBE
                00000000'EF   DF  026C                        PUSHAL  ARRAY_MSG44
                00000000'EF   DD  0272                        PUSHL   LUN
                          03  DD  0278                        PUSHL   #3
           00000000'9F    04  FB  027A                        CALLS   #$$M, a#DS$ERRHARD
                              0281  615 80$:    $DS_CKLOOP      70$                     ; SCOPE LOOP?
     00000000'9F    AE AF   FA  0281                        CALLG   70$, a#DS$CKLOOP
0000'C2    00000000'8F   C8  0289  616 90$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F   C8  0292  617         BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
     54    00000000'8F   D0  029B  618         MOVL    #INVRT_MAP_PAR,R4       ; EXPECT INVERT MAP PARITY
          0000'C2    54   D0  02A2  619         MOVL    R4,DR(R2)               ; WRITE DIAGNOSTIC REGISTER
     54    00000000'8F   C8  02A7  620         BISL    #MASS_FAIL,R4           ; EXPECT MASS FAIL
          00000000'8F   CB  02AE  621         BICL3   #MASS_CTOD!BYTE1!BYTE0,-
         53    0000'C2         02B4  622                 DR(R2),R3               ; READ DIAGNOSTIC REGISTER
              54    53   D1  02B8  623         CMPL    R3,R4                   ; CHECK
                    1B   13  02BB  624         BEQL    100$                    ; BRANCH IF INVERT MAP PARITY IS SET
                              02BD  625         $DS_ERRHARD_S   #4,LUN,-         ; PRINT ERROR
                              02BD  626                         ARRAY_MSG43,-
                              02BD  627                         PRINT_SBE
                00000000'EF   DF  02BD                        PUSHAL  PRINT_SBE
                00000000'EF   DF  02C3                        PUSHAL  ARRAY_MSG43
                00000000'EF   DD  02C9                        PUSHL   LUN
                          04  DD  02CF                        PUSHL   #4
           00000000'9F    04  FB  02D1                        CALLS   #$$M, a#DS$ERRHARD
                              02D8  628 100$:    $DS_CKLOOP      90$                     ; SCOPE LOOP?
     00000000'9F    AE AF   FA  02D8                        CALLG   90$, a#DS$CKLOOP
0000'C2    00000000'8F   C8  02E0  629 110$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F   C8  02E9  630         BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
     54    00000000'8F   D0  02F2  631         MOVL    #BLKSND_COMD,R4         ; EXPECT BLOCK SENDING COMMAND BIT
          0000'C2    54   D0  02F9  632         MOVL    R4,DR(R2)               ; WRITE DIAGNOSTIC REGISTER
     54    00000000'8F   C8  02FE  633         BISL    #MASS_FAIL,R4           ; EXPECT MASS FAIL
          00000000'8F   CB  0305  634         BICL3   #MASS_CTOD!BYTE1!BYTE0,-
         53    0000'C2         030B  635                 DR(R2),R3               ; READ DIAGNOSTIC REG
              54    53   D1  030F  636         CMPL    R3,R4                   ; CHECK
                    1B   13  0312  637         BEQL    120$                    ; BRANCH IF BLOCK SENDING COMMAND IS 1
                              0314  638         $DS_ERRHARD_S   #5,LUN,-         ; PRINT ERROR
                              0314  639                         ARRAY_MSG42,-
```

```
                                0314   640                      PRINT_SBE
          00000000'EF   DF      0314                    PUSHAL   PRINT_SBE
          00000000'EF   DF      031A                    PUSHAL   ARRAY_MSG42
          00000000'EF   DD      0320                    PUSHL    LUN
                  05    DD      0326                    PUSHL    #5
      00000000'9F   04    FB    0328                    CALLS    #$$M, a#DS$ERRHARD
                                032F   641 120$:  $DS_CKLOOP      110$                ; SCOPE LOOP?
  00000000'9F   AE AF     FA    032F                    CALLG    110$, a#DS$CKLOOP
0000'C2   00000000'8F     C8    0337   642 150$:  BISL     #PGM_INIT,CR(R2)          ; INIT FOR SCOPE LOOP
0000'C2   00000000'8F     C8    0340   643        BISL     #MAINT_MODE,CR(R2)        ; RETURN TO MAINTENANCE MODE
    54    00000000'8F     D0    0349   644        MOVL     #SIM_EBL,R4               ; EXPECT SIM_EBL BIT
        0000'C2   54      D0    0350   645        MOVL     R4,DR(R2)                 ; WRITE DIAGNOSTIC REGISTER
    54    00000000'8F     C8    0355   646        BISL     #MASS_FAIL,R4             ; EXPECT MASS FAIL
          00000000'8F     CB    035C   647        BICL3    #MASS_CTOD!BYTE1!BYTE0,-
       53   0000'C2               0362   648                 DR(R2),R3              ; READ DIAGNOSTIC REG
            54   53     D1    0366   649        CMPL     R3,R4
                 1B     13    0369   650        BEQL     160$                      ; BRANCH IF SIM_EBL IS SET
                               036B   651        $DS_ERRHARD_S    #6,LUN,-         ; PRINT ERROR
                               036B   652                         ARRAY_MSG10,-
                               036B   653                         PRINT_SBE
          00000000'EF   DF      036B                    PUSHAL   PRINT_SBE
          00000000'EF   DF      0371                    PUSHAL   ARRAY_MSG10
          00000000'EF   DD      0377                    PUSHL    LUN
                  06    DD      037D                    PUSHL    #6
      00000000'9F   04    FB    037F                    CALLS    #$$M, a#DS$ERRHARD
                                0386   654 160$:  $DS_CKLOOP      150$                ; SCOPE LOOP?
  00000000'9F   AE AF     FA    0386                    CALLG    150$, a#DS$CKLOOP
0000'C2   00000000'8F     C8    038E   655 170$:  BISL     #PGM_INIT,CR(R2)          ; INIT FOR SCOPE LOOP
0000'C2   00000000'8F     C8    0397   656        BISL     #MAINT_MODE,CR(R2)        ; RETURN TO MAINTENANCE MODE
    54    00000000'8F     D0    03A0   657        MOVL     #SIM_OCC,R4               ; EXPECT SIM_OCC BIT SET
        0000'C2   54      D0    03A7   658        MOVL     R4,DR(R2)                 ; WRITE DIAGNOSTIC REGISTER
    54    00000000'8F     C8    03AC   659        BISL     #MASS_FAIL,R4             ; EXPECT MASS FAIL
          00000000'8F     CB    03B3   660        BICL3    #MASS_CTOD!BYTE1!BYTE0,-
       53   0000'C2               03B9   661                 DR(R2),R3              ; READ DIAGNOSTIC REGISTER
            54   53     D1    03BD   662        CMPL     R3,R4                     ; CHECK
                 1B     13    03C0   663        BEQL     180$                      ; BRANCH IF SIM_OCCUPIED IS SET
                               03C2   664        $DS_ERRHARD_S    #7,LUN,-         ; PRINT ERROR
                               03C2   665                         ARRAY_MSG9,-
                               03C2   666                         PRINT_SBE
          00000000'EF   DF      03C2                    PUSHAL   PRINT_SBE
          00000000'EF   DF      03C8                    PUSHAL   ARRAY_MSG9
          00000000'EF   DD      03CE                    PUSHL    LUN
                  07    DD      03D4                    PUSHL    #7
      00000000'9F   04    FB    03D6                    CALLS    #$$M, a#DS$ERRHARD
                                03DD   667 180$:  $DS_CKLOOP      170$                ; SCOPE LOOP?
  00000000'9F   AE AF     FA    03DD                    CALLG    170$, a#DS$CKLOOP
0000'C2   00000000'8F     C8    03E5   668 190$:  BISL     #PGM_INIT,CR(R2)          ; INIT FOR SCOPE LOOP
0000'C2   00000000'8F     C8    03EE   669        BISL     #MAINT_MODE,CR(R2)        ; RETURN TO MAINTENANCE MODE
    54    00000000'8F     D0    03F7   670        MOVL     #SIM_ATTN,R4              ; EXPECT SIM ATTN BIT SET
        0000'C2   54      D0    03FE   671        MOVL     R4,DR(R2)                 ; WRITE SIM_ATTN IN DIAG REGISTER
    54    00000000'8F     C8    0403   672        BISL     #MASS_FAIL,R4             ; MERGE MASS FAIL INTO EXPECTED VALUE
          00000000'8F     CB    040A   673        BICL3    #MASS_CTOD!BYTE1!BYTE0,-
       53   0000'C2               0410   674                 DR(R2),R3              ; READ DIAGNOSTIC REGISTER
            54   53     D1    0414   675        CMPL     R3,R4                     ; CHECK
                 1B     13    0417   676        BEQL     200$                      ; BRANCH IF SIMULATE ATTENTION IS SET
                               0419   677        $DS_ERRHARD_S    #8,LUN,-         ; PRINT ERROR
                               0419   678                         ARRAY_MSG9,-
```

```
                                0419   679                          PRINT_SBE
                00000000'EF     DF  0419                     PUSHAL  PRINT_SBE
                00000000'EF     DF  041F                     PUSHAL  ARRAY_MSG9
                00000000'EF     DD  0425                     PUSHL   LUN
                          08    DD  042B                     PUSHL   #8
           00000000'9F    04    FB  042D                     CALLS   #$$M, a#DS$ERRHARD
                                0434   680 200$:    $DS_CKLOOP      190$                ; SCOPE LOOP?
     00000000'9F    AE AF     FA  0434                     CALLG   190$, a#DS$CKLOOP
0000'C2    00000000'8F     C8  043C   681 210$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F     C8  0445   682          BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
     54    00000000'8F     D0  044E   683          MOVL    #MBDIB_SEL,R4           ; EXPECT MBDIB_SEL SET
          0000'C2    54    D0  0455   684          MOVL    R4,DR(R2)               ; WRITE MBDIB_SEL INTO DIAG REGISTER
     54    00000000'8F     C8  045A   685          BISL    #MASS_FAIL,R4           ; EXPECT MASS FAIL
          00000000'8F     CB  0461   686          BICL3   #MASS_CTOD!BYTE1!BYTE0,-
         53    0000'C2             0467   687                  DR(R2),R3               ; READ DIAGNOSTIC REG
              54    53    D1  046B   688          CMPL    R3,R4                   ; CHECK
                    1B    13  046E   689          BEQL    220$                    ; BRANCH IF MBDIB SELECT = 1
                                0470   690          $DS_ERRHARD_S   #9,LUN,-        ; PRINT ERROR
                                0470   691                          ARRAY_MSG39,-
                                0470   692                          PRINT_SBE
                00000000'EF     DF  0470                     PUSHAL  PRINT_SBE
                00000000'EF     DF  0476                     PUSHAL  ARRAY_MSG39
                00000000'EF     DD  047C                     PUSHL   LUN
                          09    DD  0482                     PUSHL   #9
           00000000'9F    04    FB  0484                     CALLS   #$$M, a#DS$ERRHARD
                                048B   693 220$:    $DS_CKLOOP      210$                ; SCOPE LOOP?
     00000000'9F    AE AF     FA  048B                     CALLG   210$, a#DS$CKLOOP
0000'C2    00000000'8F     C8  0493   694 230$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F     C8  049C   695          BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
          0000'C2    D4  04A5   696          CLRL    DR(R2)                  ; CLEAR ANY RESIDUAL ONES IN THE DR
     56    00000000'EF     D0  04A9   697          MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
     66    00000000'8F     D0  04B0   698          MOVL    #NOOP,(R6)              ; ISSUES NOOP
               53    66    D0  04B7   699          MOVL    (R6),R3                 ; READ CONTROL PATH
53   0000'C2    00000000'8F     CB  04BA   700          BICL3   #BYTE1!BYTE0,DR(R2),R3  ; READ DIAGNOSTIC REG
     54    00000000'8F     D0  04C4   701          MOVL    #MASS_FAIL,R4           ; EXPECT MASS FAIL ONLY
               54    53    D1  04CB   702          CMPL    R3,R4                   ; DID CTOD GO TO ZERO?
                    1B    13  04CE   703          BEQL    240$                    ; BRANCH IF OK
                                04D0   704          $DS_ERRHARD_S   #10,LUN,-        ; PRINT ERROR
                                04D0   705                          ARRAY_MSG33,-
                                04D0   706                          PRINT_SBE
                00000000'EF     DF  04D0                     PUSHAL  PRINT_SBE
                00000000'EF     DF  04D6                     PUSHAL  ARRAY_MSG33
                00000000'EF     DD  04DC                     PUSHL   LUN
                          0A    DD  04E2                     PUSHL   #10
           00000000'9F    04    FB  04E4                     CALLS   #$$M, a#DS$ERRHARD
                                04EB   707 240$:    $DS_CKLOOP      230$                ; SCOPE LOOP?
     00000000'9F    A5 AF     FA  04EB                     CALLG   230$, a#DS$CKLOOP
0000'C2    00000000'8F     C8  04F3   708 250$:    BISL    #PGM_INIT,CR(R2)        ; INIT FOR SCOPE LOOP
0000'C2    00000000'8F     C8  04FC   709          BISL    #MAINT_MODE,CR(R2)      ; RETURN TO MAINTENANCE MODE
          0000'C2    D4  0505   710          CLRL    DR(R2)                  ; CLEAR DIAGNOSTIC REGISTER
     56    00000000'EF     D0  0509   711          MOVL    DRIVE0,R6               ; PUT ADDRESS OF DRIVE 0 IN R6
     66    00000000'8F     D0  0510   712          MOVL    #NOOP,(R6)              ; THIS SHOULD ASSERT CTOD
     54    00000000'8F     D0  0517   713          MOVL    #MASS_CTOD!MASS_FAIL,R4 ; SET UP EXPECTED RESULTS
53   0000'C2    00000000'8F     CB  051E   714          BICL3   #BYTE1!BYTE0,DR(R2),R3  ; READ DIAGNOSTIC REGISTER
               54    53    D1  0528   715          CMPL    R3,R4                   ; CHECK
                    1B    13  052B   716          BEQL    260$                    ; BRANCH IF CTOD IS SET
                                052D   717          $DS_ERRHARD_S   #11,LUN,-        ; PRINT ERROR
```

```
                                  052D   718                          ARRAY_MSG33,-
                                  052D   719                          PRINT_SBE
              00000000'EF    DF   052D                     PUSHAL   PRINT_SBE
              00000000'EF    DF   0533                     PUSHAL   ARRAY_MSG33
              00000000'EF    DD   0539                     PUSHL    LUN
                        0B   DD   053F                     PUSHL    #11
        00000000'9F     04   FB   0541                     CALLS    #$$M, a#DS$ERRHARD
                                  0548   720 260$:  $DS_CKLOOP      250$                ; SCOPE LOOP?
     00000000'9F    A8 AF    FA   0548                     CALLG    250$, a#DS$CKLOOP
   0000'C2   00000000'8F    C8   0550   721         BISL     #PGM_INIT,CR(R2)           ; CLEAN UP
                        58   D4   0559   722 310$:  CLRL     R8                         ; CLEAR DRIVE SELECT INDEX
      59   00000000'EF48    D0   055B   723 320$:  MOVL     DRIVE_ADR_TBL R8 ,R9       ; GET INDEXED DRIVE ADDRESS
                        5A   D4   0563   724 330$:  CLRL     R10                        ; CLEAR DRIVE REGISTER SELECT INDEX
   0000'C2   00000000'8F    C8   0565   725 340$:  BISL     #PGM_INIT,CR(R2)           ; INIT
   0000'C2   00000000'8F    C8   056E   726         BISL     #MAINT_MODE,CR(R2)         ; PUT RH750 INTO MAINTENANCE MODE
            5B    58   0D   78   0577   727         ASHL     #13,R8,R11                 ; PUT DRIVE # INTO DRIVE SELECT FIELD
            50    5A   08   78   057B   728         ASHL     #8,R10,R0                  ; PUT REG SELECTED # INTO RS FIELD
                  5B   50   C8   057F   729         BISL     R0,R11                     ; MERGE RS INTO DS FIELD
              00000000'8F    C9   0582   730         BISL3    #MASS_FAIL!MASS_CTOD,-     ; EXPECT MASS FAIL, MASS CTOD
                  54   5B        0588   731                  R11,R4
                     694A   D4   058A   732         CLRL     (R9) R10                   ; SELECT EXTERNAL DRIVE REGISTER WHILE
                                  058D   733                                             ; IN MAINTENANCE MODE
                                  058D   734                                             ; DS AND RS FIELD OF DR SHOULD REFLECT
                                  058D   735                                             ; DRIVE # IN R8 AND REGISTER # IN R10
 53   0000'C2   00000000'8F    CB   058D   736         BICL3    #BYTE0,DR(R2),R3           ; READ DIAG REGISTER W/O MBDIB FIELD
                  54   53   D1   0597   737         CMPL     R3,R4                      ; IS REGISTER CORRECT?
                        1B   13   059A   738         BEQL     350$                       ; BRANCH IF OK
                                  059C   739         $DS_ERRHARD_S    #12,LUN,-          ; PRINT ERROR
                                  059C   740                          ARRAY_MSG20,-
                                  059C   741                          PRINT_SBE
              00000000'EF    DF   059C                     PUSHAL   PRINT_SBE
              00000000'EF    DF   05A2                     PUSHAL   ARRAY_MSG20
              00000000'EF    DD   05A8                     PUSHL    LUN
                        0C   DD   05AE                     PUSHL    #12
        00000000'9F     04   FB   05B0                     CALLS    #$$M, a#DS$ERRHARD
                                  05B7   742 350$:  $DS_CKLOOP      340$                ; SCOPE LOOP?
     00000000'9F    AB AF    FA   05B7                     CALLG    340$, a#DS$CKLOOP
              A2 5A    0F   F3   05BF   743         AOBLEQ   #15,R10,340$               ; INDEX REGISTER SELECT
              94 58    07   F3   05C3   744         AOBLEQ   #7,R8,320$                 ; INDEX DRIVE SELECT
   0000'C2   00000000'8F    C8   05C7   745 360$:  BISL     #PGM_INIT,CR(R2)           ; INIT RH750
   0000'C2   00000000'8F    C8   05D0   746         BISL     #MAINT_MODE,CR(R2)         ; RETURN TO MAINT MODE
        54   00000000'8F    D0   05D9   747         MOVL     #INVRT_SILO_PAR,R4         ; EXPECT ISPG SET
            0000'C2    54   D0   05E0   748         MOVL     R4,DR(R2)                  ; WRITE ISPG INTO DR
        54   00000000'8F    C8   05E5   749         BISL     #MASS_FAIL,R4              ; EXPECT MASS FAIL
              00000000'8F    CB   05EC   750         BICL3    #MASS_CTOD!BYTE0!BYTE1,-
            53    0000'C2        05F2   751                  DR(R2),R3                  ; READ DIAGNOSTIC REGISTER
                  54   53   D1   05F6   752         CMPL     R3,R4                      ; CHECK FOR ISPG
                        1B   13   05F9   753         BEQL     370$                       ; BRANCH IF ISPG = 1
                                  05FB   754         $DS_ERRHARD_S    #13,LUN,-
                                  05FB   755                          ARRAY_MSG40,-
                                  05FB   756                          PRINT_SBE          ; REPORT ERROR
              00000000'EF    DF   05FB                     PUSHAL   PRINT_SBE
              00000000'EF    DF   0601                     PUSHAL   ARRAY_MSG40
              00000000'EF    DD   0607                     PUSHL    LUN
                        0D   DD   060D                     PUSHL    #13
        00000000'9F     04   FB   060F                     CALLS    #$$M, a#DS$ERRHARD
                                  0616   757 370$:  $DS_CKLOOP      360$                ; SCOPE LOOP?
```

```
      00000000'9F    AE AF   FA  0616                          CALLG   360$, a#DS$CKLOOP
    0000'C2   00000000'8F   C8  061E   758 380$:   BISL    #PGM_INIT,CR(R2)          ; INIT RH750
    0000'C2   00000000'8F   C8  0627   759         BISL    #MAINT_MODE,CR(R2)        ; RETURN TO MAINT MODE
         54   00000000'8F   D0  0630   760         MOVL    #SIM_EXCEPT,R4            ; EXPECT SIMULATE EXCEPTION
             0000'C2   54   D0  0637   761         MOVL    R4,DR(R2)                 ; WRITE SIM_EXCEPT INTO DR
         54   00000000'8F   C8  063C   762         BISL    #MASS_FAIL,R4             ; EXPECT MASS FAIL
              00000000'8F   CB  0643   763         BICL3   #NIBBLE4!BYTE0!BYTE1,-
             53   0000'C2       0649   764                 DR(R2),R3                 ; READ DIAGNOSTIC REGISTER
                  54   53   D1  064D   765         CMPL    R3,R4                     ; CHECK FOR SIM_EXCEPT
                       1B   13  0650   766         BEQL    390$                      ; BRANCH IF SIM_EXCEPT = 1
                                0652   767         $DS_ERRHARD_S   #14,LUN,-
                                0652   768                         ARRAY_MSG8,-
                                0652   769                         PRINT_SBE         ; REPORT ERROR
              00000000'EF   DF  0652                          PUSHAL  PRINT_SBE
              00000000'EF   DF  0658                          PUSHAL  ARRAY_MSG8
              00000000'EF   DD  065E                          PUSHL   LUN
                       0E   DD  0664                          PUSHL   #14
         00000000'9F   04   FB  0666                          CALLS   #$$M, a#DS$ERRHARD
                                066D   770 390$:   $DS_CKLOOP      380$              ; SCOPE LOOP?
      00000000'9F    AE AF   FA  066D                          CALLG   380$, a#DS$CKLOOP
    0000'C2   00000000'8F   C8  0675   771         BISL    #SIM_EBL,DR(R2)           ; SET SIM EBL
         1B 0000'C2   07   E0  067E   772         BBS     #7,SR(R2),400$            ; CHECK FOR MASS EXCEPTION SET, SR<07>
                                0684   773         $DS_ERRHARD_S   #15,LUN,-
                                0684   774                         ARRAY_MSG7,-
                                0684   775                         PRINT_SBE         ; PRINT ERROR
              00000000'EF   DF  0684                          PUSHAL  PRINT_SBE
              00000000'EF   DF  068A                          PUSHAL  ARRAY_MSG7
              00000000'EF   DD  0690                          PUSHL   LUN
                       0F   DD  0696                          PUSHL   #15
         00000000'9F   C4   FB  0698                          CALLS   #$$M, a#DS$ERRHARD
                                069F   776 400$:   $DS_CKLOOP      380$              ; SCOPE LOOP?
    00000000'9F    FF7B CF   FA  069F                          CALLG   380$, a#DS$CKLOOP
                                06A8   777         $DS_ENDSUB
                                06A8       T4_S4_X:
00000000'9F   00000078'EF   FA  06A8                          CALLG   $S$, a#DS$ENDSUB
```

```
                                  06B3   779 $DS_ENDTEST
                  50   01   D0    06B3                   MOVL    #1, R0          ; NORMAL EXIT
                                  06B6       TEST_004_X::
        00000000'9F   6E   FA    06B6                   CALLG   (SP), a#DS$BREAK
                             04   06BD                   RET                     ; RETURN TO TEST SEQUENCER
                                  06BE   780 $DS_PAGE
                                  06BE             .SBTTL  TEST 5:  VIRTUAL ADDRESS REGISTER SA0 AND LOADING TEST
                             00000000                   .PSECT  TEST_005, PAGE, NOWRT
                                  0030
                                  0030   782 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  0030       DATA_005:
                        00000000  0030                   .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  0034       TEST_005::
                            0000  0034                   .WORD    M<>            ; ENTRY MASK
                                  0036   783 ;++
                                  0036   784 ; TEST DESCRIPTION:
                                  0036   785 ;
                                  0036   786 ;       THIS TEST CHECKS FOR STUCK AT ZERO BITS IN THE VIRTUAL ADDRESS
                                  0036   787 ;       REGISTER, AND FOR PROPER LOADING OF THE REGISTER.
                                  0036   788 ;
                                  0036   789 ; TEST ALGORITHM: SUBTEST 1
                                  0036   790 ;
                                  0036   791 ;       FLOAT A 1 FROM BIT 0 THROUGH BIT 20
                                  0036   792 ;
                                  0036   793 ;
                                  0036   794 ; TEST ALGORITHM: SUBTEST 2
                                  0036   795 ;
                                  0036   796 ;       VARIOUS PATTERNS ARE WRITTEN TO THE VIRTUAL ADDRESS REGISTER,
                                  0036   797 ;       AND EACH TIME THE VAR IS CHECKED TO SEE IF IT HAS BEEN PROPERLY
                                  0036   798 ;       LOADED.
                                  0036   799 ;--
                                  0036   800
                                  0036   801         $DS_BGNSUB
                                  0036       T5_S1::
00000000'9F   00000084'EF   FA    0036                   CALLG   $S$, a#DS$BGNSUB
         52   00000000'EF   D0    0041   802         MOVL    RH_CUR_ADR,R2           ; MOVE ADDRESS OF MUT TO R2
    0000'C2   00000000'8F   C8    0048   803         BISL    #PGM_INIT,CR(R2)        ; INITIALIZE THE MUT
                                  0051   804         ERRPREP RHVAR_MSG,3,FMT_ANY_REG ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE    0051                   MOVAL   RHVAR_MSG,REG_NAME
         00000000'EF   03   9A    005C                   MOVZBL  #3,REG_NO
00000000'EF   00000000'EF   DE    0063                   MOVAL   FMT_ANY_REG,REG_STRING
                        5A   D4    006E   805         CLRL    R10                     ; CLEAR BIT SHIFT COUNT
                        54   D4    0070   806         CLRL    R4                      ; SET UP EXPECTED VALUE
                  54   01   88    0072   807         BISB    #1,R4                   ; INITIALLY SET TO 1
                        04   11    0075   808         BRB     20$                     ; SKIP ROTATE FOR FIRST PASS
             54   54   01   9C    0077   809 10$:    ROTL    #1,R4,R4                ; ROTATE BIT IN R4 LEFT ONE PLACE
             0000'C2   54   D0    007B   810 20$:    MOVL    R4,VAR(R2)              ; WRITE BIT INTO VIRTUAL ADDRESS REG
             53   0000'C2   D0    0080   811         MOVL    VAR(R2),R3              ; R3 = VIRTUAL ADDRESS REGISTER
                  54   53   D1    0085   812         CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1E   13    0088   813         BEQL    30$                     ; IF NO ERRORS SKIP REPORT
                  55   54   D0    008A   814         MOVL    R4,R5                   ; MOVE EXPECTED REG TO CONVERSION REG
                                  008D   815         $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                  008D   816                         ARRAY_MSG13,-
                                  008D   817                         PRINT_SBE
             00000000'EF   DF    008D                   PUSHAL  PRINT_SBE
             00000000'EF   DF    0093                   PUSHAL  ARRAY_MSG13
             00000000'EF   DD    0099                   PUSHL   LUN
```

```
                        01   DD  009F                          PUSHL   #1
          00000000'9F   04   FB  00A1                          CALLS   #$$M, a#DS$ERRHARD
                                 00A8   818 30$:    $DS_CKLOOP      20$                 ; SCOPE LOOP?
       00000000'9F   D0 AF   FA  00A8                          CALLG   20$, a#DS$CKLOOP
                C3 5A   14   F3  00B0   819         AOBLEQ  #20,R10,10$                 ; DONE FOR ALL 21 BIT POSITIONS?
                                 00B4   820         $DS_ENDSUB
                                 00B4       T5_S1_X:
   00000000'9F   00000084'EF   FA  00B4                          CALLG   $S$, a#DS$ENDSUB
                                 00BF   821

                                 00BF   822 ;
                                 00BF   823 ;       THIS SUBTEST CHECKS FOR PROPER LOADING OF THE VIRTUAL ADDRESS
                                 00BF   824 ;       REGISTER.
                                 00BF   825 ;
                                 00BF   826
                                 00BF   827         $DS_BGNSUB
                                 00BF       T5_S2::
   00000000'9F   00000090'EF   FA  00BF                          CALLG   $S$, a#DS$BGNSUB
                        55   D4  00CA   828         CLRL    R5                          ; CLEAR FAILING BIT VALUE
          52   00000000'EF   D0  00CC   829         MOVL    RH_CUR_ADR,R2               ; GET ADDRESS OF MUT
      0000'C2   00000000'8F   C8  00D3   830 10$:    BISL    #PGM_INIT,CR(R2)            ; INITIALIZE THE MUT
                                 00DC   831         ERRPREP RHVAR_MSG,3,FMT_ANY_REG     ; PREPARE TO HANDLE ERROR
   00000000'EF   00000000'EF   DE  00DC                          MOVAL   RHVAR_MSG,REG_NAME
          00000000'EF   03   9A  00E7                          MOVZBL  #3,REG_NO
   00000000'EF   00000000'EF   DE  00EE                          MOVAL   FMT_ANY_REG,REG_STRING
          54   00000000'8F   D0  00F9   832         MOVL    #VAR_PAT1,R4                ; EXPECTED DATA
              0000'C2   54   D0  0100   833         MOVL    R4,VAR(R2)                  ; PUT FIRST PATTERN INTO VAR
              53   0000'C2   D0  0105   834         MOVL    VAR(R2),R3                  ; READ BACK VAR
                   54   53   D1  010A   835         CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                        1B   13  010D   836         BEQL    20$                         ; BRANCH IF OKAY
                                 010F   837         $DS_ERRHARD_S   #1,LUN,-            ; PRINT ERROR
                                 010F   838                         ARRAY_MSG26,-
                                 010F   839                         PRINT_SBE
              00000000'EF   DF  010F                          PUSHAL  PRINT_SBE
              00000000'EF   DF  0115                          PUSHAL  ARRAY_MSG26
              00000000'EF   DD  011B                          PUSHL   LUN
                        01   DD  0121                          PUSHL   #1
          00000000'9F   04   FB  0123                          CALLS   #$$M, a#DS$ERRHARD
                                 012A   840 20$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
       00000000'9F   A6 AF   FA  012A                          CALLG   10$, a#DS$CKLOOP
          54   00000000'8F   D0  0132   841         MOVL    #VAR_PAT2,R4                ; EXPECTED DATA
              0000'C2   54   D0  0139   842         MOVL    R4,VAR(R2)                  ; WRITE SECOND PATTERN TO VAR
              53   0000'C2   D0  013E   843         MOVL    VAR(R2),R3                  ; READ BACK VAR
                   54   53   D1  0143   844         CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                        1B   13  0146   845         BEQL    30$                         ; BRANCH IF OKAY
                                 0148   846         $DS_ERRHARD_S   #2,LUN,-            ; PRINT ERROR
                                 0148   847                         ARRAY_MSG38,-
                                 0148   848                         PRINT_SBE
              00000000'EF   DF  0148                          PUSHAL  PRINT_SBE
              00000000'EF   DF  014E                          PUSHAL  ARRAY_MSG38
              00000000'EF   DD  0154                          PUSHL   LUN
                        02   DD  015A                          PUSHL   #2
          00000000'9F   04   FB  015C                          CALLS   #$$M, a#DS$ERRHARD
                                 0163   849 30$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
   00000000'9F   FF6C CF   FA  0163                          CALLG   10$, a#DS$CKLOOP
          54   00000000'8F   D0  016C   850         MOVL    #VAR_PAT3,R4                ; EXPECTED DATA
              0000'C2   54   D0  0173   851         MOVL    R4,VAR(R2)                  ; WRITE THIRD PATTERN TO VAR
```

```
53   0000'C2   D0  0178   852          MOVL    VAR(R2),R3                ; READ BACK VAR
```

```
                54   53   D1  017D   853          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                     1B   13  0180   854          BEQL    40$                     ; BRANCH IF OKAY
                          0182   855          $DS_ERRHARD_S   #3,LUN,-            ; PRINT ERROR
                          0182   856                          ARRAY_MSG26,-
                          0182   857                          PRINT_SBE
          00000000'EF       DF  0182                   PUSHAL  PRINT_SBE
          00000000'EF       DF  0188                   PUSHAL  ARRAY_MSG26
          00000000'EF       DD  018E                   PUSHL   LUN
                   03       DD  0194                   PUSHL   #3
     00000000'9F   04       FB  0196                   CALLS   #$$M, a#DS$ERRHARD
                          019D   858 40$:      $DS_CKLOOP      10$                 ; SCOPE LOOP?
00000000'9F   FF32 CF       FA  019D                   CALLG   10$, a#DS$CKLOOP
     54   00000000'8F       D0  01A6   859          MOVL    #VAR_PAT4,R4            ; EXPECTED DATA
          0000'C2   54       D0  01AD   860          MOVL    R4,VAR(R2)              ; WRITE FOURTH PATTERN TO VAR
          53   0000'C2       D0  01B2   861          MOVL    VAR(R2),R3              ; READ BACK VAR
                54   53       D1  01B7   862          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                     1B       13  01BA   863          BEQL    50$                     ; BRANCH IF OKAY
                          01BC   864          $DS_ERRHARD_S   #4,LUN,-            ; PRINT ERROR
                          01BC   865                          ARRAY_MSG41,-
                          01BC   866                          PRINT_SBE
          00000000'EF       DF  01BC                   PUSHAL  PRINT_SBE
          00000000'EF       DF  01C2                   PUSHAL  ARRAY_MSG41
          00000000'EF       DD  01C8                   PUSHL   LUN
                   04       DD  01CE                   PUSHL   #4
     00000000'9F   04       FB  01D0                   CALLS   #$$M, a#DS$ERRHARD
                          01D7   867 50$:      $DS_CKLOOP      10$                 ; SCOPE LOOP?
00000000'9F   FEF8 CF       FA  01D7                   CALLG   10$, a#DS$CKLOOP
     54   00000000'8F       D0  01E0   868          MOVL    #VAR_PAT5,R4            ; EXPECTED DATA
          0000'C2   54       D0  01E7   869          MOVL    R4,VAR(R2)              ; WRITE FIFTH PATTERN TO VAR
          53   0000'C2       D0  01EC   870          MOVL    VAR(R2),R3              ; READ BACK VAR
                54   53       D1  01F1   871          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                     1B       13  01F4   872          BEQL    60$                     ; BRANCH IF OKAY
                          01F6   873          $DS_ERRHARD_S   #5,LUN,-            ; PRINT ERROR
                          01F6   874                          ARRAY_MSG26,-
                          01F6   875                          PRINT_SBE
          00000000'EF       DF  01F6                   PUSHAL  PRINT_SBE
          00000000'EF       DF  01FC                   PUSHAL  ARRAY_MSG26
          00000000'EF       DD  0202                   PUSHL   LUN
                   05       DD  0208                   PUSHL   #5
     00000000'9F   04       FB  020A                   CALLS   #$$M, a#DS$ERRHARD
                          0211   876 60$:      $DS_CKLOOP      10$                 ; SCOPE LOOP?
00000000'9F   FEBE CF       FA  0211                   CALLG   10$, a#DS$CKLOOP
     54   00000000'8F       D0  021A   877          MOVL    #VAR_PAT6,R4            ; EXPECTED DATA
          0000'C2   54       D0  0221   878          MOVL    R4,VAR(R2)              ; WRITE SIXTH PATTERN TO VAR
          53   0000'C2       D0  0226   879          MOVL    VAR(R2),R3              ; READ BACK VAR
                54   53       D1  022B   880          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                     1B       13  022E   881          BEQL    70$                     ; BRANCH IF OKAY
                          0230   882          $DS_ERRHARD_S   #6,LUN,-            ; PRINT ERROR
                          0230   883                          ARRAY_MSG9,-
                          0230   884                          PRINT_SBE
          00000000'EF       DF  0230                   PUSHAL  PRINT_SBE
          00000000'EF       DF  0236                   PUSHAL  ARRAY_MSG9
          00000000'EF       DD  023C                   PUSHL   LUN
                   06       DD  0242                   PUSHL   #6
     00000000'9F   04       FB  0244                   CALLS   #$$M, a#DS$ERRHARD
                          024B   885 70$:      $DS_CKLOOP      10$                 ; SCOPE LOOP?
00000000'9F   FE84 CF       FA  024B                   CALLG   10$, a#DS$CKLOOP
```

```
                54   00000000'8F   D0   0254    886          MOVL    #VAR_PAT7,R4                ; EXPECTED DATA
                   0000'C2   54    D0   025B    887          MOVL    R4,VAR(R2)                  ; WRITE SEVENTH PATTERN TO VAR
                   53   0000'C2    D0   0260    888          MOVL    VAR(R2),R3                  ; READ BACK VAR
                        54   53    D1   0265    889          CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                             1B    13   0268    890          BEQL    80$                         ; BRANCH IF OKAY
                                        026A    891          $DS_ERRHARD_S   #7,LUN,-            ; PRINT ERROR
                                        026A    892                          ARRAY_MSG27,-
                                        026A    893                          PRINT_SBE
                    00000000'EF    DF   026A                               PUSHAL  PRINT_SBE
                    00000000'EF    DF   0270                               PUSHAL  ARRAY_MSG27
                    00000000'EF    DD   0276                               PUSHL   LUN
                              07   DD   027C                               PUSHL   #7
                00000000'9F   04    FB   027E                               CALLS   #$$M, a#DS$ERRHARD
                                        0285    894 80$:     $DS_CKLOOP      10$                 ; SCOPE LOOP?
           00000000'9F   FE4A CF    FA   0285                               CALLG   10$, a#DS$CKLOOP
                54   00000000'8F   D0   028E    895          MOVL    #VAR_PAT8,R4                ; EXPECTED DATA
                   0000'C2   54    D0   0295    896          MOVL    R4,VAR(R2)                  ; WRITE LAST PATTERN TO VAR
                   53   0000'C2    D0   029A    897          MOVL    VAR(R2),R3                  ; READ BACK VAR
                        54   53    D1   029F    898          CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                             1B    13   02A2    899          BEQL    90$                         ; BRANCH IF OKAY
                                        02A4    900          $DS_ERRHARD_S   #8,LUN,-            ; PRINT ERROR
                                        02A4    901                          ARRAY_MSG10,-
                                        02A4    902                          PRINT_SBE
                    00000000'EF    DF   02A4                               PUSHAL  PRINT_SBE
                    00000000'EF    DF   02AA                               PUSHAL  ARRAY_MSG10
                    00000000'EF    DD   02B0                               PUSHL   LUN
                              08   DD   02B6                               PUSHL   #8
                00000000'9F   04    FB   02B8                               CALLS   #$$M, a#DS$ERRHARD
                                        02BF    903 90$:     $DS_CKLOOP      10$                 ; SCOPE LOOP?
           00000000'9F   FE10 CF    FA   02BF                               CALLG   10$, a#DS$CKLOOP
                54   00000000'8F   D0   02C8    904          MOVL    #VAR_PAT9,R4                ; EXPECTED DATA
                   0000'C2   54    D0   02CF    905          MOVL    R4,VAR(R2)                  ; WRITE LAST PATTERN TO VAR
                   53   0000'C2    D0   02D4    906          MOVL    VAR(R2),R3                  ; READ BACK VAR
                        54   53    D1   02D9    907          CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                             1B    13   02DC    908          BEQL    100$                        ; BRANCH IF OKAY
                                        02DE    909          $DS_ERRHARD_S   #9,LUN,-            ; PRINT ERROR
                                        02DE    910                          ARRAY_MSG19,-
                                        02DE    911                          PRINT_SBE
                    00000000'EF    DF   02DE                               PUSHAL  PRINT_SBE
                    00000000'EF    DF   02E4                               PUSHAL  ARRAY_MSG19
                    00000000'EF    DD   02EA                               PUSHL   LUN
                              09   DD   02F0                               PUSHL   #9
                00000000'9F   04    FB   02F2                               CALLS   #$$M, a#DS$ERRHARD
                                        02F9    912 100$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
           00000000'9F   FDD6 CF    FA   02F9                               CALLG   10$, a#DS$CKLOOP
                54   00000000'8F   D0   0302    913          MOVL    #VAR_PAT10,R4               ; EXPECTED DATA
                   0000'C2   54    D0   0309    914          MOVL    R4,VAR(R2)                  ; WRITE LAST PATTERN TO VAR
                   53   0000'C2    D0   030E    915          MOVL    VAR(R2),R3                  ; READ BACK VAR
                        54   53    D1   0313    916          CMPL    R3,R4                       ; EXPECTED = RECEIVED?
                             1B    13   0316    917          BEQL    110$                        ; BRANCH IF OKAY
                                        0318    918          $DS_ERRHARD_S   #10,LUN,-           ; PRINT ERROR
                                        0318    919                          ARRAY_MSG10,-
                                        0318    920                          PRINT_SBE
                    00000000'EF    DF   0318                               PUSHAL  PRINT_SBE
                    00000000'EF    DF   031E                               PUSHAL  ARRAY_MSG10
                    00000000'EF    DD   0324                               PUSHL   LUN
                              0A   DD   032A                               PUSHL   #10
```

```
        00000000'9F    04    FB  032C                          CALLS   #$$M, a#DS$ERRHARD
                                 0333   921 110$:   $DS_CKLOOP      10$                ; SCOPE LOOP?
    00000000'9F    FD9C CF    FA  0333                          CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  033C   922         MOVL    #VAR_PAT11,R4              ; EXPECTED DATA
            0000'C2    54    D0  0343   923         MOVL    R4,VAR(R2)                 ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  0348   924         MOVL    VAR(R2),R3                 ; READ BACK VAR
                  54    53    D1  034D   925         CMPL    R3,R4                      ; EXPECTED = RECEIVED?
                        1B    13  0350   926         BEQL    120$                       ; BRANCH IF OKAY
                                 0352   927         $DS_ERRHARD_S   #11,LUN,-          ; PRINT ERROR
                                 0352   928                         ARRAY_MSG19,-
                                 0352   929                         PRINT_SBE
             00000000'EF    DF  0352                          PUSHAL  PRINT_SBE
             00000000'EF    DF  0358                          PUSHAL  ARRAY_MSG19
             00000000'EF    DD  035E                          PUSHL   LUN
                        0B    DD  0364                          PUSHL   #11
        00000000'9F    04    FB  0366                          CALLS   #$$M, a#DS$ERRHARD
                                 036D   930 120$:   $DS_CKLOOP      10$                ;"SCOPE LOOP?
    00000000'9F    FD62 CF    FA  036D                          CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  0376   931         MOVL    #VAR_PAT12,R4              ; EXPECTED DATA
            0000'C2    54    D0  037D   932         MOVL    R4,VAR(R2)                 ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  0382   933         MOVL    VAR(R2),R3                 ; READ BACK VAR
                  54    53    D1  0387   934         CMPL    R3,R4                      ; EXPECTED = RECEIVED?
                        1B    13  038A   935         BEQL    130$                       ; BRANCH IF OKAY
                                 038C   936         $DS_ERRHARD_S   #12,LUN,-          ; PRINT ERROR
                                 038C   937                         ARRAY_MSG11,-
                                 038C   938                         PRINT_SBE
             00000000'EF    DF  038C                          PUSHAL  PRINT_SBE
             00000000'EF    DF  0392                          PUSHAL  ARRAY_MSG11
             00000000'EF    DD  0398                          PUSHL   LUN
                        0C    DD  039E                          PUSHL   #12
        00000000'9F    04    FB  03A0                          CALLS   #$$M, a#DS$ERRHARD
                                 03A7   939 130$:   $DS_CKLOOP      10$                ; SCOPE LOOP?
    00000000'9F    FD28 CF    FA  03A7                          CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  03B0   940         MOVL    #VAR_PAT13,R4              ; EXPECTED DATA
            0000'C2    54    D0  03B7   941         MOVL    R4,VAR(R2)                 ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  03BC   942         MOVL    VAR(R2),R3                 ; READ BACK VAR
                  54    53    D1  03C1   943         CMPL    R3,R4                      ; EXPECTED = RECEIVED?
                        1B    13  03C4   944         BEQL    140$                       ; BRANCH IF OKAY
                                 03C6   945         $DS_ERRHARD_S   #13,LUN,-          ; PRINT ERROR
                                 03C6   946                         ARRAY_MSG19,-
                                 03C6   947                         PRINT_SBE
             00000000'EF    DF  03C6                          PUSHAL  PRINT_SBE
             00000000'EF    DF  03CC                          PUSHAL  ARRAY_MSG19
             00000000'EF    DD  03D2                          PUSHL   LUN
                        0D    DD  03D8                          PUSHL   #13
        00000000'9F    04    FB  03DA                          CALLS   #$$M, a#DS$ERRHARD
                                 03E1   948 140$:   $DS_CKLOOP      10$                ; SCOPE LOOP?
    00000000'9F    FCEE CF    FA  03E1                          CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  03EA   949         MOVL    #VAR_PAT14,R4              ; EXPECTED DATA
            0000'C2    54    D0  03F1   950         MOVL    R4,VAR(R2)                 ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  03F6   951         MOVL    VAR(R2),R3                 ; READ BACK VAR
                  54    53    D1  03FB   952         CMPL    R3,R4                      ; EXPECTED = RECEIVED?
                        1B    13  03FE   953         BEQL    150$                       ; BRANCH IF OKAY
                                 0400   954         $DS_ERRHARD_S   #14,LUN,-          ; PRINT ERROR
                                 0400   955                         ARRAY_MSG11,-
                                 0400   956                         PRINT_SBE
             00000000'EF    DF  0400                          PUSHAL  PRINT_SBE
```

```
                00000000'EF   DF  0406                      PUSHAL  ARRAY_MSG11
                00000000'EF   DD  040C                      PUSHL   LUN
                          0E  DD  0412                      PUSHL   #14
           00000000'9F    04  FB  0414                      CALLS   #$$M, a#DS$ERRHARD
                              041B   957 150$:   $DS_CKLOOP       10$                 ; SCOPE LOOP?
   00000000'9F    FCB4 CF    FA  041B                      CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  0424   958          MOVL    #VAR_PAT15,R4           ; EXPECTED DATA
            0000'C2    54    D0  042B   959          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  0430   960          MOVL    VAR(R2),R3              ; READ BACK VAR
                  54    53    D1  0435   961          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1B    13  0438   962          BEQL    160$                    ; BRANCH IF OKAY
                                  043A   963          $DS_ERRHARD_S   #15,LUN,-       ; PRINT ERROR
                                  043A   964                          ARRAY_MSG19,-
                                  043A   965                          PRINT_SBE
                00000000'EF   DF  043A                      PUSHAL  PRINT_SBE
                00000000'EF   DF  0440                      PUSHAL  ARRAY_MSG19
                00000000'EF   DD  0446                      PUSHL   LUN
                          0F  DD  044C                      PUSHL   #15
           00000000'9F    04  FB  044E                      CALLS   #$$M, a#DS$ERRHARD
                              0455   966 160$:   $DS_CKLOOP       10$                 ; SCOPE LOOP?
   00000000'9F    FC7A CF    FA  0455                      CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  045E   967          MOVL    #VAR_PAT16,R4           ; EXPECTED DATA
            0000'C2    54    D0  0465   968          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  046A   969          MOVL    VAR(R2),R3              ; READ BACK VAR
                  54    53    D1  046F   970          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1B    13  0472   971          BEQL    170$                    ; BRANCH IF OKAY
                                  0474   972          $DS_ERRHARD_S   #16,LUN,-       ; PRINT ERROR
                                  0474   973                          ARRAY_MSG12,-
                                  0474   974                          PRINT_SBE
                00000000'EF   DF  0474                      PUSHAL  PRINT_SBE
                00000000'EF   DF  047A                      PUSHAL  ARRAY_MSG12
                00000000'EF   DD  0480                      PUSHL   LUN
                          10  DD  0486                      PUSHL   #16
           00000000'9F    04  FB  0488                      CALLS   #$$M, a#DS$ERRHARD
                              048F   975 170$:   $DS_CKLOOP       10$                 ; SCOPE LOOP?
   00000000'9F    FC40 CF    FA  048F                      CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  0498   976          MOVL    #VAR_PAT17,R4           ; EXPECTED DATA
            0000'C2    54    D0  049F   977          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  04A4   978          MOVL    VAR(R2),R3              ; READ BACK VAR
                  54    53    D1  04A9   979          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1B    13  04AC   980          BEQL    180$                    ; BRANCH IF OKAY
                                  04AE   981          $DS_ERRHARD_S   #17,LUN,-       ; PRINT ERROR
                                  04AE   982                          ARRAY_MSG18,-
                                  04AE   983                          PRINT_SBE
                00000000'EF   DF  04AE                      PUSHAL  PRINT_SBE
                00000000'EF   DF  04B4                      PUSHAL  ARRAY_MSG18
                00000000'EF   DD  04BA                      PUSHL   LUN
                          11  DD  04C0                      PUSHL   #17
           00000000'9F    04  FB  04C2                      CALLS   #$$M, a#DS$ERRHARD
                              04C9   984 180$:   $DS_CKLOOP       10$                 ; SCOPE LOOP?
   00000000'9F    FC06 CF    FA  04C9                      CALLG   10$, a#DS$CKLOOP
        54    00000000'8F    D0  04D2   985          MOVL    #VAR_PAT18,R4           ; EXPECTED DATA
            0000'C2    54    D0  04D9   986          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
            53    0000'C2    D0  04DE   987          MOVL    VAR(R2),R3              ; READ BACK VAR
                  54    53    D1  04E3   988          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1B    13  04E6   989          BEQL    190$                    ; BRANCH IF OKAY
                                  04E8   990          $DS_ERRHARD_S   #18,LUN,-       ; PRINT ERROR
```

```
                                04E8   991                          ARRAY_MSG12,-
                                04E8   992                          PRINT_SBE
          00000000'EF    DF     04E8                     PUSHAL   PRINT_SBE
          00000000'EF    DF     04EE                     PUSHAL   ARRAY_MSG12
          00000000'EF    DD     04F4                     PUSHL    LUN
                    12   DD     04FA                     PUSHL    #18
     00000000'9F    04   FB     04FC                     CALLS    #$$M, a#DS$ERRHARD
                                0503   993 190$:   $DS_CKLOOP      10$                  ; SCOPE LOOP?
00000000'9F   FBCC CF    FA     0503                     CALLG    10$, a#DS$CKLOOP
     54   00000000'8F    D0     050C   994         MOVL    #VAR_PAT19,R4                ; EXPECTED DATA
          0000'C2   54   D0     0513   995         MOVL    R4,VAR(R2)                   ; WRITE LAST PATTERN TO VAR
          53   0000'C2   D0     0518   996         MOVL    VAR(R2),R3                   ; READ BACK VAR
               54   53   D1     051D   997         CMPL    R3,R4                        ; EXPECTED = RECEIVED?
                    1B   13     0520   998         BEQL    200$                         ; BRANCH IF OKAY
                                0522   999         $DS_ERRHARD_S   #19,LUN,-            ; PRINT ERROR
                                0522  1000                          ARRAY_MSG17,-
                                0522  1001                          PRINT_SBE
          00000000'EF    DF     0522                     PUSHAL   PRINT_SBE
          00000000'EF    DF     0528                     PUSHAL   ARRAY_MSG17
          00000000'EF    DD     052E                     PUSHL    LUN
                    13   DD     0534                     PUSHL    #19
     00000000'9F    04   FB     0536                     CALLS    #$$M, a#DS$ERRHARD
                                053D  1002 200$:   $DS_CKLOOP      10$                  ; SCOPE LOOP?
00000000'9F   FB92 CF    FA     053D                     CALLG    10$, a#DS$CKLOOP
     54   00000000'8F    D0     0546  1003         MOVL    #VAR_PAT20,R4                ; EXPECTED DATA
          0000'C2   54   D0     054D  1004         MOVL    R4,VAR(R2)                   ; WRITE LAST PATTERN TO VAR
          53   0000'C2   D0     0552  1005         MOVL    VAR(R2),R3                   ; READ BACK VAR
               54   53   D1     0557  1006         CMPL    R3,R4                        ; EXPECTED = RECEIVED?
                    1B   13     055A  1007         BEQL    210$                         ; BRANCH IF OKAY
                                055C  1008         $DS_ERRHARD_S   #20,LUN,-            ; PRINT ERROR
                                055C  1009                          ARRAY_MSG7,-
                                055C  1010                          PRINT_SBE
          00000000'EF    DF     055C                     PUSHAL   PRINT_SBE
          00000000'EF    DF     0562                     PUSHAL   ARRAY_MSG7
          00000000'EF    DD     0568                     PUSHL    LUN
                    14   DD     056E                     PUSHL    #20
     00000000'9F    04   FB     0570                     CALLS    #$$M, a#DS$ERRHARD
                                0577  1011 210$:   $DS_CKLOOP      10$                  ; SCOPE LOOP?
00000000'9F   FB58 CF    FA     0577                     CALLG    10$, a#DS$CKLOOP
     54   00000000'8F    D0     0580  1012         MOVL    #VAR_PAT21,R4                ; EXPECTED DATA
          0000'C2   54   D0     0587  1013         MOVL    R4,VAR(R2)                   ; WRITE LAST PATTERN TO VAR
          53   0000'C2   D0     058C  1014         MOVL    VAR(R2),R3                   ; READ BACK VAR
               54   53   D1     0591  1015         CMPL    R3,R4                        ; EXPECTED = RECEIVED?
                    1B   13     0594  1016         BEQL    220$                         ; BRANCH IF OKAY
                                0596  1017         $DS_ERRHARD_S   #21,LUN,-            ; PRINT ERROR
                                0596  1018                          ARRAY_MSG16,-
                                0596  1019                          PRINT_SBE
          00000000'EF    DF     0596                     PUSHAL   PRINT_SBE
          00000000'EF    DF     059C                     PUSHAL   ARRAY_MSG16
          00000000'EF    DD     05A2                     PUSHL    LUN
                    15   DD     05A8                     PUSHL    #21
     00000000'9F    04   FB     05AA                     CALLS    #$$M, a#DS$ERRHARD
                                05B1  1020 220$:   $DS_CKLOOP      10$                  ; SCOPE LOOP?
00000000'9F   FB1E CF    FA     05B1                     CALLG    10$, a#DS$CKLOOP
     54   00000000'8F    D0     05BA  1021         MOVL    #VAR_PAT22,R4                ; EXPECTED DATA
          0000'C2   54   D0     05C1  1022         MOVL    R4,VAR(R2)                   ; WRITE LAST PATTERN TO VAR
          53   0000'C2   D0     05C6  1023         MOVL    VAR(R2),R3                   ; READ BACK VAR
```

```
                     54    53    D1  05CB  1024          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                           1B    13  05CE  1025          BEQL    230$                    ; BRANCH IF OKAY
                                     05D0  1026          $DS_ERRHARD_S   #22,LUN,-       ; PRINT ERROR
                                     05D0  1027                          ARRAY_MSG12,-
                                     05D0  1028                          PRINT_SBE
                00000000'EF          DF  05D0                          PUSHAL  PRINT_SBE
                00000000'EF          DF  05D6                          PUSHAL  ARRAY_MSG12
                00000000'EF          DD  05DC                          PUSHL   LUN
                          16         DD  05E2                          PUSHL   #22
           00000000'9F    04         FB  05E4                          CALLS   #$$M, a#DS$ERRHARD
                                     05EB  1029 230$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
   00000000'9F    FAE4 CF          FA  05EB                          CALLG   10$, a#DS$CKLOOP
        54   00000000'8F          D0  05F4  1030          MOVL    #VAR_PAT23,R4           ; EXPECTED DATA
           0000'C2    54          D0  05FB  1031          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
           53   0000'C2          D0  0600  1032          MOVL    VAR(R2),R3              ; READ BACK VAR
                     54    53    D1  0605  1033          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                           1B    13  0608  1034          BEQL    240$                    ; BRANCH IF OKAY
                                     060A  1035          $DS_ERRHARD_S   #23,LUN,-       ; PRINT ERROR
                                     060A  1036                          ARRAY_MSG19,-
                                     060A  1037                          PRINT_SBE
                00000000'EF          DF  060A                          PUSHAL  PRINT_SBE
                00000000'EF          DF  0610                          PUSHAL  ARRAY_MSG19
                00000000'EF          DD  0616                          PUSHL   LUN
                          17         DD  061C                          PUSHL   #23
           00000000'9F    04         FB  061E                          CALLS   #$$M, a#DS$ERRHARD
                                     0625  1038 240$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
   00000000'9F    FAAA CF          FA  0625                          CALLG   10$, a#DS$CKLOOP
        54   00000000'8F          D0  062E  1039          MOVL    #VAR_PAT24,R4           ; EXPECTED DATA
           0000'C2    54          D0  0635  1040          MOVL    R4,VAR(R2)              ; WRITE LAST PATTERN TO VAR
           53   0000'C2          D0  063A  1041          MOVL    VAR(R2),R3              ; READ BACK VAR
                     54    53    D1  063F  1042          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                           1B    13  0642  1043          BEQL    250$                    ; BRANCH IF OKAY
                                     0644  1044          $DS_ERRHARD_S   #24,LUN,-       ; PRINT ERROR
                                     0644  1045                          ARRAY_MSG7,-
                                     0644  1046                          PRINT_SBE
                00000000'EF          DF  0644                          PUSHAL  PRINT_SBE
                00000000'EF          DF  064A                          PUSHAL  ARRAY_MSG7
                00000000'EF          DD  0650                          PUSHL   LUN
                          18         DD  0656                          PUSHL   #24
           00000000'9F    04         FB  0658                          CALLS   #$$M, a#DS$ERRHARD
                                     065F  1047 250$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
   00000000'9F    FA70 CF          FA  065F                          CALLG   10$, a#DS$CKLOOP
                                     0668  1048 $DS_ENDSUB
                                     0668       T5_S2_X:
00000000'9F    00000090'EF          FA  0668                          CALLG   $S$, a#DS$ENDSUB
                                     0673  1049

                                     0673  1050 $DS_ENDTEST
                     50    01    D0  0673                          MOVL    #1, R0          ; NORMAL EXIT
                                     0676       TEST_005_X::
           00000000'9F    6E         FA  0676                          CALLG   (SP), a#DS$BREAK
                                     04  067D                          RET                 ; RETURN TO TEST SEQUENCER
                                     067E  1051 $DS_PAGE
                                     067E               .SBTTL  TEST 6:  MAP REGISTER SA0 TEST
                                     00000000                   .PSECT  TEST_006, PAGE, NOWRT
                                     0018
                                     0018  1053 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
```

```
                    00000000  0018                          .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                              001C          TEST_006::
                        0000  001C                          .WORD   M<>             ; ENTRY MASK
                              001E  1054 ;++
                              001E  1055 ; TEST DESCRIPTION:
                              001E  1056 ;
                              001E  1057 ;       THIS TEST CHECKS TO SEE THAT WRITING TO 1 REGISTER DOESN'T AFFECT ANY
                              001E  1058 ;       OTHER REGISTERS.  IT ALSO CHECKS THE MAP REGISTERS FOR STUCK AT ZERO
                              001E  1059 ;       FAULTS AND DUAL ADDRESSING FAULTS.
                              001E  1060 ;
                              001E  1061 ;
                              001E  1062 ;
                              001E  1063 ; TEST ALGORITHM:
                              001E  1064 ;
                              001E  1065 ;       FIRST, IT CLEARS 1 REGISTER, AND WRITES A -1 TO ALL OTHERS, CHECKING
                              001E  1066 ;       THE REGISTER CLEARED FOR NON-ZERO CONTENTS (INDICATES ERROR) EACH
                              001E  1067 ;       TIME.  IT THEN ADVANCES UNTIL ALL 256 REGISTERS ARE CHECKED.  THEN
                              001E  1068 ;       WRITE ALL ONE'S PATTERN IN EACH REGISTER AND CHECK FOR ZERO BITS
                              001E  1069 ;       FLOAT A ONE THROUGH EACH DEFINED BIT POSITION IN EACH  MAP REGISTER
                              001E  1070 ;       WRITE A UNIQUE PATTERN IN A SINGLE CELL AND CHECK THAT THE
                              001E  1071 ;       PATTERN IS NOT WRITTEN INTO ANY OTHER CELL.
                              001E  1072 ;--
                              001E  1073          ERRPREP RHMAPR_MSG,10,-         ; PREPARE TO PRINT ERROR
                              001E  1074                  FMT_MAP_REG
    00000000'EF   00000000'EF   DE  001E                  MOVAL   RHMAPR_MSG,REG_NAME
          00000000'EF    0A     9A  0029                  MOVZBL  #10,REG_NO
    00000000'EF   00000000'EF   DE  0030                  MOVAL   FMT_MAP_REG,REG_STRING
            52    00000000'EF   D0  003B  1075          MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS INTO R2
        0000'C2   00000000'8F   C8  0042  1076          BISL    #PGM_INIT,CR(R2)        ; INIT IT
            52    00000000'8F   C0  004B  1077          ADDL2   #MAP_OFFSET,R2          ; R2 NOW POINTS TO THE MAP REGISTERS
                                    0052  1078          $DS_BGNSUB
                                    0052          T6_S1::
00000000'9F     0000009C'EF   FA    0052                      CALLG   $S$, a#DS$BGNSUB
                00000065'EF   94    005D  1079          CLRB    20$                     ; CLEAR THE ERROR FLAG
                         01   11    0063  1080          BRB     1$                      ; BRANCH OVER THE ERROR FLAG BYTE
                              00    0065  1081 20$:      .BYTE   0                       ; LOCATION FOR THE ERROR FLAG
                           51 D4    0066  1082 1$:       CLRL    R1                      ; CLEAR REGISTER UNDER TEST INDEX REGISTER
                           53 D4    0068  1083 2$:       CLRL    R3                      ; CLEAR DATA TO LOAD INDEX REGISTER
                         6241 D4    006A  1084          CLRL    (R2) R1                 ; CLEAR THE REGISTER UNDER TEST
                      53    51 D1    006D  1085 3$:       CMPL    R1,R3                   ; SEE IF REGISTER UNDER TEST IS THE LOAD REG
                           4F 13    0070  1086          BEQL    4$                      ; BRANCH IF SO TO CONTINUE
                6243  FFFFFFFF 8F   D0    0072  1087          MOVL    #-1,(R2) R3             ; MOVE ALL 1'S TO THE LOAD REGISTER
                         6241 D5    007A  1088          TSTL    (R2) R1                 ; SEE IF REGISTER UNDER TEST WAS CHANGED
                           42 13    007D  1089          BEQL    4$                      ; BRANCH IF OK
                        E3 AF   95    007F  1090          TSTB    20$                     ; HAS AN ERROR ALREADY BEEN CALLED?
                           3D 12    0082  1091          BNEQ    4$                      ; BRANCH IF SO
                        DE AF   96    0084  1092          INCB    20$                     ; SET FLAG SO NO MORE ERRORS CALLED AFTER TH
                54    51    52 C1    0087  1093          ADDL3   R2,R1,R4                ; FORM ADDRESS OF ERRORED REGISTER
                55    53    52 C1    008B  1094          ADDL3   R2,R3,R5                ; FORM ADDRESS OF REGISTER WRITTEN
                      56    64 D0    008F  1095          MOVL    (R4),R6                 ; MOVE DATA TO R6
                      57    65 D0    0092  1096          MOVL    (R5),R7                 ; MOVE DATA TO R7
                                    0095  1097          $DS_ERRHARD_S   #1,LUN,-
                                    0095  1098                          FMT_DUALWT_ERR  ; PRINT ERROR
                           00 DD    0095                      PUSHL   #0
                00000000'EF   DF    0097                      PUSHAL  FMT_DUALWT_ERR
                00000000'EF   DD    009D                      PUSHL   LUN
                           01 DD    00A3                      PUSHL   #1
```

```
                00000000'9F    04    FB  00A5                        CALLS   #$$M, a#DS$ERRHARD
                                     00AC  1099          $DS_PRINTX_S    FMT_DUALWT_MSG,-
                                     00AC  1100                          R5,R4,R7,R6     ; PRINT ERROR PARAMETERS
                          56    DD  00AC                        PUSHL   R6
                          57    DD  00AE                        PUSHL   R7
                          54    DD  00B0                        PUSHL   R4
                          55    DD  00B2                        PUSHL   R5
                   00000000'EF  9F  00B4                        PUSHAB  FMT_DUALWT_MSG
                00000000'9F     05  FB  00BA                        CALLS   #$$N, a#DS$PRINTX
            A4 53     00000100 8F  F2  00C1  1101 4$:     AOBLSS  # X100,R3,3$            ; LOAD REGISTER LOOP
            97 51     00000100 8F  F2  00C9  1102         AOBLSS  # X100,R1,2$            ; REGISTER UNDER TEST LOOP
                                     00D1  1103         $DS_ENDSUB
                                     00D1       T6_S1_X:
    00000000'9F     0000009C'EF  FA  00D1                        CALLG   $S$, a#DS$ENDSUB
                                     00DC  1104

                                     00DC  1105         $DS_BGNSUB
                                     00DC       T6_S2::
    00000000'9F     000000A8'EF  FA  00DC                        CALLG   $S$, a#DS$BGNSUB
                          5A    D4  00E7  1106         CLRL    R10                     ; CLEAR MAP INDEX
                54    00000000'8F  D0  00E9  1107         MOVL    #MAP_1_PATTRN,R4        ; SET UP ALL ONE'S PATTERN
                        624A    54  D0  00F0  1108 10$:     MOVL    R4,(R2) R10             ; WRITE MAP REGISTER R10
                        53    624A  D0  00F4  1109         MOVL    (R2) R10 ,R3            ; READ MAP REGISTER R10
                     55    54    53  CD  00F8  1110         XORL3   R3,R4,R5                ; PUT DIFFERNCE IN R5
                           1B        13  00FC  1111         BEQL    20$                     ; BRANCH IF NO ERRORS
                                     00FE  1112         $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                     00FE  1113                         ARRAY_MSG14,-
                                     00FE  1114                         PRINT_SBE
                   00000000'EF  DF  00FE                        PUSHAL  PRINT_SBE
                   00000000'EF  DF  0104                        PUSHAL  ARRAY_MSG14
                   00000000'EF  DD  010A                        PUSHL   LUN
                          01    DD  0110                        PUSHL   #1
                00000000'9F     04  FB  0112                        CALLS   #$$M, a#DS$ERRHARD
                                     0119  1115 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
                00000000'9F     D4 AF  FA  0119                      CALLG   10$, a#DS$CKLOOP
                C7 5A     000000FF 8F  F3  0121  1116         AOBLEQ  #255,R10,10$            ; DO FOR ALL MAP REGISTERS
                                     0129  1117         $DS_ENDSUB
                                     0129       T6_S2_X:
    00000000'9F     000000A8'EF  FA  0129                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0134  1119          $DS_BGNSUB
                                0134        T6_S3::
00000000'9F   000000B4'EF   FA  0134                CALLG   $S$, a#DS$BGNSUB
                      5A    D4  013F  1120          CLRL    R10                     ; RESET MAP REGISTER INDEX
                    624A    D4  0141  1121 30$:     CLRL    (R2) R10                ; CLEAR ALL MAP REGISTERS
        F5 5A   000000FF 8F   F3  0144  1122          AOBLEQ  #255,R10,30$            ; INCREMENT MAP REGISTER INDEX
                                014C  1123                                          ; DO FOR ALL MAP REGISTERS
                      5A    D4  014C  1124          CLRL    R10                     ; CLEAR MAP REGISTER INDEX
                      5B    D4  014E  1125          CLRL    R11                     ; CLEAR BIT SHIFT COUNT
                   54   01    D0  0150  1126 40$:     MOVL    #1,R4                   ; INITIALIZE BIT PATTERN
                        04    11  0153  1127          BRB     60$                     ; SKIP SHIFT FIRST PASS
                54 54   01    9C  0155  1128 50$:     ROTL    #1,R4,R4                ; SHIFT BIT LEFT ONE PLACE
                   624A 54    D0  0159  1129 60$:     MOVL    R4,(R2) R10             ; WRITE MAP REGISTER R10
                   53   624A  D0  015D  1130          MOVL    (R2) R10 ,R3            ; READ MAPR REGISTER R10
                     54 53    D1  0161  1131          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                        1E    13  0164  1132          BEQL    70$                     ; IF NO ERRORS SKIP THEN CHECK LOOP
                     55 54    D0  0166  1133          MOVL    R4,R5                   ; MOVE EXPECTED REG TO CONVERSION REG
                                0169  1134          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                0169  1135                          ARRAY_MSG16,-
                                0169  1136                          PRINT_SBE
                00000000'EF     DF  0169                PUSHAL  PRINT_SBE
                00000000'EF     DF  016F                PUSHAL  ARRAY_MSG16
                00000000'EF     DD  0175                PUSHL   LUN
                        02      DD  017B                PUSHL   #2
           00000000'9F   04     FB  017D                CALLS   #$$M, a#DS$ERRHARD
                                0184  1137 70$:     $DS_CKLOOP      60$             ; SCOPE LOOP?
      00000000'9F   D2 AF     FA  0184                CALLG   60$, a#DS$CKLOOP
            C5 5B   0E        F2  018C  1138          AOBLSS  #14,R11,50$             ; DO FOR ALL BIT POSITIONS
      B8 5A   000000FF 8F     F3  0190  1139          AOBLEQ  #255,R10,40$            ; INCREMENT MAP REGISTER INDEX
         54   00000000'8F     D0  0198  1140          MOVL    #VALID_BIT,R4           ; PREPARE TEST PATTERN
                      5A      D4  019F  1141          CLRL    R10                     ; INITIALIZE MAP REGISTER INDEX
                  624A  54    D0  01A1  1142 80$:     MOVL    R4,(R2) R10             ; WRITE V BIT IN MAP REGISTER R10
                  53    624A  D0  01A5  1143          MOVL    (R2) R10 ,R3            ; READ BACK MAP REGISTER
                    54  53    D1  01A9  1144          CMPL    R3,R4                   ; IS V BIT SET
                        1B    13  01AC  1145          BEQL    90$                     ; IF NO ERRORS SKIP REPORT
                                01AE  1146          $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                                01AE  1147                          ARRAY_MSG12,-
                                01AE  1148                          PRINT_SBE
                00000000'EF     DF  01AE                PUSHAL  PRINT_SBE
                00000000'EF     DF  01B4                PUSHAL  ARRAY_MSG12
                00000000'EF     DD  01BA                PUSHL   LUN
                        03      DD  01C0                PUSHL   #3
           00000000'9F   04     FB  01C2                CALLS   #$$M, a#DS$ERRHARD
                                01C9  1149 90$:     $DS_CKLOOP      80$             ; SCOPE LOOP?
      00000000'9F   D5 AF     FA  01C9                CALLG   80$, a#DS$CKLOOP
      C8 5A   000000FF 8F     F3  01D1  1150          AOBLEQ  #255,R10,80$            ; DO FOR ALL V BITS
                                01D9  1151          $DS_ENDSUB
                                01D9        T6_S3_X:
00000000'9F   000000B4'EF   FA  01D9                CALLG   $S$, a#DS$ENDSUB
```

```
                                01E4  1153          $DS_BGNSUB
                                01E4          T6_S4::
00000000'9F   000000C0'EF   FA  01E4                      CALLG   $S$, a#DS$BGNSUB
                       5A   D4  01EF  1154          CLRL    R10                     ; CLEAR MAP UNDER TEST INDEX
                       56   D4  01F1  1155 10$:     CLRL    R6                      ; CLEAR MAP INDEX
                     6246   D4  01F3  1156 20$:     CLRL    (R2) R6                 ; CLEAR ALL MAP REGISTERS
          F5 56   000000FF 8F   F3  01F6  1157          AOBLEQ  #255,R6,20$             ; DO FOR ALL MAP REGISTERS
                    624A   00   D2  01FE  1158 30$:     MCOML   #0,(R2) R10             ; WRITE MAP UNDER TEST
                       56   D4  0202  1159          CLRL    R6                      ; CLEAR MAP INDEX USED IN CHECK LOOP
    53   6246   00000000'8F   CB  0204  1160 40$:     BICL3   #MAP_1_PATTRN,(R2) R6 ,R3; READ MAP REGISTERS NOT WRITTEN
                       2D   13  020D  1161          BEQL    50$                     ; IF MAP IS ZERO THEN NO DUAL ADDR ERR
                     5A   56   D1  020F  1162          CMPL    R6,R10                  ; IS MAP SELECTED THE ONE UNDER TEST
                       28   13  0212  1163          BEQL    50$                     ; SKIP IF COMPARED MAP IS UNDER TEST
                                0214  1164          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                0214  1165                          PRINT_SBE
                00000000'EF   DF  0214                      PUSHAL  PRINT_SBE
                          00   DD  021A                      PUSHL   #0
                00000000'EF   DD  021C                      PUSHL   LUN
                          01   DD  0222                      PUSHL   #1
             00000000'9F      04  FB  0224                      CALLS   #$$M, a#DS$ERRHARD
                                022B  1166          $DS_PRINTX_S    FMT_DUADR_ERR,- ; PRINT ERROR PARAMETERS
                                022B  1167                          R10,R6
                       56   DD  022B                      PUSHL   R6
                       5A   DD  022D                      PUSHL   R10
                00000000'EF   9F  022F                      PUSHAB  FMT_DUADR_ERR
             00000000'9F      03  FB  0235                      CALLS   #$$N, a#DS$PRINTX
                                023C  1168 50$:     $DS_CKLOOP      30$             ; SCOPE LOOP?
          00000000'9F   BF AF   FA  023C                      CALLG   30$, a#DS$CKLOOP
          B8 56   000000FF 8F   F3  0244  1169          AOBLEQ  #255,R6,40$             ; CONTINUE CHECKING UNWRITTEN MAP REG
FF9B 5A      01   000000FF 8F   F1  024C  1170          ACBL    #255,#1,R10,10$         ; UPDATE MAP UNDER TEST INDEX
                                0256  1171          $DS_ENDSUB
                                0256          T6_S4_X:
 00000000'9F   000000C0'EF   FA  0256                      CALLG   $S$, a#DS$ENDSUB
```

```
                                  0261  1173 $DS_ENDTEST
                  50   01   D0    0261                        MOVL    #1, R0          ; NORMAL EXIT
                                  0264        TEST_006_X::
        00000000'9F   6E   FA    0264                        CALLG   (SP), a#DS$BREAK
                             04   026B                        RET                     ; RETURN TO TEST SEQUENCER
                                  026C  1174 $DS_PAGE
                                  026C                .SBTTL  TEST 7:  EXTERNAL REGISTER DATA PATH TEST
                             00000000                        .PSECT  TEST_007, PAGE, NOWRT
                                  0024
                                  0024  1176 $DS_BGNTEST      <DEFAULT,ALL>,,LONG
                                  0024        DATA_007:
                        00000000  0024                        .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  0028        TEST_007::
                            0000  0028                        .WORD    M<>            ; ENTRY MASK
                                  002A  1177 ;++
                                  002A  1178 ; TEST DESCRIPTION:
                                  002A  1179 ;
                                  002A  1180 ;       THIS TEST CHECKS THE MASS BUS CONTROL PATH
                                  002A  1181 ;       FOR STUCK AT 0 AND STUCK AT 1 CONDITIONS.
                                  002A  1182 ;       IT ALSO CHECKS FOR CONTROL PATH PARITY ERRORS.
                                  002A  1183 ;
                                  002A  1184 ; TEST ALGORITHM:
                                  002A  1185 ;
                                  002A  1186 ;       WRITE 0'S ONTO THE CONTROL PATH AND VERIFY NO STUCK AT 1 FAULTS
                                  002A  1187 ;       FLOAT A ONE IN EACH BIT POSITION.
                                  002A  1188 ;       WRITE INCREMENTING PATTERN ONTO CONTROL PATH CHECKING FOR
                                  002A  1189 ;       CONTROL PATH PARITY ERRORS.
                                  002A  1190 ;--
                        FFD3'  30 002A  1191          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0   002D  1192          MOVL    RH_CUR_ADR,R2           ; MOVE MUT'S ADDRESS TO R2
                                  0034  1193          ERRPREP EXT_REG_MSG,11,-        ; PREPARE TO PRINT ERROR
                                  0034  1194                  FMT_ANY_REG
  00000000'EF   00000000'EF   DE  0034                        MOVAL   EXT_REG_MSG,REG_NAME
          00000000'EF   0B   9A   003F                        MOVZBL  #11,REG_NO
  00000000'EF   00000000'EF   DE  0046                        MOVAL   FMT_ANY_REG,REG_STRING
          58   00000000'EF   D0   0051  1195          MOVL    DRIVE0,R8               ; MOVE ADDRESS OF DRIVE 0 TO R8
                  58   04   C0    0058  1196          ADDL    #4,R8                   ; SELECT REGISTER 1
     0000'C2   00000000'8F   C8   005B  1197 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
     0000'C2   00000000'8F   C8   0064  1198          BISL    #MAINT_MODE,CR(R2)      ; PLACE RH750 IN MAINTENANCE MODE
                        68   D4   006D  1199          CLRL    (R8)                    ; CLEAR DRIVE
                        54   D4   006F  1200          CLRL    R4                      ; EXPECT 0'S BACK FROM DRIVE REGISTER
                  50   68   D0    0071  1201          MOVL    (R8),R0                 ; READ DRIVE REGISTER
                  53   50   3C    0074  1202          MOVZWL  R0,R3                   ; EXTRACT CONTENTS OF DRIVE REGISTER
                  54   53   D1    0077  1203          CMPL    R3,R4                   ; ANY STUCK AT ONE BITS
                        1E   13   007A  1204          BEQL    20$                     ; BRANCH IF NOT
                  55   53   D0    007C  1205          MOVL    R3,R5                   ; DISPLAY BITS IN ERROR
                                  007F  1206          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                  007F  1207                          ARRAY_MSG14,-
                                  007F  1208                          PRINT_SBE
              00000000'EF   DF   007F                        PUSHAL  PRINT_SBE
              00000000'EF   DF   0085                        PUSHAL  ARRAY_MSG14
              00000000'EF   DD   008B                        PUSHL   LUN
                        01   DD   0091                        PUSHL   #1
          00000000'9F   04   FB   0093                        CALLS   #$$M, a#DS$ERRHARD
                                  009A  1209 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
       00000000'9F   BE AF   FA   009A                        CALLG   10$, a#DS$CKLOOP
                                  00A2  1210          ERRPREP EXT_REG_MSG,11,-        ; PREPARE TO PRINT ERROR
```

```
                                    00A2  1211                    FMT_ANY_REG
00000000'EF   00000000'EF   DE   00A2                            MOVAL   EXT_REG_MSG,REG_NAME
        00000000'EF   0B    9A   00AD                            MOVZBL  #11,REG_NO
00000000'EF   00000000'EF   DE   00B4                            MOVAL   FMT_ANY_REG,REG_STRING
                      57    D4   00BF  1212          CLRL    R7                      ; CLEAR SHIFT COUNT
                  54  01    D0   00C1  1213 30$:     MOVL    #1,R4                   ; INITIALIZE BIT PATTERN
                  68  54    D0   00C4  1214 40$:     MOVL    R4,(R8)                 ; WRITE PATTERN INTO DRIVE REGISTER
                  50  68    D0   00C7  1215          MOVL    (R8),R0                 ; READ DRIVE REGISTER
                  53  50    3C   00CA  1216          MOVZWL  R0,R3                   ; EXTRACT CONTENTS OF DRIVE REGIATER
                  54  53    D1   00CD  1217          CMPL    R3,R4                   ; IS BIT SET ON MASS BUS CONTROL PATH?
                      1E    13   00D0  1218          BEQL    50$                     ; BRANCH IF CONTROL PATH OK
                  55  54    D0   00D2  1219          MOVL    R4,R5                   ; REPORT BIT IN ERROR
                                00D5  1220          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                00D5  1221                          ARRAY_MSG14,-
                                00D5  1222                          PRINT_SBE
              00000000'EF       DF   00D5                           PUSHAL  PRINT_SBE
              00000000'EF       DF   00DB                           PUSHAL  ARRAY_MSG14
              00000000'EF       DD   00E1                           PUSHL   LUN
                        02      DD   00E7                           PUSHL   #2
          00000000'9F   04      FB   00E9                           CALLS   #$$M, a#DS$ERRHARD
                                00F0  1223 50$:     $DS_CKLOOP      40$             ; SCOPE LOOP?
      00000000'9F   D1  AF      FA   00F0                           CALLG   40$, a#DS$CKLOOP
            54  54    01      9C   00F8  1224          ROTL    #1,R4,R4                ; SHIFT BIT LEFT ONE PLACE
              C4 57    0F      F3   00FC  1225          AOBLEQ  #15,R7,40$              ; FLOAT ONE THROUGH ALL BIT POSITIONS
                       57      D4   0100  1226          CLRL    R7                      ; CLEAR SHIFT COUNT FOR PREVIOUS REG
                               0102  1227          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                               0102  1228                  FMT_STATUS_REG
00000000'EF   00000000'EF   DE   0102                            MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A   010D                            MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE   0114                            MOVAL   FMT_STATUS_REG,REG_STRING
                      54    D4   011F  1229          CLRL    R4                      ; CLEAR EXPECTED STATUS
                      56    D4   0121  1230          CLRL    R6                      ; CLEAR DRIVE REGISTER PATTERN
    0000'C2   00000000'8F   C8   0123  1231          BISL    #PGM_INIT,CR(R2)        ; INITIALIZE THE RH
    0000'C2   00000000'8F   C8   012C  1232          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
                  68  56    D0   0135  1233 60$:     MOVL    R6,(R8)                 ; WRITE DRIVE 0 REGISTER 0
                  53  68    D0   0138  1234          MOVL    (R8),R3                 ; READ DRIVE REGISTER
        53  00000000'8F     CA   013B  1235          BICL    #BYTE0!BYTE1,R3         ; CLEAR LOWER WORD
        54  00000000'EF     C8   0142  1236          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                  54  53    D1   0149  1237          CMPL    R3,R4                   ; CHECK FOR CONTROL PARITY ERROR
                      20    13   014C  1238          BEQL    70$                     ; BRANCH IF NO ERROR
            53    0000'C2   D0   014E  1239          MOVL    SR(R2),R3               ; MOVE STATUS REGISTER TO R3
                                0153  1240          $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                                0153  1241                          ARRAY_MSG32,-
                                0153  1242                          PRINT_SBE
              00000000'EF       DF   0153                           PUSHAL  PRINT_SBE
              00000000'EF       DF   0159                           PUSHAL  ARRAY_MSG32
              00000000'EF       DD   015F                           PUSHL   LUN
                        03      DD   0165                           PUSHL   #3
          00000000'9F   04      FB   0167                           CALLS   #$$M, a#DS$ERRHARD
                                016E  1243 70$:     $DS_CKLOOP      60$             ; SCOPE LOOP?
      00000000'9F   C4  AF      FA   016E                           CALLG   60$, a#DS$CKLOOP
      B7 56   00010000  8F      F3   0176  1244          AOBLEQ  # X10000,R6,60$         ; INDEX CONTROL PATH PATTERN
                               017E  1245 $DS_ENDTEST
                  50  01    D0   017E                           MOVL    #1, R0          ; NORMAL EXIT
                               0181       TEST_007_X::
          00000000'9F   6E    FA   0181                           CALLG   (SP), a#DS$BREAK
                        04    0188                                RET                     ; RETURN TO TEST SEQUENCER
```

```
                0189  1246 $DS_PAGE
```

```
                        0189                .SBTTL  TEST 8:  CAR BUS FUNCTION TEST
                 00000000                   .PSECT  TEST_008, PAGE, NOWRT
                        0018
                        0018  1249 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                        0018       DATA_008:
                 00000000       0018            .LONG   0                       ; TEST ARGUMENT TABLE TERMINATOR
                        001C       TEST_008::
                     0000       001C            .WORD    M<>                    ; ENTRY MASK
                        001E  1250 ;++
                        001E  1251 ; TEST DESCRIPTION:
                        001E  1252 ;
                        001E  1253 ;       THIS TEST IS COMPRISED OF FOUR SUBTESTS, CHECKING
                        001E  1254 ;       THE ABILITY TO PERFORM A READ COMMAND, AND THEN
                        001E  1255 ;       VERIFYING THE CMI BUS FUNCTION BITS, CAR<27:25>.
                        001E  1256 ;
                        001E  1257 ;--
                        001E  1258 $DS_BGNSUB
                        001E       T8_S1::
 00000000'9F    000000CC'EF    FA  001E                    CALLG   $S$, a#DS$BGNSUB
                        0029  1259
                        0029  1260 ;
                        0029  1261 ; Subtest one first verifies that a Read Command can be
                        0029  1262 ; successfully executed while in the Maintenance Mode and
                        0029  1263 ; that Mass_Run DR<19> can be set and cleared.
                        0029  1264 ;
                        0029  1265
                  FFD4'  30  0029  1266          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52    00000000'EF  D0  002C  1267          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
                        0033  1268          ERRPREP RHDR_MSG,5,-            ; PREPARE TO PRINT ERROR
                        0033  1269                  FMT_DIAG_REG,SA1_MSG
 00000000'EF    00000000'EF  DE  0033                  MOVAL   RHDR_MSG,REG_NAME
          00000000'EF    05  9A  003E                  MOVZBL  #5,REG_NO
 00000000'EF    00000000'EF  DE  0045                  MOVAL   FMT_DIAG_REG,REG_STRING
     0000'C2    00000000'8F  C8  0050  1270 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
     0000'C2    00000000'8F  C8  0059  1271          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
          56    00000000'EF  D0  0062  1272          MOVL    DRIVE0,R6               ; PUT ADDRESS OF DRIVE 0 IN R6
          54    00000000'8F  D0  0069  1273          MOVL    #MASS_RUN!MASS_FAIL!MASS_CTOD,R4 ; EXPECTED RESULTS
          66    00000000'8F  D0  0070  1274          MOVL    #READ,(R6)              ; ISSUE READ COMMAND TO DRIVE 0
 53  0000'C2    00000000'8F  CB  0077  1275          BICL3   #BYTE0,DR(R2),R3        ; READ DIAGNOSTIC REGISTER
     0000'C2    00000000'8F  C8  0081  1276          BISL    #SIM_OCC,DR(R2)         ; ASSERT OCC--PREVENT MISSED XFER
                    54    53  D1  008A  1277          CMPL    R3,R4                   ; CHECK
                    1B    13  008D  1278          BEQL    20$                     ; BRANCH IF MRUN IS SET
                        008F  1279          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                        008F  1280                          ARRAY_MSG53,-
                        008F  1281                          PRINT_SBE
                 00000000'EF  DF  008F                  PUSHAL  PRINT_SBE
                 00000000'EF  DF  0095                  PUSHAL  ARRAY_MSG53
                 00000000'EF  DD  009B                  PUSHL   LUN
                          01  DD  00A1                  PUSHL   #1
          00000000'9F    04  FB  00A3                  CALLS   #$$M, a#DS$ERRHARD
                        00AA  1282 20$:     $DS_CKLOOP      10$                     ; SCOPE LOOP?
        00000000'9F    A3  AF  FA  00AA                  CALLG   10$, a#DS$CKLOOP
     0000'C2    00000000'8F  C8  00B2  1283          BISL    #DT_ABORT,CR(R2)        ; CAUSE DATA TRANSFER ABORT
                00000000'8F  D0  00BB  1284          MOVL    #MASS_FAIL!SIM_OCC!-    ; SET UP EXPECTED RESULTS
                          54       00C1  1285                  MASS_CTOD,R4            ; MERGE THESE BITS
 53  0000'C2    00000000'8F  CB  00C2  1286          BICL3   #BYTE1!BYTE0,DR(R2),R3  ; READ DIAGNOSTIC REGISTER
                    54    53  D1  00CC  1287          CMPL    R3,R4                   ; CHECK
```

```
                          1B   13  00CF  1288          BEQL    30$                     ; BRANCH IF OK
                                   00D1  1289          $DS_ERRHARD_S   #2,LUN,-         ; REPORT ERROR
                                   00D1  1290                          ARRAY_MSG57,-
                                   00D1  1291                          PRINT_SBE
                00000000'EF        DF  00D1                        PUSHAL  PRINT_SBE
                00000000'EF        DF  00D7                        PUSHAL  ARRAY_MSG57
                00000000'EF        DD  00DD                        PUSHL   LUN
                          02       DD  00E3                        PUSHL   #2
           00000000'9F    04       FB  00E5                        CALLS   #$$M, a#DS$ERRHARD
                                   00EC  1292 30$:     $DS_CKLOOP          10$             ; SCOPE LOOP?
      00000000'9F    FF60 CF      FA  00EC                        CALLG   10$, a#DS$CKLOOP
                                   00F5  1293 $DS_ENDSUB
                                   00F5       T8_S1_X:
   00000000'9F    000000CC'EF      FA  00F5                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0100  1295 $DS_BGNSUB
                                0100       T8_S2::
00000000'9F   000000D8'EF   FA   0100                 CALLG   $S$, a#DS$BGNSUB
                                010B  1296 ;
                                010B  1297 ; Subtest two checks for a CMI Read Command by performing a W-T-D
                                010B  1298 ; function and then reading the CAR and checking for the correct
                                010B  1299 ; bus function code (0).
                                010B  1300 ;
                                010B  1301
      52   00000000'EF   D0   010B  1302          MOVL    RH_CUR_ADR,R2          ; STORE RH750 ADDRESS
 0000'C2   00000000'8F   C8   0112  1303 10$:     BISL    #PGM_INIT,CR(R2)       ; INIT THE MUT
      56   00000000'EF   DE   011B  1304          MOVAL   MIOBUFFER,R6           ; STORE BUFFER ADDRESS POINTER
        57   56   00'8F   78   0122  1305          ASHL    #-PF_FIELD,R6,R7       ; EXTRACT PFN
      57   00000000'8F   C8   0127  1306          BISL    #VALID_BIT,R7          ; SET V-BIT IN MAP ENTRY
5A    52   00000000'8F   C1   012E  1307          ADDL3   #MAP_OFFSET,R2,R10     ; R10 POINTS TO MAP REG 0
             6A   57   D0   0136  1308          MOVL    R7,(R10)               ; WRITE PFN INTO MAP REG 0
             0000'C2   D4   0139  1309          CLRL    VAR(R2)                ; SELECT MAP ZERO
          0000'C2   02   CE   013D  1310          MNEGL   #2,BCR(R2)             ; SET BCR FOR 2 BYTE TRANSFER
 0000'C2   00000000'8F   C8   0142  1311          BISL    #MAINT_MODE,CR(R2)     ; SET MAINT MODE
      54   00000000'EF   D0   014B  1312          MOVL    DRIVE0,R4              ; GET ADDRESS OF DRIVE 0
      64   00000000'8F   D0   0152  1313          MOVL    #WRITE,(R4)            ; PERFORM CMI READ
 0000'C2   00000000'8F   C8   0159  1314          BISL    #SIM_OCC,DR(R2)        ; ASSERT OCCUPIED
 0000'C2   00000000'8F   C8   0162  1315          BISL    #BLKSND_COMD,DR(R2)    ; DO NOT ALLOW CMI TRANSACTIONS
 0000'C2   00000000'8F   C8   016B  1316          BISL    #SIM_SCLK,DR(R2)       ; ASSERT SCLK
 0000'C2   00000000'8F   CA   0174  1317          BICL    #SIM_SCLK,DR(R2)       ; DEASSERT SCLK
        59   0000'C2   D0   017D  1318          MOVL    CAR(R2),R9             ; READ CAR
 0000'C2   00000000'8F   CA   0182  1319          BICL    #BLKSND_COMD,DR(R2)    ; ALLOW CMI TRANSACTIONS
 0000'C2   00000000'8F   C8   018B  1320          BISL    #SIM_EBL,DR(R2)        ; ASSERT EBL
 0000'C2   00000000'8F   CA   0194  1321          BICL    #SIM_EBL,DR(R2)        ; DEASSERT EBL
      57   59   E7 8F   78   019D  1322          ASHL    #-25,R9,R7             ; ISOLATE CMI COMMAND
      57   00000008'8F   CA   01A2  1323          BICL    #BIT3!NIBBLE1!BYTE1!BYTE2!BYTE3,R7  ; CLEAR SIGN EXTENSION
             57   00   91   01A9  1324          CMPB    #0,R7                  ; CHECK FOR CMI READ
                  1F   13   01AC  1325          BEQL    20$                    ; BRANCH IF COMMAND IS OK
                           01AE  1326          $DS_ERRHARD_S   #1,LUN,-       ; PRINT ERROR
                           01AE  1327                          ARRAY_MSG29,-
                           01AE  1328                          PRINT_CMDER,-
                           01AE  1329                          #0,R7
                       57   DD   01AE                 PUSHL   R7
                       00   DD   01B0                 PUSHL   #0
             00000000'EF   DF   01B2                 PUSHAL  PRINT_CMDER
             00000000'EF   DF   01B8                 PUSHAL  ARRAY_MSG29
             00000000'EF   DD   01BE                 PUSHL   LUN
                       01   DD   01C4                 PUSHL   #1
      00000000'9F   06   FB   01C6                 CALLS   #$$M, a#DS$ERRHARD
                           01CD  1330 20$:     $DS_CKLOOP      10$            ; SCOPE LOOP?
00000000'9F   FF41   CF   FA   01CD                 CALLG   10$, a#DS$CKLOOP
                           01D6  1331 $DS_ENDSUB
                           01D6       T8_S2_X:
00000000'9F   000000D8'EF   FA   01D6                 CALLG   $S$, a#DS$ENDSUB
```

```
                                01E1  1333 $DS_BGNSUB
                                01E1       T8_S3::
00000000'9F   000000E4'EF   FA  01E1                CALLG   $S$, a#DS$BGNSUB
                                01EC  1334 ;
                                01EC  1335 ; Subtest three performs a Write Check function (CMI Read), and
                                01EC  1336 ; checks the CAR for the correct code (0).
                                01EC  1337 ;
                                01EC  1338
      52   00000000'EF   D0  01EC  1339        MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
0000'C2    00000000'8F   C8  01F3  1340 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
      56   00000000'EF   DE  01FC  1341        MOVAL   MIOBUFFER,R6            ; STORE BUFFER ADDRESS POINTER
        57   56   00'8F   78  0203  1342        ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
      57   00000000'8F   C8  0208  1343        BISL    #VALID_BIT,R7           ; SET V-BIT IN MAP ENTRY
5A    52   00000000'8F   C1  020F  1344        ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
               6A   57   D0  0217  1345        MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
            0000'C2      D4  021A  1346        CLRL    VAR(R2)                 ; SELECT MAP ZERO
          0000'C2   02   CE  021E  1347        MNEGL   #2,BCR(R2)              ; SET BCR FOR 2 BYTE TRANSFER
0000'C2    00000000'8F   C8  0223  1348        BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
      54   00000000'EF   D0  022C  1349        MOVL    DRIVE0,R4               ; GET ADDRESS OF DRIVE 0
      64   00000000'8F   D0  0233  1350        MOVL    #WRITE_CHECK,(R4)       ; PERFORM CMI READ
0000'C2    00000000'8F   C8  023A  1351        BISL    #SIM_OCC,DR(R2)         ; ASSERT OCCUPIED
0000'C2    00000000'8F   C8  0243  1352        BISL    #BLKSND_COMD,DR(R2)     ; DO NOT ALLOW CMI TRANSACTIONS
0000'C2    00000000'8F   C8  024C  1353        BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
0000'C2    00000000'8F   CA  0255  1354        BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
      59   0000'C2       D0  025E  1355        MOVL    CAR(R2),R9              ; READ CAR
0000'C2    00000000'8F   CA  0263  1356        BICL    #BLKSND_COMD,DR(R2)     ; ALLOW CMI TRANSACTIONS
0000'C2    00000000'8F   C8  026C  1357        BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
0000'C2    00000000'8F   CA  0275  1358        BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
        57   59   E7 8F   78  027E  1359        ASHL    #-25,R9,R7              ; ISOLATE CMI COMMAND
      57   00000008'8F   CA  0283  1360        BICL    #BIT3!NIBBLE1!BYTE1!BYTE2!BYTE3,R7  ; CLEAR SIGN EXTENSION
               57   00   91  028A  1361        CMPB    #0,R7                   ; CHECK FOR CMI READ
                    1F   13  028D  1362        BEQL    20$                     ; BRANCH IF COMMAND IS OK
                             028F  1363        $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                             028F  1364                        ARRAY_MSG2,-
                             028F  1365                        PRINT_CMDER,-
                             028F  1366                        #0,R7
                         57  DD  028F                    PUSHL   R7
                         00  DD  0291                    PUSHL   #0
                 00000000'EF  DF  0293                    PUSHAL  PRINT_CMDER
                 00000000'EF  DF  0299                    PUSHAL  ARRAY_MSG2
                 00000000'EF  DD  029F                    PUSHL   LUN
                         01  DD  02A5                    PUSHL   #1
         00000000'9F     06  FB  02A7                    CALLS   #$$M, a#DS$ERRHARD
                             02AE  1367 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF41  CF   FA  02AE                    CALLG   10$, a#DS$CKLOOP
                             02B7  1368 $DS_ENDSUB
                             02B7       T8_S3_X:
00000000'9F   000000E4'EF   FA  02B7                    CALLG   $S$, a#DS$ENDSUB
```

```
                                02C2  1370 $DS_BGNSUB
                                02C2       T8_S4::
    00000000'9F   000000F0'EF   FA  02C2                    CALLG   $S$, a#DS$BGNSUB
                                02CD  1371 ;
                                02CD  1372 ; Subtest four performs a Read-From-Drive function (CMI Write) and
                                02CD  1373 ; then checks the CAR for the correct code (4).
                                02CD  1374 ;
                                02CD  1375
          52   00000000'EF    D0  02CD  1376          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
     0000'C2   00000000'8F    C8  02D4  1377 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
          56   00000000'EF    DE  02DD  1378          MOVAL   MIOBUFFER,R6            ; STORE BUFFER ADDRESS POINTER
           57    56    00'8F  78  02E4  1379          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
          57   00000000'8F    C8  02E9  1380          BISL    #VALID_BIT,R7           ; SET V-BIT IN MAP ENTRY
     5A   52   00000000'8F    C1  02F0  1381          ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
                    6A   57    D0  02F8  1382          MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
                    0000'C2    D4  02FB  1383          CLRL    VAR(R2)                 ; SELECT MAP ZERO
                0000'C2   02   CE  02FF  1384          MNEGL   #2,BCR(R2)              ; SET BCR FOR 2 BYTE TRANSFER
     0000'C2   00000000'8F    C8  0304  1385          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
          54   00000000'EF    D0  030D  1386          MOVL    DRIVE0,R4               ; GET ADDRESS OF DRIVE 0
          64   00000000'8F    D0  0314  1387          MOVL    #READ,(R4)              ; PERFORM CMI WRITE
     0000'C2   00000000'8F    C8  031B  1388          BISL    #SIM_OCC,DR(R2)         ; ASSERT OCCUPIED
     0000'C2   00000000'8F    C8  0324  1389          BISL    #BLKSND_COMD,DR(R2)     ; DO NOT ALLOW CMI TRANSACTIONS
     0000'C2   00000000'8F    C8  032D  1390          BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
     0000'C2   00000000'8F    CA  0336  1391          BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
               59   0000'C2    D0  033F  1392          MOVL    CAR(R2),R9              ; READ CAR
     0000'C2   00000000'8F    CA  0344  1393          BICL    #BLKSND_COMD,DR(R2)     ; ALLOW CMI TRANSACTIONS
     0000'C2   00000000'8F    C8  034D  1394          BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
     0000'C2   00000000'8F    CA  0356  1395          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
           57    59    E7 8F    78  035F  1396          ASHL    #-25,R9,R7              ; ISOLATE CMI COMMAND
          57   00000008'8F    CA  0364  1397          BICL    #BIT3!NIBBLE1!BYTE1!BYTE2!BYTE3,R7  ; CLEAR SIGN EXTENSION
                    57   04    91  036B  1398          CMPB    #4,R7                   ; CHECK FOR CMI WRITE
                         1F    13  036E  1399          BEQL    20$                     ; BRANCH IF COMMAND IS OK
                                   0370  1400          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                   0370  1401                          ARRAY_MSG2,-
                                   0370  1402                          PRINT_CMDER,-
                                   0370  1403                          #4,R7
                         57    DD  0370                      PUSHL   R7
                         04    DD  0372                      PUSHL   #4
                   00000000'EF   DF  0374                      PUSHAL  PRINT_CMDER
                   00000000'EF   DF  037A                      PUSHAL  ARRAY_MSG2
                   00000000'EF   DD  0380                      PUSHL   LUN
                         01    DD  0386                      PUSHL   #1
               00000000'9F   06    FB  0388                      CALLS   #$$M, a#DS$ERRHARD
                                   038F  1404 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF41 CF    FA  038F                      CALLG   10$, a#DS$CKLOOP
                                   0398  1405 $DS_ENDSUB
                                   0398       T8_S4_X:
00000000'9F   000000F0'EF   FA  0398                      CALLG   $S$, a#DS$ENDSUB
```

```
                                  03A3  1407 $DS_ENDTEST
                  50   01   D0    03A3                         MOVL    #1, R0              ; NORMAL EXIT
                                  03A6       TEST_008_X::
        00000000'9F   6E   FA     03A6                         CALLG   (SP), a#DS$BREAK
                             04   03AD                         RET                         ; RETURN TO TEST SEQUENCER
                                  03AE  1408 $DS_PAGE
                                  03AE                .SBTTL   TEST 9:  MBDIB DIAGNOSTIC REGISTER TEST
                             00000000                         .PSECT  TEST_009, PAGE, NOWRT
                                  0024
                                  0024  1410 $DS_BGNTEST       <DEFAULT,ALL>,,LONG
                                  0024       DATA_009:
                        00000000  0024                         .LONG   0                   ; TEST ARGUMENT TABLE TERMINATOR
                                  0028       TEST_009::
                            0000  0028                         .WORD    M<>                ; ENTRY MASK
                                  002A  1411 ;++
                                  002A  1412 ; TEST DESCRIPTION:
                                  002A  1413 ;
                                  002A  1414 ;       THIS TEST CHECKS THE INTEGRITY OF THE RH750 DATA PATHS.
                                  002A  1415 ;
                                  002A  1416 ; TEST ALGORITHM:
                                  002A  1417 ;
                                  002A  1418 ;       WRITE ALL ZERO'S, ALL ONE'S, A FLOATING ZERO, A FLOATING
                                  002A  1419 ;       ONE, AND THE WORST CASE PATTERNS F00F, AND A55A.
                                  002A  1420 ;       ONTO THE RH750 DATA PATHS BY PLACING THE RH INTO MAINTENANCE
                                  002A  1421 ;       MODE.  ALL TRANSFERS WILL BE 512 BYTES IN SIZE, EXCEPT FOR
                                  002A  1422 ;       THE SECOND SUBTEST,WHICH IS ONLY EIGHT BYTES IN SIZE.
                                  002A  1423 ;--
                                  002A  1424 $DS_BGNSUB
                                  002A       T9_S1::
  00000000'9F   000000FC'EF   FA  002A                         CALLG   $S$, a#DS$BGNSUB
                                  0035  1425 ;++
                                  0035  1426 ;
                                  0035  1427 ; ALL ZERO'S
                                  0035  1428 ;
                                  0035  1429 ;--
                      FFC8'   30  0035  1430          BSBW    RH11TB_PRES                 ; CHECK FOR RH11TB
          52   00000000'EF   D0   0038  1431          MOVL    RH_CUR_ADR,R2               ; MOVE ADDRESS OF RH750 INTO R2
     59   52   00000000'8F   C1   003F  1432          ADDL3   #MAP_OFFSET,R2,R9           ; R9 = ADDRESS OF MAP REGISTERS
                                  0047  1433          ERRPREP RHDR_MSG,5,FMT_DIAG_REG     ; PREPARE TO HANDLE ERROR
  00000000'EF   00000000'EF   DE  0047                         MOVAL   RHDR_MSG,REG_NAME
          00000000'EF   05   9A   0052                         MOVZBL  #5,REG_NO
  00000000'EF   00000000'EF   DE  0059                         MOVAL   FMT_DIAG_REG,REG_STRING
     0000'C2   00000000'8F   C8   0064  1434 10$:    BISL    #PGM_INIT,CR(R2)            ; INIT RH
                         50   D4   006D  1435          CLRL    R0                          ; CLEAR INDEX
              00000000'EF40   D4   006F  1436 12$:    CLRL    DATA_OBUFFER R0             ; FILL WRITE BUFFER WITH ZERO'S
        00000000'EF40   00   D2   0076  1437          MCOML   #0,DATA_IBUFFER R0          ; FILL READ BUFFER WITH ONE'S
       E9 50   0000007F 8F   F3   007E  1438          AOBLEQ  #127,R0,12$                 ; DO FOR ENTIRE BUFFER
          5B   00000000'EF   DE   0086  1439          MOVAL   DATA_OBUFFER,R11            ; GET ADDRESS OF BUFFER
           5A   5B   F7 8F   78   008D  1440          ASHL    #-9,R11,R10                 ; PUT PAGE FRAME # INTO R10
          5A   00000000'8F   C8   0092  1441          BISL    #VALID_BIT,R10              ; SET V BIT
                   69   5A   D0   0099  1442          MOVL    R10,(R9)                    ; WRITE PFN INTO MAP 0
          5B   00000000'8F   CA   009C  1443          BICL    #MAP_PTR_MSK,R11            ; MASK  OUT PFN TO OBTAIN BYTE ADDRESS
              0000'C2   5B   D0   00A3  1444          MOVL    R11,VAR(R2)                 ; WRITE VIRTUAL ADDRESS REGISTER
                                  00A8  1445                                              ; SELECTS MAP 0
     0000'C2   00000200 8F   CE   00A8  1446          MNEGL   #512,BCR(R2)                ; SET BYTE COUNT = 512
          56   00000000'EF   D0   00B1  1447          MOVL    DRIVE0,R6                   ; GET ADDRESS OF DRIVE 0
     0000'C2   00000000'8F   C8   00B8  1448          BISL    #MAINT_MODE,CR(R2)          ; PUT RH750 INTO MAINTENANCE MODE
```

```
                66   00000000'8F   D0   00C1  1449          MOVL    #WRITE,(R6)                ; ISSUE WRITE TO DRIVE
          0000'C2    00000000'8F   D0   00C8  1450          MOVL    #SIM_OCC,DR(R2)            ; ASSERT OCCUPIED
                                   00D1  1451          $DS_WAITUS_S    #10                    ; WAIT 100 MICRO SECONDS
                              00   DD   00D1                        PUSHL   #0
                              0A   DD   00D3                        PUSHL   #10
                00000000'9F   02   FB   00D5                        CALLS   #2, a#DS$WAITUS
                              5A   D4   00DC  1452          CLRL    R10                        ; INIT INDEX
          0000'C2    00000000'8F   C8   00DE  1453 15$:     BISL    #SIM_SCLK,DR(R2)           ; ASSERT SCLK
          0000'C2    00000000'8F   CA   00E7  1454          BICL    #SIM_SCLK,DR(R2)           ; DEASSERT SCLK
          0000'C2    00000000'8F   C8   00F0  1455          BISL    #MBDIB_SEL,DR(R2)          ; SELECT THE MBDIB
             0000'CA    0000'C2    B0   00F9  1456          MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                              5A   D6   0100  1457          INCL    R10                        ; SETUP R10 FOR AOBLSS
            D4 5A    000001FF 8F   F2   0102  1458          AOBLSS  #511,R10,15$               ; DO FOR ALL 512 BYTES
                                   010A  1459
          0000'C2    00000000'8F   C8   010A  1460          BISL    #SIM_EBL,DR(R2)            ; SET END OF BLOCK
          0000'C2    00000000'8F   CA   0113  1461          BICL    #SIM_EBL,DR(R2)            ; CLEAR END OF BLOCK
                                   011C  1462
                                   011C  1463          ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
    00000000'EF    00000000'EF   DE   011C                        MOVAL   RHSR_MSG,REG_NAME
             00000000'EF   02   9A   0127                        MOVZBL  #2,REG_NO
    00000000'EF    00000000'EF   DE   012E                        MOVAL   FMT_STATUS_REG,REG_STRING
             54    00000000'8F   D0   0139  1464          MOVL    #DATA_XFER_DONE,R4         ; SET UP EXPECTED STATUS
             54    00000000'EF   C8   0140  1465          BISL    RH11TB_MSK,R4              ; LOAD EXPECTED MASK BIT
                53    0000'C2    D0   0147  1466          MOVL    SR(R2),R3                  ; READ STATUS REGISTER
                      54   53    D1   014C  1467          CMPL    R3,R4                      ; IS STATUS AS EXPECTED
                           1B    13   014F  1468          BEQL    30$                        ; BRANCH IF STATUS IS OK
                                   0151  1469          $DS_ERRHARD_S   ,LUN,-                 ; PRINT ERROR
                                   0151  1470                          ARRAY_MSG62,-
                                   0151  1471                          PRINT_SBE
                   00000000'EF   DF   0151                        PUSHAL  PRINT_SBE
                   00000000'EF   DF   0157                        PUSHAL  ARRAY_MSG62
                   00000000'EF   DD   015D                        PUSHL   LUN
                              01   DD   0163                        PUSHL   #$ER
                                   0165              ;;; TEST 9, SUBTEST 1, ERROR 1
                00000000'9F   04   FB   0165                        CALLS   #$$M, a#DS$ERRHARD
                                   016C  1472 30$:     $DS_CKLOOP      10$                    ; SCOPE LOOP
    00000000'9F    FEF4 CF   FA   016C                        CALLG   10$, a#DS$CKLOOP
                                   0175  1473
                              55   D4   0175  1474          CLRL    R5                         ; CLEAR INDEX
                              59   D4   0177  1475          CLRL    R9                         ; CLEAR INDEX
                56    00000000'EF   DE   0179  1476          MOVAL   DATA_OBUFFER,R6            ; GET ADDRESS OF EXPECTED DATA
                57    00000000'EF   DE   0180  1477          MOVAL   DATA_IBUFFER,R7            ; GET ADDRESS OF RECEIVED DATA
             54    00000000'EF49   90   0187  1478 40$:     MOVB    DATA_OBUFFER R9 ,R4        ; EXPECTED DATA
             53    00000000'EF49   90   018F  1479          MOVB    DATA_IBUFFER R9 ,R3        ; RECEIVED DATA
                      54   53    91   0197  1480          CMPB    R3,R4                      ; EXPECTED = RECEIVED?
                           3B    13   019A  1481          BEQL    50$                        ; BRANCH IF SO
                           55    D6   019C  1482          INCL    R5                         ; ADD ONE TO ERROR COUNTER
                                   019E  1483          $DS_BNOPER      43$                    ; PRINT ONLY 8 ERRORS IF NO OPER
                              0C   E1   019E                        BBC     #DSA$V_OPER,-      ; BR IF NO OPERATOR
                08 0000FE00 9F         01A0                        a#DSA$GL_FLAGS, 43$
                                   01A6  1484          $DS_BNQUICK     45$                    ; PRINT ALL ERRORS IF NOT QUICK
                              08   E1   01A6                        BBC     #DSA$V_QUICK,-     ; BR IF NOT QUICK
                05 0000FE00 9F         01A8                        a#DSA$GL_FLAGS, 45$
                      55    08   D1   01AE  1485 43$:     CMPL    #8,R5                      ; PRINTED MAX NUMBER OF ERRORS?
                           24   19   01B1  1486          BLSS    50$                        ; BRANCH IF SO
                                   01B3  1487 45$:     $DS_ERRHARD_S   ,LUN,-                 ; PRINT ERROR
                                   01B3  1488                          ARRAY_MSG1,-
```

```
                                01B3  1489                          PRINT_DTERR
            00000000'EF    DF   01B3                        PUSHAL  PRINT_DTERR
            00000000'EF    DF   01B9                        PUSHAL  ARRAY_MSG1
            00000000'EF    DD   01BF                        PUSHL   LUN
                      02   DD   01C5                        PUSHL   #$ER
                                01C7        ;;; TEST 9, SUBTEST 1, ERROR 2
      00000000'9F     04   FB   01C7                        CALLS   #$$M, a#DS$ERRHARD
                                01CE  1490          $DS_CKLOOP      10$                 ; SCOPE LOOP?
  00000000'9F    FE92 CF   FA   01CE                        CALLG   10$, a#DS$CKLOOP
    A8 59    000001FF 8F   F3   01D7  1491 50$:     AOBLEQ  #511,R9,40$                 ; DO FOR ALL BYTES
                      55   D5   01DF  1492          TSTL    R5                          ; ANY ERRORS REPORTED?
                      0F   13   01E1  1493          BEQL    60$                         ; BRANCH IF NONE
                                01E3  1494          $DS_PRINTB_S    FMT_ERRSUM,R5       ; REPORT NUMBER OF ERRORS
                      55   DD   01E3                        PUSHL   R5
            00000000'EF    9F   01E5                        PUSHAB  FMT_ERRSUM
      00000000'9F     02   FB   01EB                        CALLS   #$$N, a#DS$PRINTB
                                01F2  1495 60$:
                                01F2  1496 $DS_ENDSUB
                                01F2        T9_S1_X:
00000000'9F    000000FC'EF   FA   01F2                        CALLG   $S$, a#DS$ENDSUB
```

```
                                01FD  1498 $DS_BGNSUB
                                01FD       T9_S2::
00000000'9F   00000108'EF   FA   01FD                 CALLG   $S$, a#DS$BGNSUB
                                0208  1499 ;++
                                0208  1500 ; EIGHT-BYTE FLOATING ZERO
                                0208  1501 ;--
        52   00000000'EF    D0   0208  1502          MOVL    RH_CUR_ADR,R2          ; MOVE ADDRESS OF RH750 INTO R2
    59  52   00000000'8F    C1   020F  1503          ADDL3   #MAP_OFFSET,R2,R9      ; R9 = ADDRESS OF MAP REGISTERS
                                0217  1504          ERRPREP RHDR_MSG,5,FMT_DIAG_REG ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE   0217                MOVAL   RHDR_MSG,REG_NAME
        00000000'EF   05    9A   0222                MOVZBL  #5,REG_NO
00000000'EF   00000000'EF   DE   0229                MOVAL   FMT_DIAG_REG,REG_STRING
    0000'C2   00000000'8F   C8   0234  1505 10$:     BISL    #PGM_INIT,CR(R2)       ; INIT RH
                     50     D4   023D  1506          CLRL    R0                     ; CLEAR INDEX
                  51   01    D2   023F  1507          MCOML   #1,R1                  ; SET UP TO FLOAT A ZERO
      00000000'EF40   51    90   0242  1508 12$:     MOVB    R1,DATA_OBUFFER R0     ; PUT PATTERN IN WRITE BUFFER
          00000000'EF40     94   024A  1509          CLRB    DATA_IBUFFER R0        ; FILL READ BUFFER WITH ZERO'S
        51   51   01       9C   0251  1510          ROTL    #1,R1,R1               ; FLOAT PATTERN LEFT ONCE
          E9 50   07       F3   0255  1511          AOBLEQ  #7,R0,12$              ; DO FOR EIGHT BYTES
     5B   00000000'EF      DE   0259  1512          MOVAL   DATA_OBUFFER,R11       ; GET ADDRESS OF BUFFER
      5A   5B   F7 8F      78   0260  1513          ASHL    #-9,R11,R10            ; PUT PAGE FRAME # INTO R10
     5A   00000000'8F      C8   0265  1514          BISL    #VALID_BIT,R10         ; SET V BIT
                  69   5A   D0   026C  1515          MOVL    R10,(R9)               ; WRITE PFN INTO MAP 0
     5B   00000000'8F      CA   026F  1516          BICL    #MAP_PTR_MSK,R11       ; MASK  OUT PFN TO OBTAIN BYTE ADDRESS
        0000'C2   5B       D0   0276  1517          MOVL    R11,VAR(R2)            ; WRITE VIRTUAL ADDRESS REGISTER
                                027B  1518                                         ; SELECTS MAP 0
        0000'C2   08       CE   027B  1519          MNEGL   #8,BCR(R2)             ; SET BYTE COUNT = 8
    56   00000000'EF      D0   0280  1520          MOVL    DRIVE0,R6              ; GET ADDRESS OF DRIVE 0
0000'C2   00000000'8F      C8   0287  1521          BISL    #MAINT_MODE,CR(R2)     ; PUT RH750 INTO MAINTENANCE MODE
    66   00000000'8F      D0   0290  1522          MOVL    #WRITE,(R6)            ; ISSUE WRITE TO DRIVE
0000'C2   00000000'8F      D0   0297  1523          MOVL    #SIM_OCC,DR(R2)        ; ASSERT OCCUPIED
                                02A0  1524          $DS_WAITUS_S    #10            ; WAIT 100 MICRO SECONDS
                     00    DD   02A0                PUSHL   #0
                     0A    DD   02A2                PUSHL   #10
     00000000'9F     02    FB   02A4                CALLS   #2, a#DS$WAITUS
                     5A    D4   02AB  1525          CLRL    R10                    ; INIT INDEX
0000'C2   00000000'8F      C8   02AD  1526 15$:     BISL    #SIM_SCLK,DR(R2)       ; ASSERT SCLK
0000'C2   00000000'8F      CA   02B6  1527          BICL    #SIM_SCLK,DR(R2)       ; DEASSERT SCLK
0000'C2   00000000'8F      C8   02BF  1528          BISL    #MBDIB_SEL,DR(R2)      ; SELECT THE MBDIB
    0000'CA   0000'C2      B0   02C8  1529          MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                     5A    D6   02CF  1530          INCL    R10                    ; SETUP R10 FOR AOBLSS
          D8 5A    07      F2   02D1  1531          AOBLSS  #7,R10,15$             ; DO FOR ALL 8 BYTES
                                02D5  1532
0000'C2   00000000'8F      C8   02D5  1533          BISL    #SIM_EBL,DR(R2)        ; SET END OF BLOCK
0000'C2   00000000'8F      CA   02DE  1534          BICL    #SIM_EBL,DR(R2)        ; CLEAR END OF BLOCK
                                02E7  1535
                                02E7  1536          ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE   02E7                MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A   02F2                MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE   02F9                MOVAL   FMT_STATUS_REG,REG_STRING
        54   00000000'8F    D0   0304  1537          MOVL    #DATA_XFER_DONE,R4     ; SET UP EXPECTED STATUS
        54   00000000'EF    C8   030B  1538          BISL    RH11TB_MSK,R4          ; LOAD EXPECTED MASK BIT
            53   0000'C2    D0   0312  1539          MOVL    SR(R2),R3              ; READ STATUS REGISTER
                 54   53    D1   0317  1540          CMPL    R3,R4                  ; IS STATUS AS EXPECTED
                      1B    13   031A  1541          BEQL    30$                    ; BRANCH IF STATUS IS OK
                                031C  1542          $DS_ERRHARD_S    ,LUN,-        ; PRINT ERROR
                                031C  1543                           ARRAY_MSG62,-
```

```
                                  031C  1544                          PRINT_SBE
          00000000'EF    DF       031C                        PUSHAL  PRINT_SBE
          00000000'EF    DF       0322                        PUSHAL  ARRAY_MSG62
          00000000'EF    DD       0328                        PUSHL   LUN
                   01    DD       032E                        PUSHL   #$ER
                                  0330        ;;; TEST 9, SUBTEST 2, ERROR 1
     00000000'9F   04    FB       0330                        CALLS   #$$M, a#DS$ERRHARD
                                  0337  1545 30$:    $DS_CKLOOP      10$                 ; SCOPE LOOP
  00000000'9F   FEF9 CF    FA       0337                        CALLG   10$, a#DS$CKLOOP
                                  0340  1546
                   55    D4       0340  1547         CLRL    R5                          ; CLEAR INDEX
                   59    D4       0342  1548         CLRL    R9                          ; CLEAR INDEX
      56   00000000'EF    DE       0344  1549         MOVAL   DATA_OBUFFER,R6             ; GET ADDRESS OF EXPECTED DATA
      57   00000000'EF    DE       034B  1550         MOVAL   DATA_IBUFFER,R7             ; GET ADDRESS OF RECEIVED DATA
   54   00000000'EF49    90       0352  1551 40$:    MOVB    DATA_OBUFFER R9 ,R4         ; EXPECTED DATA
   53   00000000'EF49    90       035A  1552         MOVB    DATA_IBUFFER R9 ,R3         ; RECEIVED DATA
             54   53    91       0362  1553         CMPB    R3,R4                       ; EXPECTED = RECEIVED?
                   2E    13       0365  1554         BEQL    50$                         ; BRANCH IF SO
                   55    D6       0367  1555         INCL    R5                          ; ADD ONE TO ERROR COUNTER
                                  0369  1556         $DS_BNQUICK     45$                 ; PRINT ALL ERRORS IF NOT QUICK
                   08    E1       0369                        BBC     #DSA$V_QUICK,-      ; BR IF NOT QUICK
        00 0000FE00 9F            036B                                a#DSA$GL_FLAGS, 45$
                                  0371  1557 45$:    $DS_ERRHARD_S   ,LUN,-              ; PRINT ERROR
                                  0371  1558                          ARRAY_MSG24,-
                                  0371  1559                          PRINT_DTERR
          00000000'EF    DF       0371                        PUSHAL  PRINT_DTERR
          00000000'EF    DF       0377                        PUSHAL  ARRAY_MSG24
          00000000'EF    DD       037D                        PUSHL   LUN
                   02    DD       0383                        PUSHL   #$ER
                                  0385        ;;; TEST 9, SUBTEST 2, ERROR 2
     00000000'9F   04    FB       0385                        CALLS   #$$M, a#DS$ERRHARD
                                  038C  1560         $DS_CKLOOP      10$                 ; SCOPE LOOP?
  00000000'9F   FEA4 CF    FA       038C                        CALLG   10$, a#DS$CKLOOP
             B9 59   07    F3       0395  1561 50$:    AOBLEQ  #7,R9,40$                   ; DO FOR ALL BYTES
                   55    D5       0399  1562         TSTL    R5                          ; ANY ERRORS REPORTED?
                   0F    13       039B  1563         BEQL    60$                         ; BRANCH IF NONE
                                  039D  1564         $DS_PRINTB_S    FMT_ERRSUM,R5       ; REPORT NUMBER OF ERRORS
                   55    DD       039D                        PUSHL   R5
          00000000'EF    9F       039F                        PUSHAB  FMT_ERRSUM
     00000000'9F   02    FB       03A5                        CALLS   #$$N, a#DS$PRINTB
                                  03AC  1565 60$:
                                  03AC  1566 $DS_ENDSUB
                                  03AC        T9_S2_X:
00000000'9F   00000108'EF    FA   03AC                        CALLG   $S$, a#DS$ENDSUB
```

```
                                03B7  1568 $DS_BGNSUB
                                03B7       T9_S3::
00000000'9F   00000114'EF   FA  03B7                    CALLG   $S$, a#DS$BGNSUB
                                03C2  1569 ;++
                                03C2  1570 ; ALL ONE'S
                                03C2  1571 ;--
                     50   D4   03C2  1572         CLRL    R0                      ; INITIALIZE INDEX
      00000000'EF40   01   CE   03C4  1573 10$:    MNEGL   #1,DATA_OBUFFER R0      ; PUT ALL ONE'S IN WRITE BUFFER
           00000000'EF40   D4   03CC  1574         CLRL    DATA_IBUFFER R0         ; PUT ZERO'S INTO READ BUFFER
      E9 50   0000007F 8F   F3   03D3  1575         AOBLEQ  #127,R0,10$             ; FILL UP ENTIRE BUFFER
                                03DB  1576         ERRPREP RHDR_MSG,5,FMT_DIAG_REG ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE   03DB                    MOVAL   RHDR_MSG,REG_NAME
      00000000'EF   05   9A   03E6                    MOVZBL  #5,REG_NO
00000000'EF   00000000'EF   DE   03ED                    MOVAL   FMT_DIAG_REG,REG_STRING
    0000'C2   00000000'8F   C8   03F8  1577 20$:    BISL    #PGM_INIT,CR(R2)        ; INIT RH
    0000'C2   00000000'8F   C8   0401  1578         BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 IN MAINTENANCE MODE
          0000'C2   5B   D0   040A  1579         MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
    0000'C2   00000200 8F   CE   040F  1580         MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT
         56   00000000'EF   D0   0418  1581         MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
         66   00000000'8F   D0   041F  1582         MOVL    #WRITE,(R6)             ; ISSUE WRITE TO DRIVE
    0000'C2   00000000'8F   D0   0426  1583         MOVL    #SIM_OCC,DR(R2)         ; ASSERT OCCUPIED
                                042F  1584         $DS_WAITUS_S    #10             ; WAIT FOR DATA TO FILL SILO
                     00   DD   042F                    PUSHL   #0
                     0A   DD   0431                    PUSHL   #10
      00000000'9F    02   FB   0433                    CALLS   #2, a#DS$WAITUS
                     5A   D4   043A  1585         CLRL    R10                     ; INIT INDEX
    0000'C2   00000000'8F   C8   043C  1586 25$:    BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
    0000'C2   00000000'8F   CA   0445  1587         BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
    0000'C2   00000000'8F   C8   044E  1588         BISL    #MBDIB_SEL,DR(R2)       ; SELECT THE MBDIB
       0000'CA   0000'C2    B0   0457  1589         MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                     5A   D6   045E  1590         INCL    R10                     ; SETUP R10 FOR AOBLSS
      D4 5A   000001FF 8F   F2   0460  1591         AOBLSS  #511,R10,25$            ; DO FOR ALL 512 BYTES
                                0468  1592
    0000'C2   00000000'8F   C8   0468  1593         BISL    #SIM_EBL,DR(R2)         ; SET END OF BLOCK
    0000'C2   00000000'8F   CA   0471  1594         BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                047A  1595
                                047A  1596         ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE   047A                    MOVAL   RHSR_MSG,REG_NAME
         00000000'EF   02   9A   0485                    MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE   048C                    MOVAL   FMT_STATUS_REG,REG_STRING
         54   00000000'8F   D0   0497  1597         MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
         54   00000000'EF   C8   049E  1598         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0   04A5  1599         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                   54   53   D1   04AA  1600         CMPL    R3,R4                   ; IS STATUS AS EXPECTED
                        1B   13   04AD  1601         BEQL    30$                     ; BRANCH IF STATUS IS OK
                                04AF  1602         $DS_ERRHARD_S   ,LUN,-          ; PRINT ERROR
                                04AF  1603                         ARRAY_MSG62,-
                                04AF  1604                         PRINT_SBE
              00000000'EF   DF   04AF                    PUSHAL  PRINT_SBE
              00000000'EF   DF   04B5                    PUSHAL  ARRAY_MSG62
              00000000'EF   DD   04BB                    PUSHL   LUN
                     01   DD   04C1                    PUSHL   #$ER
                                04C3       ;;; TEST 9, SUBTEST 3, ERROR 1
      00000000'9F    04   FB   04C3                    CALLS   #$$M, a#DS$ERRHARD
                                04CA  1605 30$:    $DS_CKLOOP      20$             ; SCOPE LOOP
00000000'9F   FF2A CF   FA   04CA                    CALLG   20$, a#DS$CKLOOP
                                04D3  1606
```

```
                    55   D4   04D3  1607          CLRL    R5                          ; CLEAR INDEX
                    59   D4   04D5  1608          CLRL    R9                          ; CLEAR INDEX
                56  00000000'EF   DE   04D7  1609          MOVAL   DATA_OBUFFER,R6     ; GET ADDRESS OF EXPECTED DATA
                57  00000000'EF   DE   04DE  1610          MOVAL   DATA_IBUFFER,R7     ; GET ADDRESS OF RECEIVED DATA
             54  00000000'EF49   90   04E5  1611 40$:     MOVB    DATA_OBUFFER R9 ,R4 ; EXPECTED DATA
             53  00000000'EF49   90   04ED  1612          MOVB    DATA_IBUFFER R9 ,R3 ; RECEIVED DATA
                      54   53   91   04F5  1613          CMPB    R3,R4               ; EXPECTED = RECEIVED?
                    3B   13   04F8  1614          BEQL    50$                         ; BRANCH IF SO
                    55   D6   04FA  1615          INCL    R5                          ; ADD ONE TO ERROR COUNTER
                         04FC  1616          $DS_BNOPER      43$                     ; PRINT ONLY 8 ERRORS IF NO OPER
                    0C   E1   04FC                      BBC     #DSA$V_OPER,-       ; BR IF NO OPERATOR
                08 0000FE00 9F   04FE                              a#DSA$GL_FLAGS, 43$
                         0504  1617          $DS_BNQUICK     45$                     ; PRINT ALL ERRORS IF NOT QUICK
                    08   E1   0504                      BBC     #DSA$V_QUICK,-      ; BR IF NOT QUICK
                05 0000FE00 9F   0506                              a#DSA$GL_FLAGS, 45$
                      55   08   D1   050C  1618 43$:     CMPL    #8,R5               ; PRINTED MAX NUMBER OF ERRORS?
                    24   19   050F  1619          BLSS    50$                         ; BRANCH IF SO
                         0511  1620 45$:     $DS_ERRHARD_S   ,LUN,-                  ; PRINT ERROR
                         0511  1621                      ARRAY_MSG24,-
                         0511  1622                      PRINT_DTERR
                00000000'EF   DF   0511                      PUSHAL  PRINT_DTERR
                00000000'EF   DF   0517                      PUSHAL  ARRAY_MSG24
                00000000'EF   DD   051D                      PUSHL   LUN
                    02   DD   0523                      PUSHL   #$ER
                         0525              ;;; TEST 9, SUBTEST 3, ERROR 2
             00000000'9F   04   FB   0525                      CALLS   #$$M, a#DS$ERRHARD
                         052C  1623          $DS_CKLOOP      20$                     ; SCOPE LOOP?
          00000000'9F   FEC8 CF   FA   052C                      CALLG   20$, a#DS$CKLOOP
            A8 59   000001FF 8F   F3   0535  1624 50$:     AOBLEQ  #511,R9,40$         ; DO FOR ALL BYTES
                    55   D5   053D  1625          TSTL    R5                          ; ANY ERRORS REPORTED?
                    0F   13   053F  1626          BEQL    60$                         ; BRANCH IF NONE
                         0541  1627          $DS_PRINTB_S    FMT_ERRSUM,R5           ; REPORT NUMBER OF ERRORS
                    55   DD   0541                      PUSHL   R5
                00000000'EF   9F   0543                      PUSHAB  FMT_ERRSUM
             00000000'9F   02   FB   0549                      CALLS   #$$N, a#DS$PRINTB
                         0550  1628 60$:
                         0550  1629 $DS_ENDSUB
                         0550       T9_S3_X:
00000000'9F   00000114'EF   FA   0550                      CALLG   $S$, a#DS$ENDSUB
```

```
                                 055B  1631 $DS_BGNSUB
                                 055B       T9_S4::
00000000'9F   00000120'EF    FA  055B                    CALLG   $S$, a#DS$BGNSUB
                                 0566  1632 ;++
                                 0566  1633 ; FLOATING ONE
                                 0566  1634 ;--
                        50   D4  0566  1635         CLRL    R0                      ; CLEAR INDEX
                     51 01   D0  0568  1636         MOVL    #1,R1                   ; SET UP PATTERN
          00000000'EF40 51   90  056B  1637 10$:    MOVB    R1,DATA_OBUFFER R0      ; PUT PATTERN IN WRITE BUFFER
               00000000'EF40 94  0573  1638         CLRB    DATA_IBUFFER R0         ; PUT ZERO'S IN READ BUFFER
              51 51 01   9C  057A  1639         ROTL    #1,R1,R1                ; FLOAT PATTERN LEFT ONCE
        E5 50  000001FF 8F   F3  057E  1640         AOBLEQ  #511,R0,10$             ; SET UP ENTIRE BUFFER
     0000'C2   00000000'8F   C8  0586  1641 20$:    BISL    #PGM_INIT,CR(R2)        ; INIT MUT
     0000'C2   00000000'8F   C8  058F  1642         BISL    #MAINT_MODE,CR(R2)      ; PUT MUT INTO MAINTENANCE MODE
               0000'C2   5B  D0  0598  1643         MOVL    R11,VAR(R2)             ; POINT VIRTUAL ADDRESS REGISTER
                                 059D  1644                                         ; TO WRITE BUFFER
     0000'C2   00000200 8F   CE  059D  1645         MNEGL   #512,BCR(R2)            ; SET RH750 BYTE COUNT TO 512
          56   00000000'EF   D0  05A6  1646         MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
          66   00000000'8F   D0  05AD  1647         MOVL    #WRITE,(R6)             ; ISSUE WRITE TO DRIVE
     0000'C2   00000000'8F   D0  05B4  1648         MOVL    #SIM_OCC,DR(R2)         ; ASSERT MASSBUS OCCUPIED
                                 05BD  1649         $DS_WAITUS_S    #10             ; WAIT 100 MICRO SECONDS
                        00   DD  05BD                    PUSHL   #0
                        0A   DD  05BF                    PUSHL   #10
          00000000'9F   02   FB  05C1                    CALLS   #2, a#DS$WAITUS
                        5A   D4  05C8  1650         CLRL    R10                     ; INIT INDEX
     0000'C2   00000000'8F   C8  05CA  1651 25$:    BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
     0000'C2   00000000'8F   CA  05D3  1652         BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
     0000'C2   00000000'8F   C8  05DC  1653         BISL    #MBDIB_SEL,DR(R2)       ; SELECT THE MBDIB
          0000'CA   0000'C2  B0  05E5  1654         MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                        5A   D6  05EC  1655         INCL    R10                     ; SETUP R10 FOR AOBLSS
       D4 5A   000001FF 8F   F2  05EE  1656         AOBLSS  #511,R10,25$            ; DO FOR ALL 512 BYTES
                                 05F6  1657
     0000'C2   00000000'8F   C8  05F6  1658         BISL    #SIM_EBL,DR(R2)         ; SET END OF BLOCK
     0000'C2   00000000'8F   CA  05FF  1659         BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                 0608  1660
                                 0608  1661         ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF    00000000'EF   DE  0608                    MOVAL   RHSR_MSG,REG_NAME
          00000000'EF    02   9A  0613                    MOVZBL  #2,REG_NO
00000000'EF    00000000'EF   DE  061A                    MOVAL   FMT_STATUS_REG,REG_STRING
          54   00000000'8F   D0  0625  1662         MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
          54   00000000'EF   C8  062C  1663         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
               53   0000'C2  D0  0633  1664         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                    54   53  D1  0638  1665         CMPL    R3,R4                   ; IS STATUS AS EXPECTED
                         1B  13  063B  1666         BEQL    30$                     ; BRANCH IF STATUS IS OK
                                 063D  1667         $DS_ERRHARD_S   ,LUN,-          ; PRINT ERROR
                                 063D  1668                         ARRAY_MSG62,-
                                 063D  1669                         PRINT_SBE
               00000000'EF   DF  063D                    PUSHAL  PRINT_SBE
               00000000'EF   DF  0643                    PUSHAL  ARRAY_MSG62
               00000000'EF   DD  0649                    PUSHL   LUN
                         01  DD  064F                    PUSHL   #$ER
                                 0651       ;;; TEST 9, SUBTEST 4, ERROR 1
          00000000'9F    04   FB  0651                    CALLS   #$$M, a#DS$ERRHARD
                                 0658  1670 30$:    $DS_CKLOOP      20$             ; SCOPE LOOP
     00000000'9F    FF2A CF   FA  0658                    CALLG   20$, a#DS$CKLOOP
                                 0661  1671
                         55  D4  0661  1672         CLRL    R5                      ; CLEAR INDEX
```

```
                    59   D4  0663  1673          CLRL    R9                              ; CLEAR INDEX
          56  00000000'EF   DE  0665  1674          MOVAL   DATA_OBUFFER,R6             ; GET ADDRESS OF EXPECTED DATA
          57  00000000'EF   DE  066C  1675          MOVAL   DATA_IBUFFER,R7             ; GET ADDRESS OF RECEIVED DATA
       54  00000000'EF49    90  0673  1676 40$:     MOVB    DATA_OBUFFER R9 ,R4         ; EXPECTED DATA
       53  00000000'EF49    90  067B  1677          MOVB    DATA_IBUFFER R9 ,R3         ; RECEIVED DATA
             54   53        91  0683  1678          CMPB    R3,R4                       ; EXPECTED = RECEIVED?
                  3B        13  0686  1679          BEQL    50$                         ; BRANCH IF SO
                  55        D6  0688  1680          INCL    R5                          ; ADD ONE TO ERROR COUNTER
                            068A  1681          $DS_BNOPER      43$                     ; PRINT ONLY 8 ERRORS IF NO OPER
                  0C        E1  068A                        BBC     #DSA$V_OPER,-       ; BR IF NO OPERATOR
        08 0000FE00 9F          068C                        a#DSA$GL_FLAGS, 43$
                            0692  1682          $DS_BNQUICK     45$                     ; PRINT ALL ERRORS IF NOT QUICK
                  08        E1  0692                        BBC     #DSA$V_QUICK,-      ; BR IF NOT QUICK
        05 0000FE00 9F          0694                        a#DSA$GL_FLAGS, 45$
             55   08        D1  069A  1683 43$:     CMPL    #8,R5                       ; PRINTED MAX NUMBER OF ERRORS?
                  24        19  069D  1684          BLSS    50$                         ; BRANCH IF SO
                            069F  1685 45$:     $DS_ERRHARD_S   ,LUN,-                  ; PRINT ERROR
                            069F  1686                      ARRAY_MSG24,-
                            069F  1687                      PRINT_DTERR
          00000000'EF       DF  069F                        PUSHAL  PRINT_DTERR
          00000000'EF       DF  06A5                        PUSHAL  ARRAY_MSG24
          00000000'EF       DD  06AB                        PUSHL   LUN
                  02        DD  06B1                        PUSHL   #$ER
                            06B3        ;;; TEST 9, SUBTEST 4, ERROR 2
       00000000'9F    04    FB  06B3                        CALLS   #$$M, a#DS$ERRHARD
                            06BA  1688          $DS_CKLOOP      20$                     ; SCOPE LOOP?
    00000000'9F   FEC8  CF    FA  06BA                        CALLG   20$, a#DS$CKLOOP
      A8 59   000001FF  8F    F3  06C3  1689 50$:     AOBLEQ  #511,R9,40$             ; DO FOR ALL BYTES
                  55    D5  06CB  1690          TSTL    R5                          ; ANY ERRORS REPORTED?
                  0F    13  06CD  1691          BEQL    60$                         ; BRANCH IF NONE
                            06CF  1692          $DS_PRINTB_S    FMT_ERRSUM,R5       ; REPORT NUMBER OF ERRORS
                  55    DD  06CF                        PUSHL   R5
          00000000'EF   9F  06D1                        PUSHAB  FMT_ERRSUM
       00000000'9F    02    FB  06D7                        CALLS   #$$N, a#DS$PRINTB
                            06DE  1693 60$:
                            06DE  1694 $DS_ENDSUB
                            06DE        T9_S4_X:
00000000'9F   00000120'EF   FA  06DE                        CALLG   $S$, a#DS$ENDSUB
```

```
                                    06E9  1696 $DS_BGNSUB
                                    06E9       T9_S5::
00000000'9F   0000012C'EF   FA   06E9                    CALLG   $S$, a#DS$BGNSUB
                                    06F4  1697 ;++
                                    06F4  1698 ; FLOATING ZERO
                                    06F4  1699 ;--
                      50    D4    06F4  1700         CLRL    R0                      ; INITIALIZE INDEX
                   51   01    D2    06F6  1701         MCOML   #1,R1                   ; SET UP TO FLOAT A ZERO
      00000000'EF40   51    90    06F9  1702 10$:    MOVB    R1,DATA_OBUFFER R0      ; PUT PATTERN IN WRITE BUFFER
           00000000'EF40    94    0701  1703         CLRB    DATA_IBUFFER R0         ; PUT ZERO'S IN READ BUFFER
          51   51   01    9C    0708  1704         ROTL    #1,R1,R1                ; FLOAT PATTERN LEFT ONCE
    E5 50    000001FF 8F    F3    070C  1705         AOBLEQ  #511,R0,10$             ; SET UP ENTIRE BUFFER
  0000'C2    00000000'8F    C8    0714  1706 20$:    BISL    #PGM_INIT,CR(R2)        ; INIT MUT
  0000'C2    00000000'8F    C8    071D  1707         BISL    #MAINT_MODE,CR(R2)      ; PUT IT INTO MAINTENANCE MODE
          0000'C2   5B    D0    0726  1708         MOVL    R11,VAR(R2)             ; POINT VAR TO WRITE BUFFER
  0000'C2    00000200 8F    CE    072B  1709         MNEGL   #512,BCR(R2)            ; SET UP BYTE COUNT
       56    00000000'EF    D0    0734  1710         MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
       66    00000000'8F    D0    073B  1711         MOVL    #WRITE,(R6)             ; ISSUE WRITE TO DRIVE
  0000'C2    00000000'8F    D0    0742  1712         MOVL    #SIM_OCC,DR(R2)         ; ASSERT MASSBUS OCCUPIED
                                    074B  1713         $DS_WAITUS_S    #10             ; WAIT FOR SILO TO FILL UP
                      00    DD    074B                    PUSHL   #0
                      0A    DD    074D                    PUSHL   #10
       00000000'9F    02    FB    074F                    CALLS   #2, a#DS$WAITUS
                      5A    D4    0756  1714         CLRL    R10                     ; INIT INDEX
  0000'C2    00000000'8F    C8    0758  1715 25$:    BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
  0000'C2    00000000'8F    CA    0761  1716         BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
  0000'C2    00000000'8F    C8    076A  1717         BISL    #MBDIB_SEL,DR(R2)       ; SELECT THE MBDIB
      0000'CA    0000'C2    B0    0773  1718         MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                      5A    D6    077A  1719         INCL    R10                     ; SETUP R10 FOR AOBLSS
    D4 5A    000001FF 8F    F2    077C  1720         AOBLSS  #511,R10,25$            ; DO FOR ALL 512 BYTES
                                    0784  1721
  0000'C2    00000000'8F    C8    0784  1722         BISL    #SIM_EBL,DR(R2)         ; SET END OF BLOCK
  0000'C2    00000000'8F    CA    078D  1723         BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                    0796  1724
                                    0796  1725         ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE    0796                    MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A    07A1                    MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE    07A8                    MOVAL   FMT_STATUS_REG,REG_STRING
        54    00000000'8F   D0    07B3  1726         MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
        54    00000000'EF   C8    07BA  1727         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
           53   0000'C2    D0    07C1  1728         MOVL    SR(R2),R3               ; READ STATUS REGISTER
              54   53    D1    07C6  1729         CMPL    R3,R4                   ; IS STATUS AS EXPECTED
                   1B    13    07C9  1730         BEQL    30$                     ; BRANCH IF STATUS IS OK
                                    07CB  1731         $DS_ERRHARD_S   ,LUN,-          ; PRINT ERROR
                                    07CB  1732                         ARRAY_MSG62,-
                                    07CB  1733                         PRINT_SBE
              00000000'EF   DF    07CB                    PUSHAL  PRINT_SBE
              00000000'EF   DF    07D1                    PUSHAL  ARRAY_MSG62
              00000000'EF   DD    07D7                    PUSHL   LUN
                   01    DD    07DD                    PUSHL   #$ER
                                    07DF        ;;; TEST 9, SUBTEST 5, ERROR 1
       00000000'9F    04    FB    07DF                    CALLS   #$$M, a#DS$ERRHARD
                                    07E6  1734 30$:    $DS_CKLOOP      20$             ; SCOPE LOOP
00000000'9F   FF2A CF    FA    07E6                    CALLG   20$, a#DS$CKLOOP
                                    07EF  1735
                      55    D4    07EF  1736         CLRL    R5                      ; CLEAR INDEX
                      59    D4    07F1  1737         CLRL    R9                      ; CLEAR INDEX
```

```
                56   00000000'EF   DE   07F3   1738          MOVAL   DATA_OBUFFER,R6          ; GET ADDRESS OF EXPECTED DATA
                57   00000000'EF   DE   07FA   1739          MOVAL   DATA_IBUFFER,R7          ; GET ADDRESS OF RECEIVED DATA
             54   00000000'EF49   90   0801   1740 40$:     MOVB    DATA_OBUFFER R9 ,R4      ; EXPECTED DATA
             53   00000000'EF49   90   0809   1741          MOVB    DATA_IBUFFER R9 ,R3      ; RECEIVED DATA
                       54   53   91   0811   1742          CMPB    R3,R4                    ; EXPECTED = RECEIVED?
                            3B   13   0814   1743          BEQL    50$                      ; BRANCH IF SO
                            55   D6   0816   1744          INCL    R5                       ; ADD ONE TO ERROR COUNTER
                                      0818   1745          $DS_BNOPER      43$              ; PRINT ONLY 8 ERRORS IF NO OPER
                                  0C  E1   0818                          BBC     #DSA$V_OPER,-   ; BR IF NO OPERATOR
                     08 0000FE00 9F       081A                          a#DSA$GL_FLAGS, 43$
                                      0820   1746          $DS_BNQUICK     45$              ; PRINT ALL ERRORS IF NOT QUICK
                                  08  E1   0820                          BBC     #DSA$V_QUICK,-  ; BR IF NOT QUICK
                     05 0000FE00 9F       0822                          a#DSA$GL_FLAGS, 45$
                           55   08   D1   0828   1747 43$:     CMPL    #8,R5                    ; PRINTED MAX NUMBER OF ERRORS?
                                24   19   082B   1748          BLSS    50$                      ; BRANCH IF SO
                                      082D   1749 45$:     $DS_ERRHARD_S   ,LUN,-           ; PRINT ERROR
                                      082D   1750                          ARRAY_MSG24,-
                                      082D   1751                          PRINT_DTERR
                      00000000'EF   DF   082D                          PUSHAL  PRINT_DTERR
                      00000000'EF   DF   0833                          PUSHAL  ARRAY_MSG24
                      00000000'EF   DD   0839                          PUSHL   LUN
                                02   DD   083F                          PUSHL   #$ER
                                      0841       ;;; TEST 9, SUBTEST 5, ERROR 2
                00000000'9F   04   FB   0841                          CALLS   #$$M, a#DS$ERRHARD
                                      0848   1752          $DS_CKLOOP      20$              ; SCOPE LOOP?
           00000000'9F   FEC8 CF   FA   0848                          CALLG   20$, a#DS$CKLOOP
             A8 59   000001FF 8F   F3   0851   1753 50$:     AOBLEQ  #511,R9,40$              ; DO FOR ALL BYTES
                            55   D5   0859   1754          TSTL    R5                       ; ANY ERRORS REPORTED?
                            0F   13   085B   1755          BEQL    60$                      ; BRANCH IF NONE
                                      085D   1756          $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                            55   DD   085D                          PUSHL   R5
                      00000000'EF   9F   085F                          PUSHAB  FMT_ERRSUM
                00000000'9F   02   FB   0865                          CALLS   #$$N, a#DS$PRINTB
                                      086C   1757 60$:
                                      086C   1758 $DS_ENDSUB
                                      086C       T9_S5_X:
00000000'9F   0000012C'EF   FA   086C                          CALLG   $S$, a#DS$ENDSUB
```

```
                                0877  1760 $DS_BGNSUB
                                0877       T9_S6::
00000000'9F   00000138'EF   FA  0877                 CALLG   $S$, a#DS$BGNSUB
                                0882  1761 ;++
                                0882  1762 ; F00F
                                0882  1763 ;--
                     50   D4    0882  1764       CLRL    R0                      ; INIT INDEX
          51   0000F00F 8F   D0    0884  1765       MOVL    # XF00F,R1              ; SET UP PATTERN
     00000000'EF40   51   B0    088B  1766 10$:    MOVW    R1,DATA_OBUFFER R0      ; PUT PATTERN IN WRITE BUFFER
          00000000'EF40   B4    0893  1767       CLRW    DATA_IBUFFER R0         ; PUT ZERO'S IN READ BUFFER
   E9 50    000000FF 8F   F3    089A  1768       AOBLEQ  #255,R0,10$             ; FILL BUFFER
0000'C2    00000000'8F   C8    08A2  1769 20$:    BISL    #PGM_INIT,CR(R2)        ; INIT MUT
0000'C2    00000000'8F   C8    08AB  1770       BISL    #MAINT_MODE,CR(R2)      ; PUT IT INTO MAINTENANCE MODE
          0000'C2   5B   D0    08B4  1771       MOVL    R11,VAR(R2)             ; POINT VIRTUAL ADDRESS REGISTER
                                08B9  1772                                       ; TO WRITE BUFFER
0000'C2    00000200 8F   CE    08B9  1773       MNEGL   #512,BCR(R2)            ; SET BYTE COUNT TO 512
     56    00000000'EF   D0    08C2  1774       MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
     66    00000000'8F   D0    08C9  1775       MOVL    #WRITE,(R6)             ; ISSUE WRITE TO DRIVE
0000'C2    00000000'8F   D0    08D0  1776       MOVL    #SIM_OCC,DR(R2)         ; ASSERT MASSBUS OCCUPIED
                                08D9  1777       $DS_WAITUS_S    #10             ; WAIT FOR SILO TO FILL UP
                     00   DD    08D9                 PUSHL   #0
                     0A   DD    08DB                 PUSHL   #10
     00000000'9F     02   FB    08DD                 CALLS   #2, a#DS$WAITUS
                     5A   D4    08E4  1778       CLRL    R10                     ; INIT INDEX
0000'C2    00000000'8F   C8    08E6  1779 25$:    BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
0000'C2    00000000'8F   CA    08EF  1780       BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
0000'C2    00000000'8F   C8    08F8  1781       BISL    #MBDIB_SEL,DR(R2)       ; SELECT THE MBDIB
    0000'CA    0000'C2   B0    0901  1782       MOVW    DR(R2),DATA_IBUFFER(R10); PUT VALUE IN READ BUFFER
                     5A   D6    0908  1783       INCL    R10                     ; SETUP R10 FOR AOBLSS
  D4 5A    000001FF 8F   F2    090A  1784       AOBLSS  #511,R10,25$            ; DO FOR ALL 512 BYTES
                                0912  1785
0000'C2    00000000'8F   C8    0912  1786       BISL    #SIM_EBL,DR(R2)         ; SET END OF BLOCK
0000'C2    00000000'8F   CA    091B  1787       BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                0924  1788
                                0924  1789       ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE  0924                 MOVAL   RHSR_MSG,REG_NAME
     00000000'EF   02   9A    092F                 MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  0936                 MOVAL   FMT_STATUS_REG,REG_STRING
     54    00000000'8F   D0    0941  1790       MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
     54    00000000'EF   C8    0948  1791       BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
          53   0000'C2   D0    094F  1792       MOVL    SR(R2),R3               ; READ STATUS REGISTER
               54   53   D1    0954  1793       CMPL    R3,R4                   ; IS STATUS AS EXPECTED
                    1B   13    0957  1794       BEQL    30$                     ; BRANCH IF STATUS IS OK
                                0959  1795       $DS_ERRHARD_S   ,LUN,-          ; PRINT ERROR
                                0959  1796                       ARRAY_MSG62,-
                                0959  1797                       PRINT_SBE
          00000000'EF   DF    0959                 PUSHAL  PRINT_SBE
          00000000'EF   DF    095F                 PUSHAL  ARRAY_MSG62
          00000000'EF   DD    0965                 PUSHL   LUN
                     01   DD    096B                 PUSHL   #$ER
                                096D       ;;; TEST 9, SUBTEST 6, ERROR 1
     00000000'9F     04   FB    096D                 CALLS   #$$M, a#DS$ERRHARD
                                0974  1798 30$:    $DS_CKLOOP      20$             ; SCOPE LOOP
00000000'9F   FF2A CF   FA  0974                 CALLG   20$, a#DS$CKLOOP
                                097D  1799
                     55   D4    097D  1800       CLRL    R5                      ; CLEAR INDEX
                     59   D4    097F  1801       CLRL    R9                      ; CLEAR INDEX
```

```
                56   00000000'EF   DE  0981  1802          MOVAL   DATA_OBUFFER,R6          ; GET ADDRESS OF EXPECTED DATA
                57   00000000'EF   DE  0988  1803          MOVAL   DATA_IBUFFER,R7          ; GET ADDRESS OF RECEIVED DATA
             54   00000000'EF49    90  098F  1804 40$:     MOVB    DATA_OBUFFER R9 ,R4      ; EXPECTED DATA
             53   00000000'EF49    90  0997  1805          MOVB    DATA_IBUFFER R9 ,R3      ; RECEIVED DATA
                       54   53    91  099F  1806          CMPB    R3,R4                    ; EXPECTED = RECEIVED?
                            3B    13  09A2  1807          BEQL    50$                      ; BRANCH IF SO
                            55    D6  09A4  1808          INCL    R5                       ; ADD ONE TO ERROR COUNTER
                                      09A6  1809          $DS_BNOPER      43$              ; PRINT ONLY 8 ERRORS IF NO OPER
                            0C    E1  09A6                BBC     #DSA$V_OPER,-            ; BR IF NO OPERATOR
                  08 0000FE00 9F      09A8                        a#DSA$GL_FLAGS, 43$
                                      09AE  1810          $DS_BNQUICK     45$              ; PRINT ALL ERRORS IF NOT QUICK
                            08    E1  09AE                BBC     #DSA$V_QUICK,-           ; BR IF NOT QUICK
                  05 0000FE00 9F      09B0                        a#DSA$GL_FLAGS, 45$
                       55   08    D1  09B6  1811 43$:     CMPL    #8,R5                    ; PRINTED MAX NUMBER OF ERRORS?
                            24    19  09B9  1812          BLSS    50$                      ; BRANCH IF SO
                                      09BB  1813 45$:     $DS_ERRHARD_S   ,LUN,-           ; PRINT ERROR
                                      09BB  1814                  ARRAY_MSG24,-
                                      09BB  1815                  PRINT_DTERR
                   00000000'EF    DF  09BB                PUSHAL  PRINT_DTERR
                   00000000'EF    DF  09C1                PUSHAL  ARRAY_MSG24
                   00000000'EF    DD  09C7                PUSHL   LUN
                            02    DD  09CD                PUSHL   #$ER
                                      09CF          ;;; TEST 9, SUBTEST 6, ERROR 2
                00000000'9F   04    FB  09CF                CALLS   #$$M, a#DS$ERRHARD
                                      09D6  1816          $DS_CKLOOP      20$              ; SCOPE LOOP?
           00000000'9F   FEC8 CF    FA  09D6                CALLG   20$, a#DS$CKLOOP
             A8 59   000001FF 8F    F3  09DF  1817 50$:     AOBLEQ  #511,R9,40$              ; DO FOR ALL BYTES
                            55    D5  09E7  1818          TSTL    R5                       ; ANY ERRORS REPORTED?
                            0F    13  09E9  1819          BEQL    60$                      ; BRANCH IF NONE
                                      09EB  1820          $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                            55    DD  09EB                PUSHL   R5
                      00000000'EF   9F  09ED                PUSHAB  FMT_ERRSUM
                00000000'9F   02    FB  09F3                CALLS   #$$N, a#DS$PRINTB
                                      09FA  1821 60$:
                                      09FA  1822 $DS_ENDSUB
                                      09FA       T9_S6_X:
00000000'9F   00000138'EF   FA  09FA                CALLG   $S$, a#DS$ENDSUB
```

```
                                    0A05  1824 $DS_BGNSUB
                                    0A05       T9_S7::
00000000'9F   00000144'EF    FA     0A05                    CALLG   $S$, a#DS$BGNSUB
                                    0A10  1825 ;++
                                    0A10  1826 ; A55A
                                    0A10  1827 ;--
                        50   D4     0A10  1828         CLRL R0                          ; INIT INDEX
          51    0000A55A 8F   D0     0A12  1829         MOVL    # XA55A,R1               ; SET UP PATTERN
       00000000'EF40   51   B0     0A19  1830 10$:    MOVW    R1,DATA_OBUFFER R0       ; PUT PATTERN IN WRITE BUFFER
            00000000'EF40   B4     0A21  1831         CLRW    DATA_IBUFFER R0          ; PUT ZERO'S IN READ BUFFER
     E9 50    000000FF 8F   F3     0A28  1832         AOBLEQ  #255,R0,10$              ; FILL BUFFER
    0000'C2   00000000'8F   C8     0A30  1833 20$:    BISL    #PGM_INIT,CR(R2)         ; INIT MUT
    0000'C2   00000000'8F   C8     0A39  1834         BISL    #MAINT_MODE,CR(R2)       ; PUT INTO MAINTENANCE MODE
           0000'C2    5B   D0     0A42  1835         MOVL    R11,VAR(R2)              ; POINT VAR TO WRITE BUFFER
    0000'C2    00000200 8F   CE     0A47  1836         MNEGL   #512,BCR(R2)             ; SET BYTE COUNT TO 512
         56    00000000'EF   D0     0A50  1837         MOVL    DRIVE0,R6                ; GET ADDRESS OF DRIVE 0
         66    00000000'8F   D0     0A57  1838         MOVL    #WRITE,(R6)              ; ISSUE WRITE TO DRIVE
    0000'C2    00000000'8F   D0     0A5E  1839         MOVL    #SIM_OCC,DR(R2)          ; ASSERT MASSBUS OCCUPIED
                                    0A67  1840         $DS_WAITUS_S    #10              ; WAIT 100 US
                        00   DD     0A67                    PUSHL   #0
                        0A   DD     0A69                    PUSHL   #10
         00000000'9F    02   FB     0A6B                    CALLS   #2, a#DS$WAITUS
                        5A   D4     0A72  1841         CLRL    R10                      ; INIT INDEX
    0000'C2    00000000'8F   C8     0A74  1842 25$:    BISL    #SIM_SCLK,DR(R2)         ; ASSERT SCLK
    0000'C2    00000000'8F   CA     0A7D  1843         BICL    #SIM_SCLK,DR(R2)         ; DEASSERT SCLK
    0000'C2    00000000'8F   C8     0A86  1844         BISL    #MBDIB_SEL,DR(R2)        ; SELECT THE MBDIB
        0000'CA    0000'C2   B0     0A8F  1845         MOVW    DR(R2),DATA_IBUFFER(R10) ; PUT VALUE IN READ BUFFER
                        5A   D6     0A96  1846         INCL    R10                      ; SETUP R10 FOR AOBLSS
      D4 5A    000001FF 8F   F2     0A98  1847         AOBLSS  #511,R10,25$             ; DO FOR ALL 512 BYTES
                                    0AA0  1848
    0000'C2    00000000'8F   C8     0AA0  1849         BISL    #SIM_EBL,DR(R2)          ; SET END OF BLOCK
    0000'C2    00000000'8F   CA     0AA9  1850         BICL    #SIM_EBL,DR(R2)          ; CLEAR END OF BLOCK
                                    0AB2  1851
                                    0AB2  1852         ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF    00000000'EF   DE     0AB2                    MOVAL   RHSR_MSG,REG_NAME
          00000000'EF    02   9A     0ABD                    MOVZBL  #2,REG_NO
00000000'EF    00000000'EF   DE     0AC4                    MOVAL   FMT_STATUS_REG,REG_STRING
          54    00000000'8F   D0     0ACF  1853         MOVL    #DATA_XFER_DONE,R4       ; SET UP EXPECTED STATUS
          54    00000000'EF   C8     0AD6  1854         BISL    RH11TB_MSK,R4            ; LOAD EXPECTED MASK BIT
              53    0000'C2   D0     0ADD  1855         MOVL    SR(R2),R3                ; READ STATUS REGISTER
                    54    53   D1     0AE2  1856         CMPL    R3,R4                    ; IS STATUS AS EXPECTED
                          1B   13     0AE5  1857         BEQL    30$                      ; BRANCH IF STATUS IS OK
                                    0AE7  1858         $DS_ERRHARD_S    ,LUN,-          ; PRINT ERROR
                                    0AE7  1859                          ARRAY_MSG62,-
                                    0AE7  1860                          PRINT_SBE
              00000000'EF   DF     0AE7                    PUSHAL  PRINT_SBE
              00000000'EF   DF     0AED                    PUSHAL  ARRAY_MSG62
              00000000'EF   DD     0AF3                    PUSHL   LUN
                          01   DD     0AF9                    PUSHL   #$ER
                                    0AFB       ;;; TEST 9, SUBTEST 7, ERROR 1
         00000000'9F    04   FB     0AFB                    CALLS   #$$M, a#DS$ERRHARD
                                    0B02  1861 30$:    $DS_CKLOOP      20$              ; SCOPE LOOP
00000000'9F    FF2A CF   FA     0B02                    CALLG   20$, a#DS$CKLOOP
                                    0B0B  1862
                          55   D4     0B0B  1863         CLRL    R5                       ; CLEAR INDEX
                          59   D4     0B0D  1864         CLRL    R9                       ; CLEAR INDEX
         56    00000000'EF   DE     0B0F  1865         MOVAL   DATA_OBUFFER,R6          ; GET ADDRESS OF EXPECTED DATA
```

```
                57   00000000'EF   DE  0B16  1866          MOVAL   DATA_1BUFFER,R7          ; GET ADDRESS OF RECEIVED DATA
              54   00000000'EF49   90  0B1D  1867 40$:     MOVB    DATA_OBUFFER R9 ,R4      ; EXPECTED DATA
              53   00000000'EF49   90  0B25  1868          MOVB    DATA_IBUFFER R9 ,R3      ; RECEIVED DATA
                         54   53   91  0B2D  1869          CMPB    R3,R4                    ; EXPECTED = RECEIVED?
                              3B   13  0B30  1870          BEQL    50$                      ; BRANCH IF SO
                              55   D6  0B32  1871          INCL    R5                       ; ADD ONE TO ERROR COUNTER
                                   0B34  1872          $DS_BNOPER      43$                  ; PRINT ONLY 8 ERRORS IF NO OPER
                              0C   E1  0B34                      BBC     #DSA$V_OPER,-    ; BR IF NO OPERATOR
                    08 0000FE00 9F       0B36                      a#DSA$GL_FLAGS, 43$
                                   0B3C  1873          $DS_BNQUICK     45$                  ; PRINT ALL ERRORS IF NOT QUICK
                              08   E1  0B3C                      BBC     #DSA$V_QUICK,-   ; BR IF NOT QUICK
                    05 0000FE00 9F       0B3E                      a#DSA$GL_FLAGS, 45$
                         55   08   D1  0B44  1874 43$:     CMPL    #8,R5                    ; PRINTED MAX NUMBER OF ERRORS?
                              24   19  0B47  1875          BLSS    50$                      ; BRANCH IF SO
                                   0B49  1876 45$:     $DS_ERRHARD_S   ,LUN,-           ; PRINT ERROR
                                   0B49  1877                      ARRAY_MSG24,-
                                   0B49  1878                      PRINT_DTERR
                        00000000'EF   DF  0B49                      PUSHAL  PRINT_DTERR
                        00000000'EF   DF  0B4F                      PUSHAL  ARRAY_MSG24
                        00000000'EF   DD  0B55                      PUSHL   LUN
                              02   DD  0B5B                      PUSHL   #$ER
                                   0B5D        ;;; TEST 9, SUBTEST 7, ERROR 2
                   00000000'9F   04   FB  0B5D                      CALLS   #$$M, a#DS$ERRHARD
                                   0B64  1879          $DS_CKLOOP      20$                  ; SCOPE LOOP?
              00000000'9F   FEC8 CF   FA  0B64                      CALLG   20$, a#DS$CKLOOP
                A8 59   000001FF 8F   F3  0B6D  1880 50$:     AOBLEQ  #511,R9,40$          ; DO FOR ALL BYTES
                              55   D5  0B75  1881          TSTL    R5                       ; ANY ERRORS REPORTED?
                              0F   13  0B77  1882          BEQL    60$                      ; BRANCH IF NONE
                                   0B79  1883          $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                              55   DD  0B79                      PUSHL   R5
                        00000000'EF   9F  0B7B                      PUSHAB  FMT_ERRSUM
                   00000000'9F   02   FB  0B81                      CALLS   #$$N, a#DS$PRINTB
                                   0B88  1884 60$:
                                   0B88  1885 $DS_ENDSUB
                                   0B88       T9_S7_X:
00000000'9F   00000144'EF   FA  0B88                      CALLG   $S$, a#DS$ENDSUB
```

```
                                  0B93  1887 $DS_ENDTEST
                  50   01   D0    0B93                         MOVL    #1, R0          ; NORMAL EXIT
                                  0B96       TEST_009_X::
        00000000'9F   6E   FA     0B96                         CALLG   (SP), @#DS$BREAK
                             04   0B9D                         RET                     ; RETURN TO TEST SEQUENCER
                                  0B9E  1888 $DS_PAGE
                                  0B9E                .SBTTL  TEST 10:  STATUS REGISTER STATIC SA0 TEST
                             00000000                        .PSECT  TEST_010, PAGE, NOWRT
                                  0024
                                  0024  1890 $DS_BGNTEST      <DEFAULT,ALL>,,LONG
                                  0024       DATA_010:
                        00000000  0024                         .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  0028       TEST_010::
                            0000  0028                         .WORD    M<>            ; ENTRY MASK
                                  002A  1891 ;++
                                  002A  1892 ; TEST DESCRIPTION:
                                  002A  1893 ;
                                  002A  1894 ;       THIS TEST CHECKS THE FOLLOWING BITS FOR SA0 CONDITIONS:
                                  002A  1895 ;
                                  002A  1896 ;       DT_BUSY,DT_ABORT,ATTENTION,NON_XIST_DRIVE,MISSED_XFER,
                                  002A  1897 ;       DATA_XFER_LATE,PROGRAM_ERROR
                                  002A  1898 ;
                                  002A  1899 ; TEST ALGORITHM:
                                  002A  1900 ;
                                  002A  1901 ;       FORCE THE RH750 INTO A STATE IN WHICH AN EXPECTED STATUS
                                  002A  1902 ;       CONDITION SHOULD BE PRESENT. IF THE STATE CONDITION IS NOT
                                  002A  1903 ;       AS EXPECTED, THEN REPORT AN ERROR WITH THE EXPECTED STATE
                                  002A  1904 ;       THE RECEIVED STATE, AND THE LOGICAL DIFFERENCE.
                                  002A  1905 ;
                                  002A  1906 ; NOTE:
                                  002A  1907 ;
                                  002A  1908 ;       R4 WILL ALWAYS INDICATE THE EXPECTED STATE AND R3 WILL
                                  002A  1909 ;       CONTAIN THE RECEIVED STATE.
                                  002A  1910 ;--
                                  002A  1911         ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                  002A  1912                 FMT_STATUS_REG
00000000'EF   00000000'EF   DE    002A                         MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A    0035                         MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE    003C                         MOVAL   FMT_STATUS_REG,REG_STRING
        52    00000000'EF   D0    0047  1913         MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS INTO R2
        50    00000000'EF   DE    004E  1914         MOVAL   MIOBUFFER,R0            ; GET BUFFER ADDRESS
         50    50   F7 8F   78    0055  1915         ASHL    #-9,R0,R0               ; NORMALIZE FOR MAP ENTRY
        50    00000000'8F   C8    005A  1916         BISL    #VALID_BIT,R0           ; VALIDATE ENTRY
   51   52    00000000'8F   C1    0061  1917         ADDL3   #MAP_OFFSET,R2,R1       ; R1 ---> MAP 0
                  61   50   D0    0069  1918         MOVL    R0,(R1)                 ; WRITE MAP ZERO
                                  006C  1919 ;++
                                  006C  1920 ; DATA TRANSFER BUSY/DATA TRANSFER ABORT TEST
                                  006C  1921 ;--
                                  006C  1922         $DS_BGNSUB
                                  006C       T10_S1::
00000000'9F   00000150'EF   FA    006C                         CALLG   $S$, @#DS$BGNSUB
                      FF86'   30    0077  1923         BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
    0000'C2   00000000'8F   C8    007A  1924 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT RH
    0000'C2   00000000'8F   C8    0083  1925         BISL    #MAINT_MODE,CR(R2)      ; PLACE RH750 IN MAINTENANCE MODE
        56    00000000'EF   D0    008C  1926         MOVL    DRIVE0,R6               ; MOVE ADDRESS OF DRIVE 0 TO R6
        66    00000000'8F   D0    0093  1927         MOVL    #READ,(R6)              ; ISSUE READ COMMAND
    0000'C2   00000000'8F   C8    009A  1928         BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
```

```
                54   00000000'8F   D0  00A3  1929          MOVL    #DT_BUSY,R4                     ; SET EXPECTED DATA
                54   00000000'EF   C8  00AA  1930          BISL    RH11TB_MSK,R4                   ; LOAD EXPECTED MASK BIT
                   53   0000'C2   D0  00B1  1931          MOVL    SR(R2),R3                       ; READ STATUS REGISTER
                        54   53   D1  00B6  1932          CMPL    R3,R4                           ; CHECK IF OK
                             1B   13  00B9  1933          BEQL    20$                             ; BRANCH IF DT BUSY IS SET
                                  00BB  1934          $DS_ERRHARD_S   #1,LUN,-                    ; PRINT ERROR
                                  00BB  1935                          ARRAY_MSG52,-
                                  00BB  1936                          PRINT_SBE
                     00000000'EF   DF  00BB                          PUSHAL  PRINT_SBE
                     00000000'EF   DF  00C1                          PUSHAL  ARRAY_MSG52
                     00000000'EF   DD  00C7                          PUSHL   LUN
                                01   DD  00CD                          PUSHL   #1
                00000000'9F     04   FB  00CF                          CALLS   #$$M, a#DS$ERRHARD
                                  00D6  1937 20$:       $DS_CKLOOP      10$                     ; SCOPE LOOP?
              00000000'9F   A1 AF   FA  00D6                          CALLG   10$, a#DS$CKLOOP
         0000'C2   00000000'8F   C8  00DE  1938          BISL    #DT_ABORT,CR(R2)                ; SET UP TO CAUSE ABORT DATA XFER
         0000'C2   00000000'8F   C8  00E7  1939          BISL    #SIM_EBL,DR(R2)                 ; ASSERT EBL
         0000'C2   00000000'8F   CA  00F0  1940          BICL    #SIM_EBL,DR(R2)                 ; DEASSEERT EBL
                   00000000'8F   D0  00F9  1941          MOVL    #DATA_XFER_ABRT!-
                                54   00FF  1942                      DATA_XFER_DONE,R4           ; EXPECTED
                54   00000000'EF   C8  0100  1943          BISL    RH11TB_MSK,R4                   ; LOAD EXPECTED MASK BIT
                   53   0000'C2   D0  0107  1944          MOVL    SR(R2),R3                       ; READ STATUS REGISTER
                        54   53   D1  010C  1945          CMPL    R3,R4                           ; EXPECTED = RECEIVED?
                             1B   13  010F  1946          BEQL    50$                             ; BRANCH IF YES
                                  0111  1947          $DS_ERRHARD_S   #2,LUN,-                    ; PRINT ERROR
                                  0111  1948                          ARRAY_MSG51,-
                                  0111  1949                          PRINT_SBE
                     00000000'EF   DF  0111                          PUSHAL  PRINT_SBE
                     00000000'EF   DF  0117                          PUSHAL  ARRAY_MSG51
                     00000000'EF   DD  011D                          PUSHL   LUN
                                02   DD  0123                          PUSHL   #2
                00000000'9F     04   FB  0125                          CALLS   #$$M, a#DS$ERRHARD
                                  012C  1950 50$:
                                  012C  1951 70$:       $DS_CKLOOP      10$                     ; SCOPE LOOP?
              00000000'9F   FF4A CF   FA  012C                          CALLG   10$, a#DS$CKLOOP
                   0000'C2   53   D0  0135  1952 80$:     MOVL    R3,SR(R2)                       ; CLEAR STATUS REGISTER BITS
                             54   D4  013A  1953          CLRL    R4                              ; SET EXPECTED DATA
                54   00000000'EF   C8  013C  1954          BISL    RH11TB_MSK,R4                   ; LOAD EXPECTED MASK BIT
                   53   0000'C2   D0  0143  1955          MOVL    SR(R2),R3                       ; READ STATUS REGISTER
                        54   53   D1  0148  1956          CMPL    R3,R4                           ; CHECK RESULTS
                             1B   13  014B  1957          BEQL    90$                             ; BRANCH IF OK
                                  014D  1958          $DS_ERRHARD_S   #3,LUN,-                    ; PRINT ERROR
                                  014D  1959                          ARRAY_MSG4,-
                                  014D  1960                          PRINT_SBE
                     00000000'EF   DF  014D                          PUSHAL  PRINT_SBE
                     00000000'EF   DF  0153                          PUSHAL  ARRAY_MSG4
                     00000000'EF   DD  0159                          PUSHL   LUN
                                03   DD  015F                          PUSHL   #3
                00000000'9F     04   FB  0161                          CALLS   #$$M, a#DS$ERRHARD
                                  0168  1961 90$:       $DS_CKLOOP      80$                     ; SCOPE LOOP?
              00000000'9F   CA AF   FA  0168                          CALLG   80$, a#DS$CKLOOP
                                  0170  1962          $DS_ENDSUB
                                  0170        T10_S1_X:
00000000'9F   00000150'EF   FA  0170                          CALLG   $S$, a#DS$ENDSUB
```

```
                                017B  1964 ;++
                                017B  1965 ; ATTENTION TEST
                                017B  1966 ;--
                                017B  1967          $DS_BGNSUB
                                017B        T10_S2::
00000000'9F   0000015C'EF   FA  017B                    CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0186  1968 60$:     BISL    #PGM_INIT,CR(R2)        ; INIT MUT
    0000'C2   00000000'8F   C8  018F  1969          BISL    #MAINT_MODE,CR(R2)      ; PUT INTO MAINTENANCE MODE
          54  00000000'8F   D0  0198  1970          MOVL    #ATTENTION,R4           ; EXPECT ATTENTION SET
    0000'C2   00000000'8F   C8  019F  1971          BISL    #SIM_ATTN,DR(R2)        ; CAUSE ATTN TO GET SET IN STATUS REG
          53  0000'C2      D0  01A8  1972          MOVL    SR(R2),R3               ; READ STATUS REGISTER
    0000'C2   00000000'8F   CA  01AD  1973          BICL    #SIM_ATTN,DR(R2)        ; CLEAR SIMULATED ATTENTION
              54  53        D1  01B6  1974          CMPL    R3,R4                   ; CHECK IF OK
                  1B        13  01B9  1975          BEQL    70$                     ; BRANCH IF ATTENTION IS SET
                                01BB  1976          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                01BB  1977                          ARRAY_MSG46,-
                                01BB  1978                          PRINT_SBE
              00000000'EF       DF  01BB                    PUSHAL  PRINT_SBE
              00000000'EF       DF  01C1                    PUSHAL  ARRAY_MSG46
              00000000'EF       DD  01C7                    PUSHL   LUN
                  01            DD  01CD                    PUSHL   #1
        00000000'9F   04        FB  01CF                    CALLS   #$$M, a#DS$ERRHARD
                                01D6  1979 70$:     $DS_CKLOOP      60$             ; SCOPE LOOP?
      00000000'9F   AD AF       FA  01D6                    CALLG   60$, a#DS$CKLOOP
            0000'C2   53        D0  01DE  1980 80$:     MOVL    R3,SR(R2)               ; CLEAR STATUS BITS
                      54        D4  01E3  1981          CLRL    R4                      ; SET THE EXPECTED DATA
          54  00000000'EF       C8  01E5  1982          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53  0000'C2      D0  01EC  1983          MOVL    SR(R2),R3               ; READ STATUS REGISTER
                  54  53        D1  01F1  1984          CMPL    R3,R4                   ; CHECK FOR ERRORS
                      1B        13  01F4  1985          BEQL    90$                     ; BRANCH IF NO ERRORS IN STATUS REG
                                01F6  1986          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                01F6  1987                          ARRAY_MSG3,-
                                01F6  1988                          PRINT_SBE
              00000000'EF       DF  01F6                    PUSHAL  PRINT_SBE
              00000000'EF       DF  01FC                    PUSHAL  ARRAY_MSG3
              00000000'EF       DD  0202                    PUSHL   LUN
                  02            DD  0208                    PUSHL   #2
        00000000'9F   04        FB  020A                    CALLS   #$$M, a#DS$ERRHARD
                                0211  1989 90$:     $DS_CKLOOP      80$             ; SCOPE LOOP?
      00000000'9F   CA AF       FA  0211                    CALLG   80$, a#DS$CKLOOP
                                0219  1990          $DS_ENDSUB
                                0219        T10_S2_X:
00000000'9F   0000015C'EF   FA  0219                    CALLG   $S$, a#DS$ENDSUB
```

```
                                0224  1992          $DS_BGNSUB
                                0224          T10_S3::
00000000'9F   00000168'EF   FA  0224                        CALLG   $S$, a#DS$BGNSUB
                                022F  1993 ;++
                                022F  1994 ; NOTE:
                                022F  1995 ;
                                022F  1996 ; NED CAN NOT BE TESTED WHILE IN MAINT MODE
                                022F  1997 ;
                                022F  1998 ; BEGIN ACCESSING DRIVES TO DETERMINE WHICH DRIVES ARE PRESENT
                                022F  1999 ;--
                         5A     D4  022F  2000          CLRL    R10                     ; CLEAR DRIVE DETECTED FLAG
              57   00000000'EF   D0  0231  2001          MOVL    DRIVE0,R7               ; PUT DRIVE0'S ADDRESS INTO R7
                         56     D4  0238  2002          CLRL    R6                      ; CLEAR DRIVE COUNTER
         0000'C2   00000000'8F   C8  023A  2003 110$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                      5B   67   D0  0243  2004          MOVL    (R7),R11                ; TRY TO READ EXT REGISTER 0
                   53   0000'C2  D0  0246  2005          MOVL    SR(R2),R3               ; READ STATUS
                    34 53   12   E0  024B  2006          BBS     #18,R3,120$             ; IF NED IS SET BRANCH
              0000FE54'EF   01   D1  024F  2007          CMPL    #1,DSA$GL_PASSNO        ; IS THIS THE FIRST PASS?
                         0D   18  0256  2008          BGEQ    115$                    ; BRANCH IF YES
0000FE00'EF   00000400 8F   D3  0258  2009          BITL    #DSA$M_TRACE,DSA$GL_FLAGS; TRACE FLAG SET?
                         0F   13  0263  2010          BEQL    116$                    ; BRANCH IF NO
                                0265  2011 115$:    $DS_PRINTX_S    FMT_DRIVERES,R6 ; PRINT DRIVE RESPONDED MESSAGE
                         56   DD  0265                        PUSHL   R6
                   00000000'EF   9F  0267                        PUSHAB  FMT_DRIVERES
              00000000'9F   02   FB  026D                        CALLS   #$$N, a#DS$PRINTX
                   00000000'EF   D4  0274  2012 116$:    CLRL    NODRIVE                 ; CLEAR NO DRIVE FLAG
              5A   FFFFFFFF 8F   CE  027A  2013          MNEGL   #-1,R10                 ; SET DRIVE DETECTED FLAG
                         23   11  0281  2014          BRB     130$                    ; SKIP NED MESSAGE
                         5B   B5  0283  2015 120$:    TSTW    R11                     ; ZERO SHOULD COME BACK FROM NED
                         17   13  0285  2016          BEQL    125$                    ; SKIP ERROR REPORT IF NED = 0
                                0287  2017          $DS_ERRHARD_S   #1,LUN,-        ; REPORT MCP ERROR
                                0287  2018                          ARRAY_MSG14
                         00   DD  0287                        PUSHL   #0
                   00000000'EF   DF  0289                        PUSHAL  ARRAY_MSG14
                   00000000'EF   DD  028F                        PUSHL   LUN
                         01   DD  0295                        PUSHL   #1
              00000000'9F   04   FB  0297                        CALLS   #$$M, a#DS$ERRHARD
                                029E  2019 125$:    $DS_CKLOOP      110$            ; SCOPE LOOP?
         00000000'9F   99   AF   FA  029E                        CALLG   110$, a#DS$CKLOOP
                0000'C2   53   D0  02A6  2020 130$:    MOVL    R3,SR(R2)               ; CLEAR STATUS BITS THAT WERE SET
                         54   D4  02AB  2021          CLRL    R4                      ; SET THE EXPECTED DATA
                   53   0000'C2  D0  02AD  2022          MOVL    SR(R2),R3               ; READ STATUS REGISTER
                        54   53   D1  02B2  2023          CMPL    R3,R4                   ; CHECK RESULTS
                         24   13  02B5  2024          BEQL    135$                    ; SKIP IF NO ERRS
                                02B7  2025          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                02B7  2026                          ARRAY_MSG5,-
                                02B7  2027                          PRINT_SBE
                   00000000'EF   DF  02B7                        PUSHAL  PRINT_SBE
                   00000000'EF   DF  02BD                        PUSHAL  ARRAY_MSG5
                   00000000'EF   DD  02C3                        PUSHL   LUN
                         02   DD  02C9                        PUSHL   #2
              00000000'9F   04   FB  02CB                        CALLS   #$$M, a#DS$ERRHARD
                                02D2  2028          $DS_CKLOOP      110$            ; SCOPE LOOP?
00000000'9F   FF64 CF   FA  02D2                        CALLG   110$, a#DS$CKLOOP
     57   00000000'8F   C0  02DB  2029 135$:    ADDL    #DRIVE_OFFSET,R7        ; INCREMENT DRIVE ADDRESS POINTER
   FF52 56   01   07   3D  02E2  2030          ACBW    #7,#1,R6,110$           ; LOOP IF ALL DRIVES NOT TESTED
                         5A   D5  02E8  2031          TSTL    R10                     ; IF NO DRIVE RESPONDED
```

```
                        2A   12  02EA  2032          BNEQ    140$                      ; BRANCH IF A DRIVE RESPONDED
          0000FE54'EF   01   D1  02EC  2033          CMPL    #1,DSA$GL_PASSNO          ; IS THIS THE FIRST PASS?
                        0D   18  02F3  2034          BGEQ    137$                      ; BRANCH IF YES
  0000FE00'EF   00000400 8F  D3  02F5  2035          BITL    #DSA$M_TRACE,DSA$GL_FLAGS; TRACE FLAG SET?
                        0D   13  0300  2036          BEQL    138$                      ; BRANCH IF NO
                                 0302  2037 137$:    $DS_PRINTX_S    FMT_DRIVENORES    ; NO DRIVE RESPONDED
              00000000'EF   9F  0302                          PUSHAB  FMT_DRIVENORES
          00000000'9F   01   FB  0308                          CALLS   #$$N, @#DS$PRINTX
          00000000'EF   01   D0  030F  2038 138$:    MOVL    #1,NODRIVE                ; INDICATE NO DRIVE PRESENT
                                 0316  2039 140$:    $DS_ENDSUB
                                 0316       T10_S3_X:
  00000000'9F   00000168'EF   FA  0316                          CALLG   $S$, @#DS$ENDSUB
```

```
                                0321  2041 ;++
                                0321  2042 ; MISSED TRANSFER TEST
                                0321  2043 ;--
                                0321  2044         $DS_BGNSUB
                                0321        T10_S4::
    00000000'9F   00000174'EF   FA  0321                    CALLG   $S$, a#DS$BGNSUB
                                032C  2045 140$:   ERRPREP RHSR_MSG,2,-          ; PREPARE TO PRINT ERROR
                                032C  2046                 FMT_STATUS_REG
    00000000'EF   00000000'EF   DE  032C                    MOVAL   RHSR_MSG,REG_NAME
          00000000'EF    02     9A  0337                    MOVZBL  #2,REG_NO
    00000000'EF   00000000'EF   DE  033E                    MOVAL   FMT_STATUS_REG,REG_STRING
        0000'C2   00000000'8F   C8  0349  2047         BISL    #PGM_INIT,CR(R2)        ; INIT THE RH UNDER TEST
        0000'C2   00000000'8F   C8  0352  2048         BISL    #MAINT_MODE,CR(R2)      ; PUT IT INTO MAINTENANCE MODE
              56  00000000'EF   D0  035B  2049         MOVL    DRIVE0,R6               ; PUT ADDRESS OF DRIVE IN R6
              54  00000000'8F   D0  0362  2050         MOVL    #MISSED_XFER,R4         ; SET UP EXPECTED STATUS
                  00000000'8F   C8  0369  2051         BISL    #DATA_XFER_DONE!-
                                036F  2052                 DATA_XFER_ABRT,R4       ; EXPECT THESE STATUS BITS
              66  00000000'8F   D0  0370  2053         MOVL    #READ,(R6)              ; ISSUE READ COMMAND TO DRIVE
                                0377  2054         $DS_WAITUS_S    #200            ; WAIT 2 MILLISECONDS
                           00   DD  0377                    PUSHL   #0
                  000000C8 8F   DD  0379                    PUSHL   #200
                00000000 9F     02  FB  037F                    CALLS   #2, a#DS$WAITUS
              54  00000000'EF   C8  0386  2055         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                  53  0000'C2   D0  038D  2056         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                      54  53    D1  0392  2057         CMPL    R3,R4                   ; CHECK IF OK
                          1B    13  0395  2058         BEQL    150$                    ; BRANCH IF MISSED XFER SET
                                0397  2059         $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                0397  2060                 ARRAY_MSG41,-
                                0397  2061                 PRINT_SBE
                00000000'EF     DF  0397                    PUSHAL  PRINT_SBE
                00000000'EF     DF  039D                    PUSHAL  ARRAY_MSG41
                00000000'EF     DD  03A3                    PUSHL   LUN
                           01   DD  03A9                    PUSHL   #1
            00000000'9F     04  FB  03AB                    CALLS   #$$M, a#DS$ERRHARD
                                03B2  2062 150$:   $DS_CKLOOP      140$             ; SCOPE LOOP?
    00000000'9F   FF76 CF       FA  03B2                    CALLG   140$, a#DS$CKLOOP
          0000'C2   53          D0  03BB  2063 160$:   MOVL    R3,SR(R2)               ; CLEAR IN STATUS REGISTER BITS
          53  0000'C2          D0  03C0  2064         MOVL    SR(R2),R3               ; CHECK FOR SA1 BITS IN STATUS REG
                    54          D4  03C5  2065         CLRL    R4                      ; CLEAR EXPECTED RESULTS REG
        54  00000000'EF          C8  03C7  2066         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              54    53          D1  03CE  2067         CMPL    R3,R4                   ; CHECK RESULTS
                    1B          13  03D1  2068         BEQL    170$                    ; BRANCH IF STATE IS OK
                                03D3  2069         $DS_ERRHARD_S   #2,LUN,-         ; PRINT ERROR
                                03D3  2070                 ARRAY_MSG4,-
                                03D3  2071                 PRINT_SBE
                00000000'EF     DF  03D3                    PUSHAL  PRINT_SBE
                00000000'EF     DF  03D9                    PUSHAL  ARRAY_MSG4
                00000000'EF     DD  03DF                    PUSHL   LUN
                           02   DD  03E5                    PUSHL   #2
            00000000'9F     04  FB  03E7                    CALLS   #$$M, a#DS$ERRHARD
                                03EE  2072 170$:   $DS_CKLOOP      160$             ; SCOPE LOOP?
      00000000'9F     CA AF     FA  03EE                    CALLG   160$, a#DS$CKLOOP
                                03F6  2073         $DS_ENDSUB
                                03F6        T10_S4_X:
00000000'9F   00000174'EF       FA  03F6                    CALLG   $S$, a#DS$ENDSUB
```

```
                                0401  2075 ;++
                                0401  2076 ; DATA LATE WHEN SILO IS EMPTY AND SCLK RECEIVED
                                0401  2077 ;--
                                0401  2078          $DS_BGNSUB
                                0401        T10_S5::
00000000'9F   00000180'EF   FA  0401                  CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  040C  2079 180$:      BISL    #PGM_INIT,CR(R2)          ; INIT
    0000'C2   00000000'8F   C8  0415  2080            BISL    #MAINT_MODE,CR(R2)        ; PUT INTO MAINTENANCE MODE
    50   52   00000000'8F   C1  041E  2081            ADDL3   #MAP_OFFSET,R2,R0         ; R0 NOW POINTS TO MAP REG 0
         60   00000000'8F   D0  0426  2082            MOVL    #VALID_BIT,(R0)           ; VALIDATE MAP REG 0
                              042D  2083                                              ; THIS IS REQUIRED TO PREVENT STATUS
                              042D  2084                                              ; INDICATING INVALID MAP WHEN THE
                              042D  2085                                              ; PHYSICAL ADDRESS IS ASSEMBLED
    0000'C2   FFFFFFD7 8F   D0  042D  2086            MOVL    #-41,BCR(R2)              ; WRITE BCR
                 0000'C2    D4  0436  2087            CLRL    VAR(R2)                   ; CLEAR VIRTUAL ADDRESS REGISTER
         54   00000000'8F   D0  043A  2088            MOVL    #DATA_XFER_LATE,R4        ; SET UP EXPECTED STATUS
              00000000'8F   C8  0441  2089            BISL    #DATA_XFER_ABRT!-
                           54   0447  2090                    DATA_XFER_DONE,R4         ; EXPECT THESE BITS
         66   00000000'8F   D0  0448  2091            MOVL    #WRITE,(R6)               ; ISSUE WRITE COMMAND TO PSEUDO DRIVE
    0000'C2   00000000'8F   D0  044F  2092            MOVL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
                              0458  2093            $DS_WAITUS_S    #10               ; WAIT FOR SILO TO FILL
                           00   DD  0458                    PUSHL   #0
                           0A   DD  045A                    PUSHL   #10
         00000000'9F       02   FB  045C                    CALLS   #2, a#DS$WAITUS
    0000'C2   00000000'8F   C8  0463  2094            BISL    #BLKSND_COMD,DR(R2)       ; BLOCK CMI COMMANDS
                           50   D4  046C  2095            CLRL    R0                        ; CLEAR SCLK COUNTER
    0000'C2   00000000'8F   C8  046E  2096 2000$:     BISL    #SIM_SCLK,DR(R2)          ; ASSERT SCLK
    0000'C2   00000000'8F   CA  0477  2097            BICL    #SIM_SCLK,DR(R2)          ; NEGATE SCLK
               EA 50        15  F3  0480  2098            AOBLEQ  #21,R0,2000$              ; DO FOR 22 CLOCKS
    0000'C2   00000000'8F   C8  0484  2099            BISL    #SIM_EBL,DR(R2)           ; SET SIMULATED END OF BLOCK
    0000'C2   00000000'8F   CA  048D  2100            BICL    #SIM_EBL,DR(R2)           ; CLEAR SIMULATED END OF BLOCK
         54   00000000'EF   C8  0496  2101            BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  049D  2102            MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                   54   53   D1  04A2  2103            CMPL    R3,R4                     ; CHECK IF OK
                        1B   13  04A5  2104            BEQL    190$                      ; BRANCH IF DATA LATE BIT IS SET
                              04A7  2105            $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                              04A7  2106                            ARRAY_MSG50,-
                              04A7  2107                            PRINT_SBE
                   00000000'EF   DF  04A7                    PUSHAL  PRINT_SBE
                   00000000'EF   DF  04AD                    PUSHAL  ARRAY_MSG50
                   00000000'EF   DD  04B3                    PUSHL   LUN
                              01   DD  04B9                    PUSHL   #1
         00000000'9F       04   FB  04BB                    CALLS   #$$M, a#DS$ERRHARD
                              04C2  2108 190$:      $DS_CKLOOP      180$
00000000'9F   FF46 CF   FA  04C2                    CALLG   180$, a#DS$CKLOOP
         0000'C2   53   D0  04CB  2109 200$:      MOVL    R3,SR(R2)                 ; RESET ONE BITS
         53   0000'C2   D0  04D0  2110            MOVL    SR(R2),R3                 ; READ STATUS BIS
                        54   D4  04D5  2111            CLRL    R4                        ; CLEAR EXPECTED RESULTS REG
    54   00000000'EF   C8  04D7  2112            BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53   D1  04DE  2113            CMPL    R3,R4                     ; CHECK RESULTS
                   1B   13  04E1  2114            BEQL    210$                      ; BRANCH IS STATUS IS OK
                              04E3  2115            $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                              04E3  2116                            ARRAY_MSG6,-
                              04E3  2117                            PRINT_SBE
                   00000000'EF   DF  04E3                    PUSHAL  PRINT_SBE
                   00000000'EF   DF  04E9                    PUSHAL  ARRAY_MSG6
                   00000000'EF   DD  04EF                    PUSHL   LUN
```

```
                        02   DD  04F5                          PUSHL   #2
          00000000'9F   04   FB  04F7                          CALLS   #$$M, a#DS$ERRHARD
                                 04FE  2118 210$:   $DS_CKLOOP      200$
       00000000'9F   CA AF   FA  04FE                          CALLG   200$, a#DS$CKLOOP
                                 0506  2119         $DS_ENDSUB
                                 0506       T10_S5_X:
  00000000'9F   00000180'EF   FA  0506                          CALLG   $S$, a#DS$ENDSUB
```

```
                                  0511  2121 ;++
                                  0511  2122 ; PROGRAM ERROR--TRANSFER IN PROGRESS AND TRANSFER COMMAND ISSUED
                                  0511  2123 ;--
                                  0511  2124          $DS_BGNSUB
                                  0511        T10_S6::
00000000'9F   0000018C'EF   FA    0511                        CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8    051C  2125 220$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH UNDER TEST
    0000'C2   00000000'8F   C8    0525  2126         BISL    #MAINT_MODE,CR(R2)        ; PUT IT INTO MAINTENANCE MODE
                  0000'C2   D4    052E  2127         CLRL    VAR(R2)                   ; CLEAR VIRTUAL ADDRESS REGISTER
         66   00000000'8F   D0    0532  2128         MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
    0000'C2   00000000'8F   C8    0539  2129         BISL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
         66   00000000'8F   D0    0542  2130         MOVL    #READ,(R6)                ; ISSUE SECOND READ
             53   0000'C2   D0    0549  2131         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
         54   00000000'8F   D0    054E  2132         MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; SET UP EXPECTED STATUS
         54   00000000'EF   C8    0555  2133         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                  54   53   D1    055C  2134         CMPL    R3,R4                     ; CHECK IF OK
                       1B   13    055F  2135         BEQL    230$                      ; BRANCH IF PROGRAM ERROR IS SET
                                  0561  2136         $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                  0561  2137                         ARRAY_MSG36,-
                                  0561  2138                         PRINT_SBE
             00000000'EF   DF    0561                        PUSHAL  PRINT_SBE
             00000000'EF   DF    0567                        PUSHAL  ARRAY_MSG36
             00000000'EF   DD    056D                        PUSHL   LUN
                       01   DD    0573                        PUSHL   #1
         00000000'9F   04   FB    0575                        CALLS   #$$M, a#DS$ERRHARD
                                  057C  2139 230$:   $DS_CKLOOP      220$              ; SCOPE LOOP?
      00000000'9F   9D AF   FA    057C                        CALLG   220$, a#DS$CKLOOP
             0000'C2   53   D0    0584  2140         MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
             53   0000'C2   D0    0589  2141         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
         54   00000000'8F   D0    058E  2142         MOVL    #DT_BUSY,R4               ; SET UP EXPECTED STATUS
         54   00000000'EF   C8    0595  2143         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                  54   53   D1    059C  2144         CMPL    R3,R4                     ; IS PGE CLEAR AND DT BUSY SET
                       1B   13    059F  2145         BEQL    250$                      ; BRANCH IF STATUS IS OK
                                  05A1  2146         $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                  05A1  2147                         ARRAY_MSG6,-
                                  05A1  2148                         PRINT_SBE
             00000000'EF   DF    05A1                        PUSHAL  PRINT_SBE
             00000000'EF   DF    05A7                        PUSHAL  ARRAY_MSG6
             00000000'EF   DD    05AD                        PUSHL   LUN
                       02   DD    05B3                        PUSHL   #2
         00000000'9F   04   FB    05B5                        CALLS   #$$M, a#DS$ERRHARD
                                  05BC  2149 250$:   $DS_CKLOOP      220$              ; SCOPE LOOP?
    00000000'9F   FF5C CF   FA    05BC                        CALLG   220$, a#DS$CKLOOP
                                  05C5  2150         $DS_ENDSUB
                                  05C5        T10_S6_X:
00000000'9F   0000018C'EF   FA    05C5                        CALLG   $S$, a#DS$ENDSUB
```

```
                                05D0  2152 ;++
                                05D0  2153 ; PROGRAM ERROR--TRANSFER IN PROGRESS AND ATTEMPT TO WRITE BCR
                                05D0  2154 ;--
                                05D0  2155          $DS_BGNSUB
                                05D0        T10_S7::
00000000'9F   00000198'EF   FA  05D0                CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  05DB  2156 220$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH UNDER TEST
    0000'C2   00000000'8F   C8  05E4  2157          BISL    #MAINT_MODE,CR(R2)        ; PUT RH INTO MAINTENANCE MODE
              0000'C2       D4  05ED  2158          CLRL    VAR(R2)                   ; CLEAR VIRT ADR REG
         66   00000000'8F   D0  05F1  2159          MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
    0000'C2   00000000'8F   C8  05F8  2160          BISL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
              0000'C2       D4  0601  2161          CLRL    BCR(R2)                   ; ATTEMPT TO WRITE BYTE COUNTER
           53 0000'C2       D0  0605  2162          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
         54   00000000'8F   D0  060A  2163          MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; SET UP EXPECTED STATUS
         54   00000000'EF   C8  0611  2164          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53       D1  0618  2165          CMPL    R3,R4                     ; CHECK IF OK
                   1B       13  061B  2166          BEQL    230$                      ; BRANCH IF PROGRAM ERROR IS SET
                                061D  2167          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                061D  2168                          ARRAY_MSG5,-
                                061D  2169                          PRINT_SBE
              00000000'EF       DF  061D                    PUSHAL  PRINT_SBE
              00000000'EF       DF  0623                    PUSHAL  ARRAY_MSG5
              00000000'EF       DD  0629                    PUSHL   LUN
                   01           DD  062F                    PUSHL   #1
         00000000'9F   04       FB  0631                    CALLS   #$$M, a#DS$ERRHARD
                                0638  2170 230$:    $DS_CKLOOP      220$              ; SCOPE LOOP?
      00000000'9F   A0 AF       FA  0638                    CALLG   220$, a#DS$CKLOOP
           0000'C2   53         D0  0640  2171          MOVL    R3,SR(R2)             ; WRITE STATUS BITS TO CLEAR
           53   0000'C2         D0  0645  2172          MOVL    SR(R2),R3             ; READ STATUS REGISTER
      54   00000000'8F         D0  064A  2173          MOVL    #DT_BUSY,R4           ; SET UP EXPECTED STATUS
      54   00000000'EF         C8  0651  2174          BISL    RH11TB_MSK,R4         ; LOAD EXPECTED MASK BIT
                54   53         D1  0658  2175          CMPL    R3,R4                 ; IS PGE CLEAR AND DT BUSY SET
                     1B         13  065B  2176          BEQL    250$                  ; BRANCH IF STATUS IS OK
                                065D  2177          $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                065D  2178                          ARRAY_MSG5,-
                                065D  2179                          PRINT_SBE
              00000000'EF       DF  065D                    PUSHAL  PRINT_SBE
              00000000'EF       DF  0663                    PUSHAL  ARRAY_MSG5
              00000000'EF       DD  0669                    PUSHL   LUN
                   02           DD  066F                    PUSHL   #2
         00000000'9F   04       FB  0671                    CALLS   #$$M, a#DS$ERRHARD
                                0678  2180 250$:    $DS_CKLOOP      220$              ; SCOPE LOOP?
   00000000'9F   FF5F CF       FA  0678                    CALLG   220$, a#DS$CKLOOP
                                0681  2181          $DS_ENDSUB
                                0681        T10_S7_X:
00000000'9F   00000198'EF   FA  0681                CALLG   $S$, a#DS$ENDSUB
```

```
                                    068C  2183 ;++
                                    068C  2184 ; PROGRAM ERROR--TRANSFER IN PROGRESS AND ATTEMPT
                                    068C  2185 ; TO WRITE VIRTUAL ADDRESS REGISTER
                                    068C  2186 ;--
                                    068C  2187          $DS_BGNSUB
                                    068C        T10_S8::
00000000'9F   000001A4'EF   FA      068C                        CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8      0697  2188 220$:    BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8      06A0  2189          BISL    #MAINT_MODE,CR(R2)        ; PUT IT INTO MAINTENANCE MODE
              0000'C2       D4      06A9  2190          CLRL    VAR(R2)                   ; CLEAR VIRT ADR REG
          66  00000000'8F   D0      06AD  2191          MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
    0000'C2   00000000'8F   C8      06B4  2192          BISL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
              0000'C2       D4      06BD  2193          CLRL    VAR(R2)                   ; ATTEMPT TO CLEAR VAR
          53  0000'C2       D0      06C1  2194          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
          54   00000000'8F  D0      06C6  2195          MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; SET UP EXPECTED STATUS
          54   00000000'EF  C8      06CD  2196          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
               54   53      D1      06D4  2197          CMPL    R3,R4                     ; CHECK IF OK
                    1B      13      06D7  2198          BEQL    230$                      ; BRANCH IF PROGRAM ERROR
                                    06D9  2199          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                    06D9  2200                          ARRAY_MSG5,-
                                    06D9  2201                          PRINT_SBE
              00000000'EF   DF      06D9                        PUSHAL  PRINT_SBE
              00000000'EF   DF      06DF                        PUSHAL  ARRAY_MSG5
              00000000'EF   DD      06E5                        PUSHL   LUN
                    01      DD      06EB                        PUSHL   #1
         00000000'9F   04     FB      06ED                        CALLS   #$$M, a#DS$ERRHARD
                                    06F4  2202 230$:    $DS_CKLOOP      220$              ; SCOPE LOOP?
   00000000'9F   A0 AF      FA      06F4                        CALLG   220$, a#DS$CKLOOP
         0000'C2   53       D0      06FC  2203          MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
         53   0000'C2       D0      0701  2204          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
      54   00000000'8F      D0      0706  2205          MOVL    #DT_BUSY,R4               ; SET UP EXPECTED STATUS
      54   00000000'EF      C8      070D  2206          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
           54   53          D1      0714  2207          CMPL    R3,R4                     ; IS PGE CLEAR AND DT BUSY SET
                1B          13      0717  2208          BEQL    250$                      ; BRANCH IF STATUS IS OK
                                    0719  2209          $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                    0719  2210                          ARRAY_MSG5,-
                                    0719  2211                          PRINT_SBE
              00000000'EF   DF      0719                        PUSHAL  PRINT_SBE
              00000000'EF   DF      071F                        PUSHAL  ARRAY_MSG5
              00000000'EF   DD      0725                        PUSHL   LUN
                    02      DD      072B                        PUSHL   #2
         00000000'9F   04     FB      072D                        CALLS   #$$M, a#DS$ERRHARD
                                    0734  2212 250$:    $DS_CKLOOP      220$              ; SCOPE LOOP
   00000000'9F   FF5F CF    FA      0734                        CALLG   220$, a#DS$CKLOOP
                                    073D  2213          $DS_ENDSUB
                                    073D        T10_S8_X:
00000000'9F   000001A4'EF   FA      073D                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0748  2215 ;++
                                0748  2216 ; PROGRAM ERROR--ATTEMPT TO WRITE A MAP REGISTER WHEN TRANSFER IS IN PROGRESS
                                0748  2217 ;--
                                0748  2218          $DS_BGNSUB
                                0748        T10_S9::
00000000'9F    000001B0'EF   FA  0748                CALLG   $S$, a#DS$BGNSUB
    0000'C2    00000000'8F   C8  0753  2219 220$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH
    0000'C2    00000000'8F   C8  075C  2220          BISL    #MAINT_MODE,CR(R2)      ; PUT IT INTO MAINTENANCE MODE
               0000'C2       D4  0765  2221          CLRL    VAR(R2)                 ; CLEAR VIRT ADR REG
          66   00000000'8F   D0  0769  2222          MOVL    #READ,(R6)              ; ISSUE READ COMMAND TO DRIVE
    0000'C2    00000000'8F   C8  0770  2223          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
               0800 C2       D4  0779  2224          CLRL     X800(R2)               ; ATTEMPT TO CLEAR MAP REGISTER 0
             53   0000'C2    D0  077D  2225          MOVL    SR(R2),R3               ; READ STATUS REGISTER
          54   00000000'8F   D0  0782  2226          MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; SET UP EXPECTED STATUS
          54   00000000'EF   C8  0789  2227          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
               54   53       D1  0790  2228          CMPL    R3,R4                   ; CHECK IF OK
                    1B       13  0793  2229          BEQL    230$                    ; BRANCH IF PROGRAM ERROR SET
                                0795  2230          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                0795  2231                          ARRAY_MSG60,-
                                0795  2232                          PRINT_SBE
               00000000'EF      DF  0795                        PUSHAL  PRINT_SBE
               00000000'EF      DF  079B                        PUSHAL  ARRAY_MSG60
               00000000'EF      DD  07A1                        PUSHL   LUN
                    01          DD  07A7                        PUSHL   #1
          00000000'9F   04      FB  07A9                        CALLS   #$$M, a#DS$ERRHARD
                                07B0  2233 230$:    $DS_CKLOOP      220$            ; SCOPE LOOP?
    00000000'9F   A0 AF         FA  07B0                        CALLG   220$, a#DS$CKLOOP
          0000'C2   53          D0  07B8  2234          MOVL    R3,SR(R2)               ; WRITE STATUS BITS TO CLEAR
          53   0000'C2          D0  07BD  2235          MOVL    SR(R2),R3               ; READ STATUS REGISTER
       54   00000000'8F         D0  07C2  2236          MOVL    #DT_BUSY,R4             ; SET UP EXPECTED STATUS
       54   00000000'EF         C8  07C9  2237          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
               54   53           D1  07D0  2238          CMPL    R3,R4                   ; IS PGE CLEAR AND DT BUSY SET
                    1B           13  07D3  2239          BEQL    250$                    ; BRANCH IF STATUS IS OK
                                07D5  2240          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                07D5  2241                          ARRAY_MSG5,-
                                07D5  2242                          PRINT_SBE
               00000000'EF      DF  07D5                        PUSHAL  PRINT_SBE
               00000000'EF      DF  07DB                        PUSHAL  ARRAY_MSG5
               00000000'EF      DD  07E1                        PUSHL   LUN
                    02          DD  07E7                        PUSHL   #2
          00000000'9F   04      FB  07E9                        CALLS   #$$M, a#DS$ERRHARD
                                07F0  2243 250$:    $DS_CKLOOP      220$            ; SCOPE LOOP?
    00000000'9F   FF5F CF       FA  07F0                        CALLG   220$, a#DS$CKLOOP
                                07F9  2244          $DS_ENDSUB
                                07F9        T10_S9_X:
00000000'9F    000001B0'EF   FA  07F9                CALLG   $S$, a#DS$ENDSUB
```

```
                                  0804  2246 ;++
                                  0804  2247 ; DATA LATE-- OVERFLOW
                                  0804  2248 ;--
                                  0804  2249          $DS_BGNSUB
                                  0804        T10_S10::
00000000'9F   000001BC'EF   FA    0804                        CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8    080F  2250 260$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8    0818  2251         BISL    #MAINT_MODE,CR(R2)        ; PUT IT INTO MAINTENANCE MODE
              0000'C2   32   CE    0821  2252         MNEGL   #50,BCR(R2)               ; SET UP DUMMY BYTE COUNT
    0000'C2   00000000'8F   C8    0826  2253         BISL    #BLKSND_COMD,DR(R2)       ; SET UP TO CAUSE OVERFLOW DATA LATE
                        50   D4    082F  2254         CLRL    R0                        ; CLEAR SCLK COUNTER
                   0000'C2   D4    0831  2255         CLRL    VAR(R2)                   ; CAUSE SELECTION OF MAP0
         66   00000000'8F   D0    0835  2256         MOVL    #READ,(R6)                ; ISSUE READ COMMAND
    0000'C2   00000000'8F   C8    083C  2257         BISL    #SIM_OCC,DR(R2)           ; PREVENT MISSED XFER
    0000'C2   00000000'8F   C8    0845  2258 270$:   BISL    #SIM_SCLK,DR(R2)          ; ASSERT SCLK
    0000'C2   00000000'8F   CA    084E  2259         BICL    #SIM_SCLK,DR(R2)          ; DEASSERT SCLK
               EA 50   14   F3    0857  2260         AOBLEQ  #20,R0,270$               ; ISSUE 21 SCLK'S
    0000'C2   00000000'8F   C8    085B  2261         BISL    #SIM_EBL,DR(R2)           ; ASSERT EBL
    0000'C2   00000000'8F   CA    0864  2262         BICL    #SIM_EBL,DR(R2)           ; CLEAR EBL
         54   00000000'8F   D0    086D  2263         MOVL    #DATA_XFER_LATE,R4        ; SET UP EXPECTED STATUS
         54   00000000'8F   C8    0874  2264         BISL    #DATA_XFER_DONE!DATA_XFER_ABRT,R4  ; MERGE THESE BITS
         54   00000000'EF   C8    087B  2265         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0    0882  2266         MOVL    SR(R2),R3                 ; READ STATUS
                   54   53   D1    0887  2267         CMPL    R3,R4                     ; CHECK IF OK
                        1B   13    088A  2268         BEQL    280$                      ; BRANCH IF DATA LATE SET
                                  088C  2269         $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                  088C  2270                         ARRAY_MSG6,-
                                  088C  2271                         PRINT_SBE
              00000000'EF   DF    088C                        PUSHAL  PRINT_SBE
              00000000'EF   DF    0892                        PUSHAL  ARRAY_MSG6
              00000000'EF   DD    0898                        PUSHL   LUN
                        01   DD    089E                        PUSHL   #1
         00000000'9F   04   FB    08A0                        CALLS   #$$M, a#DS$ERRHARD
                                  08A7  2272 280$:   $DS_CKLOOP      260$              ; SCOPE LOOP?
    00000000'9F   FF64 CF   FA    08A7                        CALLG   260$, a#DS$CKLOOP
                                  08B0  2273         $DS_ENDSUB
                                  08B0        T10_S10_X:
00000000'9F   000001BC'EF   FA    08B0                        CALLG   $S$, a#DS$ENDSUB
```

```
                                08BB  2275 ;++
                                08BB  2276 ; PROGRAM ERROR--DT_BUSY AND ATTEMPT TO PUT INTO MAINTENANCE MODE
                                08BB  2277 ;--
                                08BB  2278          $DS_BGNSUB
                                08BB                T10_S11::
00000000'9F   000001C8'EF   FA  08BB                        CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  08C6  2279 10$:     BISL    #PGM_INIT,CR(R2)          ; INITIALIZE THE RH
    0000'C2   00000000'8F   C8  08CF  2280          BISL    #MAINT_MODE,CR(R2)        ; PUT IN MAINTENANCE MODE
              0000'C2       D4  08D8  2281          CLRL    VAR(R2)                   ; CLEAR VIRTUAL ADDRESS REGISTER
          66  00000000'8F   D0  08DC  2282          MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
    0000'C2   00000000'8F   C8  08E3  2283          BISL    #SIM_OCC,DR(R2)           ; SET OCCUPIED--BLOCKS MISSED XFER
          54  00000000'8F   D0  08EC  2284          MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; SET UP EXPECTED STATUS
    0000'C2   00000000'8F   CA  08F3  2285          BICL    #MAINT_MODE,CR(R2)        ; CLEAR MAINTENANCE MODE
    0000'C2   00000000'8F   C8  08FC  2286          BISL    #MAINT_MODE,CR(R2)        ; PERFORM ILLEGAL CONTROL FUNCTION
                                0905  2287                                            ; WHILE DT_BUSY IS SET
             53   0000'C2   D0  0905  2288          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
    0000'C2   00000000'8F   C8  090A  2289          BISL    #PGM_INIT,CR(R2)          ; INIT--CLEAR POTENTIAL NED
          54  00000000'EF   C8  0913  2290          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53       D1  091A  2291          CMPL    R3,R4                     ; CHECK IF OK
                   1B       13  091D  2292          BEQL    20$                       ; BRANCH IF PROGRAM ERROR IS SET
                                091F  2293          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                091F  2294                          ARRAY_MSG5,-
                                091F  2295                          PRINT_SBE
              00000000'EF       DF  091F                        PUSHAL  PRINT_SBE
              00000000'EF       DF  0925                        PUSHAL  ARRAY_MSG5
              00000000'EF       DD  092B                        PUSHL   LUN
                   01           DD  0931                        PUSHL   #1
       00000000'9F   04       FB  0933                        CALLS   #$$M, a#DS$ERRHARD
                                093A  2296 20$:     $DS_CKLOOP      10$               ; SCOPE LOOP?
       00000000'9F   89 AF    FA  093A                        CALLG   10$, a#DS$CKLOOP
                                0942  2297          $DS_ENDSUB
                                0942        T10_S11_X:
00000000'9F   000001C8'EF   FA  0942                        CALLG   $S$, a#DS$ENDSUB
```

```
                                094D  2299 ;++
                                094D  2300 ; DT_BUSY ONE-SHOT TOO SLOW TEST
                                094D  2301 ;--
                                094D  2302          $DS_BGNSUB
                                094D        T10_S12::
00000000'9F   000001D4'EF   FA  094D                    CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0958  2303 10$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE THE RH
    0000'C2   00000000'8F   C8  0961  2304          BISL    #MAINT_MODE,CR(R2)      ; PUT IN MAINTENANCE MODE
         56   00000000'EF   D0  096A  2305          MOVL    DRIVE0,R6               ; PUT ADDRESS OF DRIVE IN R6
         54   00000000'8F   D0  0971  2306          MOVL    #MISSED_XFER,R4         ; SET UP EXPECTED STATUS
              00000000'8F   C8  0978  2307          BISL    #DATA_XFER_DONE!-       ; EXPECT THEST STATUS BITS
                         54       097E  2308                  DATA_XFER_ABRT,R4       ; ...
                                  097F  2309          $DS_SETIPL_S    #31             ; BLOCK OUT ALL INTERRUPTS      ;DPM002
                         1F   DD  097F                        PUSHL   #31
         00000000'9F    01   FB  0981                        CALLS   #1, a#DS$SETIPL
         56   00000000'8F   D0  0988  2310          MOVL    #READ,(R6)              ; ISSUE READ COMMAND TO DRIVE
                                  098F  2311          $DS_WAITUS_S    #100            ; WAIT 1000 MICRO SECONDS       ;DPM002
                         00   DD  098F                        PUSHL   #0
              00000064   8F   DD  0991                        PUSHL   #100
         00000000'9F    02   FB  0997                        CALLS   #2, a#DS$WAITUS
    0000'C2   00000000'8F   D0  099E  2312          MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                  09A7  2313          $DS_SETIPL_S    #0              ; RESET IPL                     ;DPM002
                         00   DD  09A7                        PUSHL   #0
         00000000'9F    01   FB  09A9                        CALLS   #1, a#DS$SETIPL
         54   00000000'EF   C8  09B0  2314          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  09B7  2315          MOVL    SR(R2),R3               ; READ STATUS REGISTER
                   54   53   D1  09BC  2316          CMPL    R3,R4                   ; CHECK IF OK
                        1B   13  09BF  2317          BEQL    20$                     ; BRANCH IF OK
                                  09C1  2318          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR DT_BUSY ONE-SHOT TOO SLOW!
                                  09C1  2319                          COMP_MSG1,-     ; MESSAGE ADDRESS
                                  09C1  2320                          PRINT_SBE       ; PRINT LINK
                   00000000'EF   DF  09C1                        PUSHAL  PRINT_SBE
                   00000000'EF   DF  09C7                        PUSHAL  COMP_MSG1
                   00000000'EF   DD  09CD                        PUSHL   LUN
                                 01  DD  09D3                        PUSHL   #1
              00000000'9F        04  FB  09D5                        CALLS   #$$M, a#DS$ERRHARD
                                  09DC  2321 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF78 CF       FA  09DC                        CALLG   10$, a#DS$CKLOOP
                                  09E5  2322          $DS_ENDSUB
                                  09E5        T10_S12_X:
00000000'9F   000001D4'EF   FA  09E5                    CALLG   $S$, a#DS$ENDSUB
```

```
                                  09F0  2324 ;++
                                  09F0  2325 ; DT_BUSY ONE-SHOT TOO FAST TEST
                                  09F0  2326 ;--
                                  09F0  2327         $DS_BGNSUB
                                  09F0       T10_S13::
00000000'9F   000001E0'EF   FA    09F0                       CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8    09FB  2328 10$:    BISL    #PGM_INIT,CR(R2)          ; INITIALIZE THE RH
    0000'C2   00000000'8F   C8    0A04  2329         BISL    #MAINT_MODE,CR(R2)        ; PUT IN MAINTENANCE MODE
         54   00000000'8F   D0    0A0D  2330         MOVL    #DT_BUSY,R4               ; SET UP EXPECTED STATUS
                                  0A14  2331         $DS_SETIPL_S    #31               ; BLOCK OUT ALL INTERRUPTS      ;DPM002
                        1F   DD   0A14                       PUSHL   #31
         00000000'9F    01   FB   0A16                       CALLS   #1, a#DS$SETIPL
         66   00000000'8F   D0    0A1D  2332         MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
                                  0A24  2333         $DS_WAITUS_S    #30               ; WAIT 300 MICRO SECONDS        ;DPM002
                        00   DD   0A24                       PUSHL   #0
                        1E   DD   0A26                       PUSHL   #30
         00000000'9F    02   FB   0A28                       CALLS   #2, a#DS$WAITUS
    0000'C2   00000000'8F   D0    0A2F  2334         MOVL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
                                  0A38  2335         $DS_SETIPL_S    #0                ; RESET IPL                     ;DPM002
                        00   DD   0A38                       PUSHL   #0
         00000000'9F    01   FB   0A3A                       CALLS   #1, a#DS$SETIPL
         54   00000000'EF   C8    0A41  2336         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0    0A48  2337         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                   54   53   D1    0A4D  2338         CMPL    R3,R4                     ; CHECK IF OK
                        1B   13   0A50  2339         BEQL    20$                       ; BRANCH IF OK
                                  0A52  2340         $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR DT_BUSY ONE-SHOT TOO FAST!
                                  0A52  2341                         COMP_MSG2,-       ; MESSAGE ADDRESS
                                  0A52  2342                         PRINT_SBE         ; PRINT LINK
              00000000'EF   DF    0A52                       PUSHAL  PRINT_SBE
              00000000'EF   DF    0A58                       PUSHAL  COMP_MSG2
              00000000'EF   DD    0A5E                       PUSHL   LUN
                        01   DD   0A64                       PUSHL   #1
         00000000'9F    04   FB   0A66                       CALLS   #$$M, a#DS$ERRHARD
                                  0A6D  2343 20$:    $DS_CKLOOP      10$               ; SCOPE LOOP?
      00000000'9F   8B AF   FA    0A6D                       CALLG   10$, a#DS$CKLOOP
                                  0A75  2344         $DS_ENDSUB
                                  0A75       T10_S13_X:
00000000'9F   000001E0'EF   FA    0A75                       CALLG   $S$, a#DS$ENDSUB
```

```
                                  0A80  2346 $DS_ENDTEST
                  50    01    D0  0A80                  MOVL    #1, R0          ; NORMAL EXIT
                                  0A83        TEST_010_X::
        00000000'9F     6E    FA  0A83                  CALLG   (SP), @#DS$BREAK
                              04  0A8A                  RET                     ; RETURN TO TEST SEQUENCER
                                  0A8B  2347 $DS_PAGE
                                  0A8B              .SBTTL  TEST 11:  MAPPING FUNCTION TEST
                              00000000                  .PSECT  TEST_011, PAGE, NOWRT
                                  0018
                                  0018  2349 $DS_BGNTEST    <DEFAULT,ALL>,,LONG
                                  0018        DATA_011:
                        00000000  0018                  .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  001C        TEST_011::
                            0000  001C                  .WORD    M<>            ; ENTRY MASK
                                  001E  2350 ;++
                                  001E  2351 ; TEST DESCRIPTION:
                                  001E  2352 ;
                                  001E  2353 ;       THIS TEST VERIFIES THAT THE MAP REGISTER SELECTION FUNCTION
                                  001E  2354 ;       IS CORRECTLY SELECTING THE MAP REGISTER POINTED AT BY THE
                                  001E  2355 ;       VIRTUAL ADDRESS REGISTER.
                                  001E  2356 ;       IT ALSO TESTS THE PARITY ASSOCIATED WITH THE MAP REGISTERS.
                                  001E  2357 ;
                                  001E  2358 ; REGISTER CONVENTIONS:
                                  001E  2359 ;
                                  001E  2360 ;       R2 = RH750 ADDRESS
                                  001E  2361 ;       R3 = MAP REGISTER ACTUALLY SELECTED
                                  001E  2362 ;       R4 = MAP REGISTER THAT SHOULD HAVE BEEN SELECTED
                                  001E  2363 ;       R6 = VALUE THAT SHOULD HAVE BEEN IN THE MAP REGISTER NOT INCLUDING
                                  001E  2364 ;            THE V BIT
                                  001E  2365 ;       R7 = RECEIVED VALUE OF MAP REGISTER
                                  001E  2366 ;       R8 = PHYSICAL ADDRESS EXPECTED
                                  001E  2367 ;       R9 = ACTUAL PHYSICAL ADDRESS SELECTED
                                  001E  2368 ;
                                  001E  2369 ;       EACH MAP REGISTER IS SET EQUAL TO ITS ONE'S COMPLEMENT
                                  001E  2370 ;       MODULO 256. EG. MAP REGISTER 0 EQUALS 255, MAP REGISTER
                                  001E  2371 ;       1 EQUALS 254, MAP REGISTER 255 EQUALS 0.
                                  001E  2372 ;
                                  001E  2373 ;
                                  001E  2374 ;--
                      FFDF'   30  001E  2375         BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
            52   00000000'EF   D0  0021  2376         MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS TO R2
      0000'C2   00000000'8F   C8  0028  2377         BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                                  0031  2378         $DS_BGNSUB
                                  0031        T11_S1::
   00000000'9F   000001EC'EF   FA  0031                  CALLG   $S$, @#DS$BGNSUB
            50   000000FF 8F   D0  003C  2379         MOVL    #255,R0                 ; PFN TO LOAD IN FIRST MAP REGISTER
            50   80000000 8F   C8  0043  2380         BISL    #BIT31,R0               ; SET VALID BIT
            52   00000000'8F   C0  004A  2381         ADDL    #MAP_OFFSET,R2          ; R2 NOW POINTS TO THE MAP REGISTERS
                          5A   D4  0051  2382         CLRL    R10                     ; CLEAR MAP SELECTOR INDEX
                     624A 50   D0  0053  2383 20$:    MOVL    R0,(R2) R10             ; WRITE MAP REGISTER R10
                          50   D7  0057  2384         DECL    R0                      ; MAKE EACH MAP POINT TO ITS
                                  0059  2385                                         ; COMPLEMENT MOD 256
         F2 5A   000000FF 8F   F3  0059  2386         AOBLEQ  #255,R10,20$            ; INCREMENT MAP INDEX AND BRANCH IF
                                  0061  2387                                         ; ALL MAP REGISTERS NOT SET UP
            52   00000000'EF   D0  0061  2388         MOVL    RH_CUR_ADR,R2           ; RESET RH750 BASE ADDRESS
                          55   D4  0068  2389         CLRL    R5                      ; CLEAR MAP SELECTOR INDEX USED BY VAR
            56   000000FF 8F   D0  006A  2390         MOVL    #255,R6                 ; SET UP EXPECTED PFN
```

```
                                                
```

```
                                  00F9  2419          $DS_BGNSUB
                                  00F9        T11_S2::
00000000'9F   000001F8'EF   FA    00F9                          CALLG   $S$, a#DS$BGNSUB
                   56   01   9A    0104  2420          MOVZBL  #1,R6                   ; SET UP INITIAL MAP ENTRY
          52   00000000'8F   C0    0107  2421          ADDL    #MAP_OFFSET,R2          ; R2 NOW POINTS TO THE MAP REGISTERS
                        5A   D4    010E  2422          CLRL    R10                     ; CLEAR MAP SELECTOR INDEX
     50   56   00000000'8F   C9    0110  2423 20$:     BISL3   #VALID_BIT,R6,R0        ; VALIDATE MAP ENTRY
                  624A   50   D0    0118  2424          MOVL    R0,(R2) R10             ; WRITE MAP REGISTER R10
               56   56   01   78    011C  2425          ASHL    #1,R6,R6                ; SHIFT BIT INTO NEXT POSITION
                 EC 5A   0E   F3    0120  2426          AOBLEQ  #14,R10,20$             ; SET UP 15 MAP REGISTERS WITH A ONE
                                  0124  2427                                          ; IN SUCCESSIVE BIT POSITIONS
          52   00000000'EF   D0    0124  2428          MOVL    RH_CUR_ADR,R2           ; RESET RH750 BASE ADDRESS
               0838 C2   00   D2    012B  2429          MCOML   #0, X838(R2)            ; WRITE ONE'S INTO MAP #15
     083C C2   00000000'8F   D0    0130  2430          MOVL    #VALID_BIT, X83C(R2)    ; WRITE VALID 0 INTO MAP # 16
                        55   D4    0139  2431          CLRL    R5                      ; CLEAR MAP SELECTOR INDEX USED BY VAR
     0000'C2   00000000'8F   C8    013B  2432 30$:     BISL    #PGM_INIT,CR(R2)        ; INIT
               5A   55   09   9C    0144  2433          ROTL    #9,R5,R10               ; CREATE VIRTUAL ADDRESS
               0000'C2   5A   D0    0148  2434          MOVL    R10,VAR(R2)             ; WRITE VIRTUAL ADDRESS IN VAR
     0000'C2   00000064 8F   CE    014D  2435          MNEGL   #100,BCR(R2)            ; WRITE BCR (VALUE ARBITRARY)
     0000'C2   00000000'8F   C8    0156  2436          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
     0000'C2   00000000'8F   C8    015F  2437          BISL    #BLKSND_COMD,DR(R2)     ; BLOCK CMI COMMANDS
00000000'FF   00000000'8F   D0    0168  2438          MOVL    #READ,aDRIVE0           ; ISSUE READ COMMAND
     0000'C2   00000000'8F   C8    0173  2439          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
               53   0000'C2   D0    017C  2440          MOVL    SR(R2),R3               ; READ STATUS REGISTER
          54   00000000'8F   D0    0181  2441          MOVL    #DT_BUSY,R4             ; SET UP EXPECTED VALUE
          54   00000000'EF   C8    0188  2442          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                   54   53   D1    018F  2443          CMPL    R3,R4                   ; CHECK IF OK
                        1B   13    0192  2444          BEQL    60$                     ; BRANCH IF NO MAP PARITY ERRORS
                                  0194  2445          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                  0194  2446                          ARRAY_MSG4,-
                                  0194  2447                          PRINT_SBE
               00000000'EF   DF    0194                          PUSHAL  PRINT_SBE
               00000000'EF   DF    019A                          PUSHAL  ARRAY_MSG4
               00000000'EF   DD    01A0                          PUSHL   LUN
                        01   DD    01A6                          PUSHL   #1
          00000000'9F   04   FB    01A8                          CALLS   #$$M, a#DS$ERRHARD
                                  01AF  2448 60$:     $DS_CKLOOP      30$             ; SCOPE LOOP?
     00000000'9F   89 AF   FA    01AF                          CALLG   30$, a#DS$CKLOOP
                        55   D6    01B7  2449          INCL    R5                      ; INCREMENT MAP SELECTOR INDEX
                   10   55   B1    01B9  2450          CMPW    R5,#16                  ; HAVE ALL MAPS BEEN SELECTED?
                        04   14    01BC  2451          BGTR    70$                     ; BRANCH IF SO
                   FF79 CF   17    01BE  2452          JMP     30$                     ; LOOP IF ALL MAPS NOT SELECTED
                                  01C2  2453 70$:
                                  01C2  2454          $DS_ENDSUB
                                  01C2        T11_S2_X:
00000000'9F   000001F8'EF   FA    01C2                          CALLG   $S$, a#DS$ENDSUB
```

```
                                  01CD  2456 $DS_ENDTEST
                  50   01   D0    01CD                  MOVL    #1, R0          ; NORMAL EXIT
                                  01D0        TEST_011_X::
        00000000'9F   6E   FA     01D0                  CALLG   (SP), a#DS$BREAK
                             04   01D7                  RET                     ; RETURN TO TEST SEQUENCER
                                  01D8  2457 $DS_PAGE
                                  01D8              .SBTTL  TEST 12:  STATUS REGISTER XFER BITS TEST
                             00000000                  .PSECT  TEST_012, PAGE, NOWRT
                                  0024
                                  0024  2459 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  0024        DATA_012:
                        00000000  0024                  .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  0028        TEST_012::
                            0000  0028                  .WORD    M<>            ; ENTRY MASK
                                  002A  2460 ;++
                                  002A  2461 ; TEST DESCRIPTION:
                                  002A  2462 ;
                                  002A  2463 ; THIS TEST VERIFIES THAT THE FOLLOWING STATUS REGISTER BITS ARE NOT SA0:
                                  002A  2464 ;
                                  002A  2465 ;       MDPE,MCPE,MAPPE,INVMAP,WCK_LOW_ERR,WCK_UP_ERR,
                                  002A  2466 ;       NR_STATUS,CRD,ERR_STAT,SPE,MB_EXC
                                  002A  2467 ;
                                  002A  2468 ; IN ADDITION, THIS TEST WILL CHECK THAT THE RH750 WILL NOT RESPOND
                                  002A  2469 ; WITH ADDRESS BIT<12>SET, THAT MCPE WILL NOT SET WHEN READING
                                  002A  2470 ; THE ATTENTION SUMMARY REGISTER, AND THAT SR<31:16> IS SUPPLIED WHEN
                                  002A  2471 ; READING ANY EXTERNAL REGISTER.
                                  002A  2472 ;
                                  002A  2473 ;--
                                  002A  2474 $DS_BGNSUB
                                  002A        T12_S1::
00000000'9F   00000204'EF   FA    002A                  CALLG   $S$, a#DS$BGNSUB
                                  0035  2475
                                  0035  2476 ;
                                  0035  2477 ; Subtest one checks that the RH750 will not respond with address
                                  0035  2478 ; bit<12> set.
                                  0035  2479 ;
                       FFC8'  30  0035  2480          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0   0038  2481          MOVL    RH_CUR_ADR,R2           ; MOVE ADDRESS OF RH750 TO R2
                                  003F  2482          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                  003F  2483                  FMT_STATUS_REG,SA0_MSG
00000000'EF   00000000'EF   DE    003F                  MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02   9A     004A                  MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE    0051                  MOVAL   FMT_STATUS_REG,REG_STRING
    59   52   00001000 8F   C9    005C  2484          BISL3   #BIT12,R2,R9            ; CREATE PSUEDO NEXUS ADDRESS
                                  0064  2485          $DS_SETVEC_S    #4,MACH_CHK_SRV ; GET MACHINE CHECK VECTOR
                         00   DD  0064                  PUSHL   #0
              00000000'EF   DF    0066                  PUSHAL  MACH_CHK_SRV
                         04   DD  006C                  PUSHL   #4
        00000000'9F   03   FB     006E                  CALLS   #3, a#DS$SETVEC
        00000000'EF   01   88     0075  2486 10$:     BISB2   #1,NXM_RD_FLAG          ; SET FLAG FOR EXP RD NXM MACH CHECK
00000000'EF   0000008A'EF   DE    007C  2487          MOVAL   20$,CONT_PC             ; SET UP ADDRESS FOR RETURN
                  53   69   D0    0087  2488          MOVL    (R9),R3                 ; TRY TO READ NEXUS WITH BIT 12 SET
                                  008A  2489                                          ; THIS SHOULD CAUSE A MACHINE CHECK
      26 00000000'EF   00   E1    008A  2490 20$:     BBC     #0,NXM_RD_FLAG,30$      ; BRANCH IF RD NXM EXC OCCURRED
                                  0092  2491          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERR #1 TIME OUT FAILURE
                                  0092  2492                          ARRAY_MSG47
                         00   DD  0092                  PUSHL   #0
```

```
                00000000'EF   DF  0094                      PUSHAL  ARRAY_MSG47
                00000000'EF   DD  009A                      PUSHL   LUN
                          01  DD  00A0                      PUSHL   #1
              00000000'9F   04  FB  00A2                      CALLS   #$$M, a#DS$ERRHARD
                                    00A9  2493                          ; NEXUS RESPONDED WHICH IS INCORRECT
                                    00A9  2494          $DS_PRINTB_S    FMT_BADNEXUS,R9 ; PRINT ADDRESS OF RESPONDING NEXUS
                          59  DD  00A9                      PUSHL   R9
                00000000'EF   9F  00AB                      PUSHAB  FMT_BADNEXUS
              00000000'9F   02  FB  00B1                      CALLS   #$$N, a#DS$PRINTB
                                    00B8  2495 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
           00000000'9F   BA AF  FA  00B8                      CALLG   10$, a#DS$CKLOOP
                                    00C0  2496          $DS_CLRVEC_S    #4              ; RETURN VECTOR TO SUPERVISOR
                          04  DD  00C0                      PUSHL   #4
              00000000'9F   01  FB  00C2                      CALLS   #1, a#DS$CLRVEC
                                    00C9  2497 $DS_ENDSUB
                                    00C9       T12_S1_X:
    00000000'9F   00000204'EF   FA  00C9                      CALLG   $S$, a#DS$ENDSUB
```

```
                                00D4  2499          $DS_BGNSUB
                                00D4        T12_S2::
00000000'9F   00000210'EF   FA  00D4                CALLG   $S$, a#DS$BGNSUB
                                00DF  2500 ;++
                                00DF  2501 ; MASS BUS DATA PARITY ERROR DETECTION TEST
                                00DF  2502 ;       Massbus Data Parity is forced by reading data from the Massbus
                                00DF  2503 ;       to memory with the Invert Massbus Data Parity Bit set in the
                                00DF  2504 ;       Diagnostic Register.
                                00DF  2505 ;--
    0000'C2   00000000'8F   C8  00DF  2506 80$:     BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8  00E8  2507          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
    0000'C2   00000000'8F   C8  00F1  2508          BISL    #INVRT_MB_DPAR,DR(R2)     ; SET UP TO INV DATA PARITY ON MASSBUS
         57   00000000'EF   DE  00FA  2509          MOVAL   MIOBUFFER,R7              ; GET BUFFER ADDRESS
          59   57   00'8F   78  0101  2510          ASHL    #-PF_FIELD,R7,R9          ; PUT INTO PFN FIELD
         59   00000000'8F   C8  0106  2511          BISL    #VALID_BIT,R9             ; SET V BIT
    5A   52   00000000'8F   C1  010D  2512          ADDL3   #MAP_OFFSET,R2,R10        ; MAKE R10 POINTER TO MAP REGISTER 0
                   6A   59   D0  0115  2513          MOVL    R9,(R10)                  ; WRITE MAP REGISTER 0
         57   00000000'8F   CA  0118  2514          BICL    #MAP_PTR_MSK,R7           ; CLEAR MAP SELECT FIELD IN VAR ENTRY
              0000'C2   57   D0  011F  2515          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS FOR WRITE
              0000'C2   04   CE  0124  2516          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT = 4 INTO BCR
               00000000'8F   DD  0129  2517          PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                        04   DD  012F  2518          PUSHL   #4                        ; PUSH BYTE COUNT
                        52   DD  0131  2519          PUSHL   R2                        ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  0133  2520          CALLS   #3,MM_XFER                ; INITIATE TRANSFER
                                013A  2521
    0000'C2   00000000'8F   CA  013A  2522          BICL    #INVRT_MB_DPAR,DR(R2)     ; CLEAR INVERT PARITY BIT
              0000'C2   57   D0  0143  2523          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS FOR READ
              0000'C2   04   CE  0148  2524          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT = 4 INTO BCR
               00000000'8F   DD  014D  2525          PUSHL   #READ                     ; PUSH READ COMMAND FOR MM_XFER
                        04   DD  0153  2526          PUSHL   #4                        ; PUSH BYTE COUNT
                        52   DD  0155  2527          PUSHL   R2                        ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  0157  2528          CALLS   #3,MM_XFER                ; INITIATE TRANSFER
    54   00000000'8F   D0  015E  2529          MOVL    #MASS_DATA_PE!DATA_XFER_ABRT,R4 ; EXPECTED
    54   00000000'8F   C8  0165  2530          BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
    54   00000000'EF   C8  016C  2531          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
         53   0000'C2   D0  0173  2532          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
              54   53   D1  0178  2533          CMPL    R3,R4                     ; CHECK IF OK
                   1B   13  017B  2534          BEQL    90$                       ; BRANCH IF MASS BUS DATA PARITY ERROR
                            017D  2535          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                            017D  2536                          ARRAY_MSG15,-
                            017D  2537                          PRINT_SBE
               00000000'EF   DF  017D                       PUSHAL  PRINT_SBE
               00000000'EF   DF  0183                       PUSHAL  ARRAY_MSG15
               00000000'EF   DD  0189                       PUSHL   LUN
                        01   DD  018F                       PUSHL   #1
         00000000'9F   04   FB  0191                       CALLS   #$$M, a#DS$ERRHARD
                            0198  2538 90$:     $DS_CKLOOP      80$               ; SCOPE LOOP?
00000000'9F   FF43   CF   FA  0198                       CALLG   80$, a#DS$CKLOOP
         0000'C2   53   D0  01A1  2539          MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
         53   0000'C2   D0  01A6  2540          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                        54   D4  01AB  2541          CLRL    R4                        ; CLEAR EXP RESULTS REG
    54   00000000'EF   C8  01AD  2542          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53   D1  01B4  2543          CMPL    R3,R4                     ; CHECK RESULTS
                   1B   13  01B7  2544          BEQL    95$                       ; IF BITS CLEARED THEN BRANCH
                            01B9  2545          $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                            01B9  2546                          ARRAY_MSG6,-
                            01B9  2547                          PRINT_SBE
```

```
                00000000'EF    DF  01B9                        PUSHAL  PRINT_SBE
                00000000'EF    DF  01BF                        PUSHAL  ARRAY_MSG6
                00000000'EF    DD  01C5                        PUSHL   LUN
                         02    DD  01CB                        PUSHL   #2
             00000000'9F   04   FB  01CD                        CALLS   #$$M, a#DS$ERRHARD
                                    01D4  2548 95$:    $DS_CKLOOP      80$                 ; SCOPE LOOP?
          00000000'9F   FF07 CF   FA  01D4                        CALLG   80$, a#DS$CKLOOP
                                    01DD  2549         $DS_ENDSUB
                                    01DD       T12_S2_X:
     00000000'9F   00000210'EF   FA  01DD                        CALLG   $S$, a#DS$ENDSUB
```

```
                                01E8  2551          $DS_BGNSUB
                                01E8          T12_S3::
00000000'9F   0000021C'EF   FA   01E8                        CALLG   $S$, a#DS$BGNSUB
                                01F3  2552 ;++
                                01F3  2553 ; MASS BUS CONTROL PARITY ERROR DETECTION TEST
                                01F3  2554 ;       Massbus Control Parity Error is forced by setting Invert
                                01F3  2555 ;       Massbus Control Parity and reading a Drive Register while
                                01F3  2556 ;       in Maintenance Mode.
                                01F3  2557 ;--
    0000'C2   00000000'8F   C8  01F3  2558 100$:   BISL    #PGM_INIT,CR(R2)          ; INIT RH
    0000'C2   00000000'8F   C8  01FC  2559         BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINT MODE
    0000'C2   00000000'8F   C8  0205  2560         BISL    #INVRT_MB_CPAR,DR(R2)     ; SET UP TO INVERT MB CONTROL PARITY
         56   00000000'EF   D0  020E  2561         MOVL    DRIVE0,R6                 ; MOVE ADDRESS OF DRIVE 0 TO R6
              66   00       D0  0215  2562         MOVL    #0,(R6)                   ; DO A WRITE TO LOAD PARITY ERROR
              53   66       D0  0218  2563         MOVL    (R6),R3                   ; READ REG 0 DRIVE 0
         53   0000'C2       D0  021B  2564         MOVL    SR(R2),R3                 ; READ STATUS REGISTER--MASSBUS
                                0220  2565                                           ; CONTROL PARITY ERROR SHOULD BE SET
              0000'C2       D4  0220  2566         CLRL    DR(R2)                    ; CLEAR INVERT MB CONTROL PARITY
         54   00000000'8F   D0  0224  2567         MOVL    #MASS_CNTRL_PE,R4         ; SET UP EXPECTED VALUE REGISTER
         54   00000000'EF   C8  022B  2568         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53       D1  0232  2569         CMPL    R3,R4                     ; CHECK IF OK
                   1B       13  0235  2570         BEQL    110$                      ; BRANCH IF MASS BUS CTRL PARITY ERROR
                                0237  2571         $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                0237  2572                         ARRAY_MSG32,-
                                0237  2573                         PRINT_SBE
              00000000'EF   DF  0237                        PUSHAL  PRINT_SBE
              00000000'EF   DF  023D                        PUSHAL  ARRAY_MSG32
              00000000'EF   DD  0243                        PUSHL   LUN
                   01       DD  0249                        PUSHL   #1
         00000000'9F   04   FB  024B                        CALLS   #$$M, a#DS$ERRHARD
                                0252  2574 110$:   $DS_CKLOOP       100$              ; SCOPE LOOP?
    00000000'9F   9E AF   FA  0252                        CALLG   100$, a#DS$CKLOOP
         0000'C2   53     D0  025A  2575         MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
         53   0000'C2     D0  025F  2576         MOVL    SR(R2),R3                 ; READ STATUS REGISTER-- SHOULD BE 0
                   54     D4  0264  2577         CLRL    R4                        ; CLEAR EXP RESULTS REG
         54   00000000'EF   C8  0266  2578         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54   53       D1  026D  2579         CMPL    R3,R4                     ; CHECK RESULTS
                   1B       13  0270  2580         BEQL    115$                      ; BRANCH IF SO
                                0272  2581         $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                0272  2582                         ARRAY_MSG4,-
                                0272  2583                         PRINT_SBE
              00000000'EF   DF  0272                        PUSHAL  PRINT_SBE
              00000000'EF   DF  0278                        PUSHAL  ARRAY_MSG4
              00000000'EF   DD  027E                        PUSHL   LUN
                   02       DD  0284                        PUSHL   #2
         00000000'9F   04   FB  0286                        CALLS   #$$M, a#DS$ERRHARD
                                028D  2584 115$:   $DS_CKLOOP       100$              ; SCOPE LOOP?
    00000000'9F   FF62 CF   FA  028D                        CALLG   100$, a#DS$CKLOOP
                                0296  2585         $DS_ENDSUB
                                0296          T12_S3_X:
00000000'9F   0000021C'EF   FA  0296                        CALLG   $S$, a#DS$ENDSUB
```

```
                                02A1  2587          $DS_BGNSUB
                                02A1          T12_S4::
00000000'9F   00000228'EF   FA  02A1                        CALLG   $S$, a#DS$BGNSUB
                                02AC  2588 ;++
                                02AC  2589 ; MAP PARITY ERROR DETECTION TEST
                                02AC  2590 ;       Map Parity Error is forced by setting Invert Map Parity
                                02AC  2591 ;       and executing a transfer.
                                02AC  2592 ;--
    0000'C2   00000000'8F   C8  02AC  2593 140$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE MUT
    0000'C2   00000000'8F   C8  02B5  2594         BISL    #MAINT_MODE,CR(R2)        ; PUT MUT INTO MAINTENANCE MODE
    0000'C2   00000000'8F   C8  02BE  2595         BISL    #INVRT_MAP_PAR,DR(R2)     ; PREPARE FOR INVERTING MAP PARITY
         57   00000000'EF   DE  02C7  2596         MOVAL   MIOBUFFER,R7              ; R7 <-- BUFFER ADDRESS
          58    57    F7 8F   78  02CE  2597         ASHL    #-9,R7,R8                 ; NORMALIZE FOR MAP ENTRY
         58   00000000'8F   C8  02D3  2598         BISL    #VALID_BIT,R8             ; VALIDATE MAP ENTRY
              0800 C2    58   D0  02DA  2599         MOVL    R8, X800(R2)              ; WRITE MAP REGISTER 0
         57   00000000'8F   CA  02DF  2600         BICL    #MAP_PTR_MSK,R7           ; CLEAR MAP SELECTOR IN VA
              0000'C2    57   D0  02E6  2601         MOVL    R7,VAR(R2)                ; WRITE THE VAR
              0000'C2    04   CE  02EB  2602         MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
               00000000'8F   DD  02F0  2603         PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                         04   DD  02F6  2604         PUSHL   #4                        ; PUSH BYTE COUNT
                         52   DD  02F8  2605         PUSHL   R2                        ; PUSH RH750 ADDRESS
    00000000'EF          03   FB  02FA  2606         CALLS   #3,MM_XFER                ; INITIATE TRANSFER
               00000000'8F   D0  0301  2607         MOVL    #MAP_PE!DATA_XFER_ABRT!-  ; EXPECTED RESULTS
                         54        0307  2608                 DATA_XFER_DONE,R4
         54   00000000'EF   C8  0308  2609         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  030F  2610         MOVL    SR(R2),R3                 ; READ STATUS EXPECT MAP PARITY ERROR
                   54   53   D1  0314  2611         CMPL    R3,R4                     ; CHECK IF OK
                         57   13  0317  2612         BEQL    155$                      ; BRANCH IF MAP PARITY ERROR IS SET
                              0319  2613         $DS_ERRHARD_S   #1,LUN,-           ; PRINT ERROR
                              0319  2614                         ARRAY_MSG37,-
                              0319  2615                         PRINT_SBE
               00000000'EF   DF  0319                        PUSHAL  PRINT_SBE
               00000000'EF   DF  031F                        PUSHAL  ARRAY_MSG37
               00000000'EF   DD  0325                        PUSHL   LUN
                         01   DD  032B                        PUSHL   #1
         00000000'9F   04   FB  032D                        CALLS   #$$M, a#DS$ERRHARD
                              0334  2616 150$:   $DS_CKLOOP       140$              ; SCOPE LOOP
    00000000'9F   FF74 CF   FA  0334                        CALLG   140$, a#DS$CKLOOP
              0000'C2   53   D0  033D  2617         MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
              53   0000'C2   D0  0342  2618         MOVL    SR(R2),R3                 ; READ STATUS REGISTER--SB 0
                         54   D4  0347  2619         CLRL    R4                        ; CLEAR EXPECTED RESULTS REG
         54   00000000'EF   C8  0349  2620         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                   54   53   D1  0350  2621         CMPL    R3,R4                     ; CHECK RESULTS
                         1B   13  0353  2622         BEQL    155$                      ; IF STATUS IS OK THEN BRANCH
                              0355  2623         $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                              0355  2624                         ARRAY_MSG4,-
                              0355  2625                         PRINT_SBE
               00000000'EF   DF  0355                        PUSHAL  PRINT_SBE
               00000000'EF   DF  035B                        PUSHAL  ARRAY_MSG4
               00000000'EF   DD  0361                        PUSHL   LUN
                         02   DD  0367                        PUSHL   #2
         00000000'9F   04   FB  0369                        CALLS   #$$M, a#DS$ERRHARD
                              0370  2626 155$:   $DS_CKLOOP       140$              ; SCOPE LOOP?
    00000000'9F   FF38 CF   FA  0370                        CALLG   140$, a#DS$CKLOOP
                              0379  2627         $DS_ENDSUB
                              0379          T12_S4_X:
00000000'9F   00000228'EF   FA  0379                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0384  2629          $DS_BGNSUB
                                0384        T12_S5::
    00000000'9F   00000234'EF   FA  0384                CALLG   $S$, a#DS$BGNSUB
                                038F  2630 ;++
                                038F  2631 ; INVALID MAP DETECTION TEST
                                038F  2632 ;       Invalid Map Error is done by transferring data to the Massbus
                                038F  2633 ;       using an Invalid Map.
                                038F  2634 ;--
      0000'C2   00000000'8F     C8  038F  2635 160$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
      0000'C2   00000000'8F     C8  0398  2636         BISL    #MAINT_MODE,CR(R2)        ; PUT IT INTO MAINT MODE
           57   00000000'EF     DE  03A1  2637         MOVAL   MIOBUFFER,R7              ; SET UP VA AND MAP
            58    57    F7 8F   78  03A8  2638         ASHL    #-9,R7,R8                 ; PUT IN PFN FIELD
           57   00000000'8F     CA  03AD  2639         BICL    #MAP_PTR_MSK,R7           ; SET UP MAP POINTER
           58   00000000'8F     CA  03B4  2640         BICL    #VALID_BIT,R8             ; CLEAR VALID BIT IN MAP ENTRY
                0800 C2   58    D0  03BB  2641         MOVL    R8, X800(R2)              ; WRITE MAP REGISTER 0
                0000'C2   57    D0  03C0  2642         MOVL    R7,VAR(R2)                ; WRITE THE VAR
                0000'C2   04    CE  03C5  2643         MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
                0000'C2   57    D0  03CA  2644         MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
                 00000000'8F    DD  03CF  2645         PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                          04    DD  03D5  2646         PUSHL   #4                        ; PUSH BYTE COUNT
                          52    DD  03D7  2647         PUSHL   R2                        ; PUSH RH750 ADDRESS
           00000000'EF    03    FB  03D9  2648         CALLS   #3,MM_XFER                ; INITIATE TRANSFER WITH INVALID MAP
                00000000'8F     D0  03E0  2649         MOVL    #DATA_XFER_ABRT!-         ; EXPECTED RESULTS
                          54        03E6  2650                 MAP_INVALID,R4
           54   00000000'8F     C8  03E7  2651         BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
           54   00000000'EF     C8  03EE  2652         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                53   0000'C2    D0  03F5  2653         MOVL    SR(R2),R3                 ; MOVE STATUS REG INTO R3
                     54   53    D1  03FA  2654         CMPL    R3,R4                     ; CHECK IF OK
                          1B    13  03FD  2655         BEQL    170$                      ; BRANCH IF INVALID MAP ERROR BIT IS 1
                                03FF  2656         $DS_ERRHARD_S   #1,LUN,-           ; PRINT ERROR
                                03FF  2657                         ARRAY_MSG37,-
                                03FF  2658                         PRINT_SBE
                00000000'EF     DF  03FF                   PUSHAL  PRINT_SBE
                00000000'EF     DF  0405                   PUSHAL  ARRAY_MSG37
                00000000'EF     DD  040B                   PUSHL   LUN
                          01    DD  0411                   PUSHL   #1
           00000000'9F    04    FB  0413                   CALLS   #$$M, a#DS$ERRHARD
                                041A  2659 170$:   $DS_CKLOOP      160$               ; SCOPE LOOP?
    00000000'9F   FF71 CF       FA  041A                   CALLG   160$, a#DS$CKLOOP
                0000'C2   53    D0  0423  2660         MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
                53   0000'C2    D0  0428  2661         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                          54    D4  042D  2662         CLRL    R4                        ; CLEAR EXP RESULTS REG
           54   00000000'EF     C8  042F  2663         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                     54   53    D1  0436  2664         CMPL    R3,R4                     ; CHECK RESULTS
                          1B    13  0439  2665         BEQL    175$                      ; BRANCH IF STATUS OK
                                043B  2666         $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                                043B  2667                         ARRAY_MSG4,-
                                043B  2668                         PRINT_SBE
                00000000'EF     DF  043B                   PUSHAL  PRINT_SBE
                00000000'EF     DF  0441                   PUSHAL  ARRAY_MSG4
                00000000'EF     DD  0447                   PUSHL   LUN
                          02    DD  044D                   PUSHL   #2
           00000000'9F    04    FB  044F                   CALLS   #$$M, a#DS$ERRHARD
                                0456  2669 175$:   $DS_CKLOOP      160$               ; SCOPE LOOP?
    00000000'9F   FF35 CF       FA  0456                   CALLG   160$, a#DS$CKLOOP
                                045F  2670         $DS_ENDSUB
                                045F        T12_S5_X:
```

```
     00000000'9F   00000234'EF   FA  045F                    CALLG   $S$, a#DS$ENDSUB
```

```
                                    046A  2672          $DS_BGNSUB
                                    046A        T12_S6::
00000000'9F   00000240'EF   FA      046A                      CALLG   $S$, a#DS$BGNSUB
                                    0475  2673 ;++
                                    0475  2674 ; WRITE CHECK ERROR LOW TEST
                                    0475  2675 ;       Write Check Errors are forced by writing data onto the
                                    0475  2676 ;       Massbus while in the Maintenance Mode and then doing a
                                    0475  2677 ;       Write Check with a different data pattern in memory.
                                    0475  2678 ;--
    0000'C2   00000000'8F   C8      0475  2679 180$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH
    0000'C2   00000000'8F   C8      047E  2680          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
              00000000'EF   7C      0487  2681          CLRQ    MIOBUFFER                 ; CLEAR FIRST 8 BYTES OF BUFFER
         0000'C2    57      D0      048D  2682          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
         0000'C2    02      CE      0492  2683          MNEGL   #2,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
      58   00000000'8F   C8      0497  2684          BISL    #VALID_BIT,R8             ; SET V BIT IN MAP ENTRY
         0800 C2    58      D0      049E  2685          MOVL    R8, X800(R2)              ; WRITE MAP REGISTER 0
          00000000'8F   DD      04A3  2686          PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                     02      DD      04A9  2687          PUSHL   #2                        ; PUSH BYTE COUNT
                     52      DD      04AB  2688          PUSHL   R2                        ; PUSH RH750 ADDRESS
      00000000'EF    03      FB      04AD  2689          CALLS   #3,MM_XFER                ; INITIATE XFER--PUT 0 ONTO MASSBUS
          0000'C2    57      D0      04B4  2690          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
          0000'C2    04      CE      04B9  2691          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
      00000000'EF    03      C8      04BE  2692          BISL    #3,MIOBUFFER              ; SET TWO BITS IN OUTPUT BUFFER
           00000000'8F   DD      04C5  2693          PUSHL   #WRITE_CHECK              ; PUSH COMMAND FOR MM_XFER
                     04      DD      04CB  2694          PUSHL   #4                        ; PUSH BYTE COUNT
                     52      DD      04CD  2695          PUSHL   R2                        ; PUSH RH750 ADDRESS
      00000000'EF    03      FB      04CF  2696          CALLS   #3,MM_XFER                ; INITIATE WRITE CHECK
      54   00000000'8F   D0      04D6  2697          MOVL    #WRITE_CHK_LOW!DATA_XFER_ABRT,R4 ; EXPECTED STATUS
      54   00000000'8F   C8      04DD  2698          BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
      54   00000000'EF   C8      04E4  2699          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
          53   0000'C2   D0      04EB  2700          MOVL    SR(R2),R3                 ; READ STATUS AFTER WRITE CHECK
               54   53   D1      04F0  2701          CMPL    R3,R4                     ; CHECK IF OK
                     1B   13      04F3  2702          BEQL    190$                      ; BRANCH IF WRITE CHECK LOW IS SET
                                    04F5  2703          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                    04F5  2704                          ARRAY_MSG13,-
                                    04F5  2705                          PRINT_SBE
           00000000'EF   DF      04F5                      PUSHAL  PRINT_SBE
           00000000'EF   DF      04FB                      PUSHAL  ARRAY_MSG13
           00000000'EF   DD      0501                      PUSHL   LUN
                     01      DD      0507                      PUSHL   #1
      00000000'9F    04      FB      0509                      CALLS   #$$M, a#DS$ERRHARD
                                    0510  2706 190$:    $DS_CKLOOP      180$              ; SCOPE LOOP?
00000000'9F    FF61  CF      FA      0510                      CALLG   180$, a#DS$CKLOOP
          0000'C2    53      D0      0519  2707          MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
          53   0000'C2    D0      051E  2708          MOVL    SR(R2),R3                 ; IS STATUS REGISTER CORRECT?
                     54      D4      0523  2709          CLRL    R4                        ; CLEAR EXP RESULTS REG
      54   00000000'EF    C8      0525  2710          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
               54   53   D1      052C  2711          CMPL    R3,R4                     ; CHECK RESULTS
                     1B   13      052F  2712          BEQL    195$                      ; BRANCH IF SO
                                    0531  2713          $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                    0531  2714                          ARRAY_MSG4,-
                                    0531  2715                          PRINT_SBE
           00000000'EF   DF      0531                      PUSHAL  PRINT_SBE
           00000000'EF   DF      0537                      PUSHAL  ARRAY_MSG4
           00000000'EF   DD      053D                      PUSHL   LUN
                     02      DD      0543                      PUSHL   #2
      00000000'9F    04      FB      0545                      CALLS   #$$M, a#DS$ERRHARD
```

```
                        054C  2716 195$:   $DS_CKLOOP      180$              ; SCOPE LOOP?
        00000000'9F   FF25 CF   FA  054C                    CALLG   180$, a#DS$CKLOOP
                        0555  2717        $DS_ENDSUB
                        0555        T12_S6_X:
    00000000'9F   00000240'EF   FA  0555                    CALLG   $S$, a#DS$ENDSUB
```

```
                                0560  2719 ;++
                                0560  2720 ; WRITE CHECK ERROR HIGH TEST
                                0560  2721 ;       Write Check Errors are forced by writing data onto the
                                0560  2722 ;       Massbus while in the Maintenance Mode and then doing a
                                0560  2723 ;       Write Check with a different data pattern in memory.
                                0560  2724 ;--
                                0560  2725          $DS_BGNSUB
                                0560        T12_S7::
00000000'9F   0000024C'EF   FA  0560                CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  056B  2726 200$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750 UNDER TEST
    0000'C2   00000000'8F   C8  0574  2727          BISL    #MAINT_MODE,CR(R2)      ; PUT THE MUT INTO MAINTENANCE MODE
              00000000'EF   7C  057D  2728          CLRQ    MIOBUFFER               ; EMPTY MIDDLE IO BUFFER
            0000'C2   57    D0  0583  2729          MOVL    R7,VAR(R2)              ; WRITE VIRTUAL ADDRESS INTO VAR
            0000'C2   02    CE  0588  2730          MNEGL   #2,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
             00000000'8F    DD  058D  2731          PUSHL   #WRITE                  ; PUSH WRITE COMMAND FOR MM_XFER
                    02      DD  0593  2732          PUSHL   #2                      ; PUSH BYTE COUNT FOR MM_XFER
                    52      DD  0595  2733          PUSHL   R2                      ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  0597  2734          CALLS   #3,MM_XFER              ; INITIATE WRITE--PUT 0 ON THE MASSBUS
            0000'C2   57    D0  059E  2735          MOVL    R7,VAR(R2)              ; WRITE VIRTUAL ADDRESS INTO VAR
            0000'C2   04    CE  05A3  2736          MNEGL   #4,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
         00000001'EF   03   D0  05A8  2737          MOVL    #3,MIOBUFFER + 1        ; SET UP BAD DATA IN UPPER HALF OF
                               05AF  2738                                          ; WRITE CHECK WORD
              00000000'8F   DD  05AF  2739          PUSHL   #WRITE_CHECK            ; PUSH COMMAND FOR MM_XFER
                    04      DD  05B5  2740          PUSHL   #4                      ; PUSH BYTE COUNT
                    52      DD  05B7  2741          PUSHL   R2                      ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  05B9  2742          CALLS   #3,MM_XFER              ; INITIATE WRITE CHECK WITH BAD DATA
                               05C0  2743                                          ; IN MEMORY
      54   00000000'8F   D0  05C0  2744          MOVL    #WRITE_CHK_HIGH!DATA_XFER_ABRT,R4 ; EXPECTED STATUS
      54   00000000'8F   C8  05C7  2745          BISL    #DATA_XFER_DONE,R4      ; EXPECT THIS BIT ALSO
      54   00000000'EF   C8  05CE  2746          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
          53   0000'C2   D0  05D5  2747          MOVL    SR(R2),R3               ; CHECK STATUS REGSITER FOR WCK ERROR
               54   53    D1  05DA  2748          CMPL    R3,R4                   ; CHECK IF OK
                    1B    13  05DD  2749          BEQL    210$                    ; BRANCH IF WCK HIGH ERROR IS SET
                               05DF  2750          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                               05DF  2751                          ARRAY_MSG4,-
                               05DF  2752                          PRINT_SBE
              00000000'EF   DF  05DF                        PUSHAL  PRINT_SBE
              00000000'EF   DF  05E5                        PUSHAL  ARRAY_MSG4
              00000000'EF   DD  05EB                        PUSHL   LUN
                    01      DD  05F1                        PUSHL   #1
         00000000'9F   04   FB  05F3                        CALLS   #$$M, a#DS$ERRHARD
                               05FA  2753 210$:   $DS_CKLOOP      200$             ; SCOPE LOOP?
00000000'9F   FF6D  CF     FA  05FA                        CALLG   200$, a#DS$CKLOOP
         0000'C2   53    D0  0603  2754          MOVL    R3,SR(R2)               ; WRITE STATUS BITS TO CLEAR
         53   0000'C2    D0  0608  2755          MOVL    SR(R2),R3               ; IS STATUS REGISTER CORRECT?
                    54    D4  060D  2756          CLRL    R4                      ; CLEAR EXP RESULTS REG
      54   00000000'EF   C8  060F  2757          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
               54   53    D1  0616  2758          CMPL    R3,R4                   ; CHECK RESULTS
                    1B    13  0619  2759          BEQL    215$                    ; BRANCH IF SO
                               061B  2760          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                               061B  2761                          ARRAY_MSG4,-
                               061B  2762                          PRINT_SBE
              00000000'EF   DF  061B                        PUSHAL  PRINT_SBE
              00000000'EF   DF  0621                        PUSHAL  ARRAY_MSG4
              00000000'EF   DD  0627                        PUSHL   LUN
                    02      DD  062D                        PUSHL   #2
         00000000'9F   04   FB  062F                        CALLS   #$$M, a#DS$ERRHARD
```

```
                        0636  2763 215$:   $DS_CKLOOP      200$             ; SCOPE LOOP?
      00000000'9F   FF31 CF   FA  0636                      CALLG   200$, a#DS$CKLOOP
                        063F  2764          $DS_ENDSUB
                        063F       T12_S7_X:
   00000000'9F   0000024C'EF   FA  063F                      CALLG   $S$, a#DS$ENDSUB
```

```
                                064A  2766 ;++
                                064A  2767 ; NO RESPONSE
                                064A  2768 ;       NR Status is forced by attempting to transfer data from
                                064A  2769 ;       non-existent memory.
                                064A  2770 ;--
                                064A  2771         $DS_BGNSUB
                                064A       T12_S8::
00000000'9F   00000258'EF   FA  064A                    CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0655  2772 220$:   BISL    #PGM_INIT,CR(R2)        ; INIT THE RH
    0000'C2   00000000'8F   C8  065E  2773         BISL    #MAINT_MODE,CR(R2)      ; PUT INTO MAINTENANCE MODE
        57    00E00000 8F   D0  0667  2774         MOVL    # XE00000,R7            ; GET NON EXISTANT ADDRESS
         58    57    00'8F   78  066E  2775         ASHL    #-PF_FIELD,R7,R8        ; GET PFN
        58    00000000'8F   C8  0673  2776         BISL    #VALID_BIT,R8           ; SET VALID BIT IN MAP ENTRY
        57    00000000'8F   CA  067A  2777         BICL    #MAP_PTR_MSK,R7         ; CLEAR MAP POINTER IN VIRTUAL ADDRESS
            0800 C2   58    D0  0681  2778         MOVL    R8, X800(R2)            ; WRITE PFN OF ILLEGAL TRANSFER
                              0686  2779                                         ; ADDRESS INTO MAP REG 0
            0000'C2   57    D0  0686  2780         MOVL    R7,VAR(R2)              ; WRITE VIRTUAL ADDRESS INTO VAR
            0000'C2   08    CE  068B  2781         MNEGL   #8,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
             00000000'8F    DD  0690  2782         PUSHL   #WRITE                  ; PUSH WTD COMMAND FOR MM_XFER
                       08    DD  0696  2783         PUSHL   #8                      ; PUSH BYTE COUNT
                       52    DD  0698  2784         PUSHL   R2                      ; PUSH RH750 ADDRESS
        00000000'EF    03    FB  069A  2785         CALLS   #3,MM_XFER              ; INITIATE WRITE TO DRIVE
                              06A1  2786                                         ; A READ DATA TIME OUT WILL OCCUR
        54    00000000'8F   D0  06A1  2787         MOVL    #DATA_XFER_ABRT,R4      ; EXPECTED STATUS
              00000000'8F   C8  06A8  2788         BISL    #DATA_XFER_DONE!-
                       54         06AE  2789                 NO_RESPONSE,R4
                              06AF  2790                                         ; EXPECT DATA XFER COMPLETE
        54    00000000'EF   C8  06AF  2791         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
            53    0000'C2   D0  06B6  2792         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                  54    53    D1  06BB  2793         CMPL    R3,R4                   ; CHECK IF OK
                        1B    13  06BE  2794         BEQL    235$                    ; BRANCH IF INTERFACE SEQ TIMEOUT IS 1
                              06C0  2795         $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                              06C0  2796                         ARRAY_MSG48,-
                              06C0  2797                         PRINT_SBE
             00000000'EF      DF  06C0                    PUSHAL  PRINT_SBE
             00000000'EF      DF  06C6                    PUSHAL  ARRAY_MSG48
             00000000'EF      DD  06CC                    PUSHL   LUN
                       01     DD  06D2                    PUSHL   #1
        00000000'9F    04     FB  06D4                    CALLS   #$$M, a#DS$ERRHARD
                              06DB  2798 235$:   $DS_CKLOOP      220$            ; SCOPE LOOP?
00000000'9F   FF76 CF   FA  06DB                    CALLG   220$, a#DS$CKLOOP
          0000'C2   53    D0  06E4  2799         MOVL    R3,SR(R2)               ; WRITE STATUS BITS TO CLEAR
          53    0000'C2   D0  06E9  2800         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                  54    D4  06EE  2801         CLRL    R4                      ; CLEAR EXP RESULTS REG
        54    00000000'EF   C8  06F0  2802         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              54    53    D1  06F7  2803         CMPL    R3,R4                   ; CHECK RESULTS
                    1B    13  06FA  2804         BEQL    240$                    ; IF NO RESIDUAL BITS SET BRANCH
                              06FC  2805         $DS_ERRHARD_S   #2,LUN,-         ; PRINT ERROR
                              06FC  2806                         ARRAY_MSG3,-
                              06FC  2807                         PRINT_SBE
             00000000'EF      DF  06FC                    PUSHAL  PRINT_SBE
             00000000'EF      DF  0702                    PUSHAL  ARRAY_MSG3
             00000000'EF      DD  0708                    PUSHL   LUN
                       02     DD  070E                    PUSHL   #2
        00000000'9F    04     FB  0710                    CALLS   #$$M, a#DS$ERRHARD
                              0717  2808 240$:   $DS_CKLOOP      220$            ; SCOPE LOOP?
00000000'9F   FF3A CF   FA  0717                    CALLG   220$, a#DS$CKLOOP
```

```
                                0720  2809          $DS_ENDSUB
                                0720          T12_S8_X:
    00000000'9F   00000258'EF   FA  0720                        CALLG   $S$, a#DS$ENDSUB
```

```
                                    072B  2811 $DS_BGNSUB
                                    072B       T12_S9::
    00000000'9F   00000264'EF   FA  072B                    CALLG   $S$, a#DS$BGNSUB
                                    0736  2812 ;++
                                    0736  2813 ; CORRECTED READ DATA BIT
                                    0736  2814 ;       This subtest checks that the Corrected Read Data Bit, SR<29>, is
                                    0736  2815 ;       not SA0, This is accomplished by setting up the memory controller
                                    0736  2816 ;       to force a single bit error in a longword address. A W-T-D command
                                    0736  2817 ;       is then issued from the RH750 to read that memory address containing
                                    0736  2818 ;       the CRD.
                                    0736  2819 ;--
                                    0736  2820
          52   00000000'EF   D0  0736  2821          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
       0000'C2   00000000'8F   C8  073D  2822 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
          56   00000000'EF   DE  0746  2823          MOVAL   MIOBUFFER,R6            ; STORE BUFFER ADDRESS
       0000'C2   00000000'8F   C8  074D  2824          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
            57    56   00'8F   78  0756  2825          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
          57   00000000'8F   C8  075B  2826          BISL    #VALID_BIT,R7           ; SET V-BIT IN MAP ENTRY
       5A 52   00000000'8F   C1  0762  2827          ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
                   6A   57   D0  076A  2828          MOVL    R7,(R10)                ; WRITE PFN TO MAP REG 0
    0000'C2   56   00000000'8F   CB  076D  2829          BICL3   #MAP_PTR_MSK,R6,VAR(R2) ; LOAD VAR
                 0000'C2   02   CE  0777  2830          MNEGL   #2,BCR(R2)              ; SET BCR FOR 2 BYTE TRANSFER
          55   00000000'EF   D0  077C  2831          MOVL    DRIVE0,R5               ; GET ADDRESS OF DRIVE
               00000000'8F   D0  0783  2832          MOVL    #CRTED_READ_DATA!-      ; SET UP EXPECTED RESULTS IN
                        54   D0  0789  2833                  DATA_XFER_DONE,R4       ; R4
00000000'EF    56   000001FF 8F   CB  078A  2834          BICL3   # X1FF,R6,MEM_CSR1      ; PUT PFN IN MEMORY CONTROLLER
     00000000'EF   00000000'8F   C8  0796  2835          BISL    #PM!EDM,MEM_CSR1        ; TURN ON ECC DISABLE
                     66   01   D0  07A1  2836          MOVL    #1,(R6)                 ; WRITE A ONE INTO THE PAGE
                          66   D5  07A4  2837          TSTL    (R6)                    ; READ LOCATION
     00000000'EF   00000000'8F   CA  07A6  2838          BICL    #EDM,MEM_CSR1           ; TURN OFF ECC DISABLE
                          66   D4  07B1  2839          CLRL    (R6)                    ; SIMULATE A SINGLE-BIT ERROR
00000000'EF   00000000'8F   C8  07B3  2840          BISL    #PM!DCM!ERCE,MEM_CSR1   ; TURN ON DIAGNOSTIC CHECK MODE
          65   00000000'8F   D0  07BE  2841          MOVL    #WRITE,(R5)             ; EXECUTE CMI READ TO CRD PAGE
       0000'C2   00000000'8F   C8  07C5  2842          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
       0000'C2   00000000'8F   C8  07CE  2843          BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
       0000'C2   00000000'8F   CA  07D7  2844          BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
       0000'C2   00000000'8F   C8  07E0  2845          BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
       0000'C2   00000000'8F   CA  07E9  2846          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
               00000000'EF   D4  07F2  2847          CLRL    MEM_CSR1                ; CLEAR CSR1
          54   00000000'EF   C8  07F8  2848          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
             53   0000'C2   D0  07FF  2849          MOVL    SR(R2),R3               ; READ SR
                  54   53   D1  0804  2850          CMPL    R3,R4                   ; CHECK FOR CRD BIT SET
                       1B   13  0807  2851          BEQL    20$                     ; BRANCH IS CRD BIT SET
                            0809  2852          $DS_ERRHARD_S   #1,LUN,-            ; REPORT ERROR
                            0809  2853                          ARRAY_MSG42,-
                            0809  2854                          PRINT_SBE       ;
               00000000'EF   DF  0809                      PUSHAL  PRINT_SBE
               00000000'EF   DF  080F                      PUSHAL  ARRAY_MSG42
               00000000'EF   DD  0815                      PUSHL   LUN
                        01   DD  081B                      PUSHL   #1
          00000000'9F   04   FB  081D                      CALLS   #$$M, a#DS$ERRHARD
                            0824  2855 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
     00000000'9F   FF15 CF   FA  0824                      CALLG   10$, a#DS$CKLOOP
                            082D  2856 $DS_ENDSUB
                            082D       T12_S9_X:
00000000'9F   00000264'EF   FA  082D                      CALLG   $S$, a#DS$ENDSUB
```

```
                                0838  2858 $DS_BGNSUB
                                0838       T12_S10::
    00000000'9F   00000270'EF   FA  0838                CALLG   $S$, a#DS$BGNSUB
                                0843  2859 ;++
                                0843  2860 ; ERR STATUS BIT
                                0843  2861 ;       This subtest checks that the Error Status Bit, SR<03>, is
                                0843  2862 ;       not SA0, This is accomplished by setting up the memory controller
                                0843  2863 ;       to force a double-bit error in a longword address. A W-T-D command
                                0843  2864 ;       is then issued from the RH750 to read that memory address containing
                                0843  2865 ;       the double-bit error
                                0843  2866 ;--
                                0843  2867
          52   00000000'EF   D0  0843  2868        MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
      0000'C2  00000000'8F   C8  084A  2869 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
          56   00000000'EF   DE  0853  2870        MOVAL   MIOBUFFER,R6            ; STORE BUFFER ADDRESS
      0000'C2  00000000'8F   C8  085A  2871        BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
           57    56   00'8F   78  0863  2872        ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
          57   00000000'8F   C8  0868  2873        BISL    #VALID_BIT,R7           ; SET V-BIT IN MAP ENTRY
      5A  52   00000000'8F   C1  086F  2874        ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
                   6A   57   D0  0877  2875        MOVL    R7,(R10)                ; WRITE PFN TO MAP REG 0
  0000'C2  56   00000000'8F   CB  087A  2876        BICL3   #MAP_PTR_MSK,R6,VAR(R2) ; LOAD VAR
              0000'C2   02   CE  0884  2877        MNEGL   #2,BCR(R2)              ; SET BCR FOR 2 BYTE TRANSFER
          55   00000000'EF   D0  0889  2878        MOVL    DRIVE0,R5               ; GET ADDRESS OF DRIVE
                             D0  0890  2879        MOVL    #DATA_XFER_ABRT!-        ; SET UP EXPECTED RESULTS
                             0891  2880                DATA_XFER_DONE!-         ; SET UP EXPECTED RESULTS
          54   00000000'8F   0891  2881                ERR_STAT,R4              ;
00000000'EF   56   000001FF 8F   CB  0897  2882        BICL3   # XIFF,R6,MEM_CSR1      ; PUT PFN IN MEMORY CONTROLLER
    00000000'EF   00000000'8F   C8  08A3  2883        BISL    #PM!EDM,MEM_CSR1        ; TURN ON ECC DISABLE
                   66   03   D0  08AE  2884        MOVL    #3,(R6)                 ; WRITE A THREE INTO THE PAGE
                        66   D5  08B1  2885        TSTL    (R6)                    ; READ LOCATION
00000000'EF   00000000'8F   CA  08B3  2886        BICL    #EDM,MEM_CSR1           ; TURN OFF ECC DISABLE
                        66   D4  08BE  2887        CLRL    (R6)                    ; SIMULATE A DOUBLE-BIT ERROR
00000000'EF   00000000'8F   C8  08C0  2888        BISL    #PM!DCM!ERCE,MEM_CSR1   ; TURN ON DIAGNOSTIC CHECK MODE
          65   00000000'8F   D0  08CB  2889        MOVL    #WRITE,(R5)             ; EXECUTE CMI READ TO CRD PAGE
      0000'C2  00000000'8F   C8  08D2  2890        BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
      0000'C2  00000000'8F   C8  08DB  2891        BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
      0000'C2  00000000'8F   CA  08E4  2892        BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
      0000'C2  00000000'8F   C8  08ED  2893        BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
      0000'C2  00000000'8F   CA  08F6  2894        BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
               00000000'EF   D4  08FF  2895        CLRL    MEM_CSR1                ; CLEAR CSR1
          54   00000000'EF   C8  0905  2896        BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  090C  2897        MOVL    SR(R2),R3               ; READ SR
                   54   53   D1  0911  2898        CMPL    R3,R4                   ; CHECK FOR ERROR STATUS SET
                        1B   13  0914  2899        BEQL    20$                     ; BRANCH IS ERROR STATUS IS SET
                             0916  2900        $DS_ERRHARD_S   #1,LUN,-          ; REPORT ERROR
                             0916  2901                        ARRAY_MSG49,-
                             0916  2902                        PRINT_SBE       ;
               00000000'EF   DF  0916                PUSHAL  PRINT_SBE
               00000000'EF   DF  091C                PUSHAL  ARRAY_MSG49
               00000000'EF   DD  0922                PUSHL   LUN
                        01   DD  0928                PUSHL   #1
          00000000'9F   04   FB  092A                CALLS   #$$M, a#DS$ERRHARD
00000000'EF   E0000000 8F   C8  0931  2903 20$:    BISL2   # XE0000000, MEM_CSR0   ;  1.8  SET <31:29> TO CLEAR
                             093C  2904                                        ; ERROR STATUS FLAGS
               00000000'EF   D4  093C  2905        CLRL    MEM_CSR0                ; NOW REGISTER MAY BE CLEARED
                             0942  2906 30$:    $DS_CKLOOP      10$               ; SCOPE LOOP?
    00000000'9F   FF04 CF   FA  0942                CALLG   10$, a#DS$CKLOOP
```

```
                                094B  2907 $DS_ENDSUB
                                094B       T12_S10_X:
    00000000'9F   00000270 EF   FA  094B                       CALLG   $S$, a#DS$ENDSUB
```

```
                                0956  2909 $DS_BGNSUB
                                0956       T12_S11::
00000000'9F   0000027C'EF   FA  0956                       CALLG   $S$, a#DS$BGNSUB
                                0961  2910 ;++
                                0961  2911 ; MCPE NOT SET WHEN READING THE ASR TEST
                                0961  2912 ;       This subtest checks that the RH750 will not set MCPE when reading the
                                0961  2913 ;       Attention Summary Register. This is accomplished by inverting Massbus
                                0961  2914 ;       Control Parity Generation, reading the ASR, and then checking that a
                                0961  2915 ;       MCPE did not occur.
                                0961  2916 ;--
                                0961  2917
      52   00000000'EF    D0   0961  2918 10$:     MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
0000'C2    00000000'8F    C8   0968  2919         BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
0000'C2    00000000'8F    C8   0971  2920         BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
0000'C2    00000000'8F    C8   097A  2921         BISL    #INVRT_MB_CPAR,DR(R2)   ; WRITE INVERT CONTROL PARITY
          53    0000'C2    D0   0983  2922         MOVL    ASR_OFFSET(R2),R3       ; READ ATTENTION SUMMARY REGISTER
          54    0000'C2    D0   0988  2923         MOVL    SR(R2),R4               ; READ SR
                0000'C2    D4   098D  2924         CLRL    DR(R2)                  ; CLEAR ISPG
            17 54    11    E1   0991  2925         BBC     #17,R4,20$              ; BRANCH IF NO MCPE
                                0995  2926         $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                0995  2927                         ARRAY_MSG60
                      00    DD   0995                       PUSHL   #0
              00000000'EF   DF   0997                       PUSHAL  ARRAY_MSG60
              00000000'EF   DD   099D                       PUSHL   LUN
                      01    DD   09A3                       PUSHL   #1
      00000000'9F     04    FB   09A5                       CALLS   #$$M, a#DS$ERRHARD
                                09AC  2928 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
   00000000'9F    B2 AF    FA   09AC                       CALLG   10$, a#DS$CKLOOP
                                09B4  2929 $DS_ENDSUB
                                09B4       T12_S11_X:
00000000'9F   0000027C'EF   FA  09B4                       CALLG   $S$, a#DS$ENDSUB
```

```
                                09BF  2931 $DS_BGNSUB
                                09BF       T12_S12::
00000000'9F   00000288'EF   FA  09BF                       CALLG   $S$, a#DS$BGNSUB
                                09CA  2932 ;++
                                09CA  2933 ; SR<31:16> SUPPLIED TEST
                                09CA  2934 ;       This subtest checks that the RH750 will supply Status Register<31:16>,
                                09CA  2935 ;       when reading any External Register. This is accomplished by forcing the
                                09CA  2936 ;       Attention bit,SR<16>, and  the Massbus Control Parity bit,SR<17>, to be
                                09CA  2937 ;       set while the rest of SR<31:18> are zero, then a Drive Register is read
                                09CA  2938 ;       and CMI<31:16> is compared to SR<31:16>.
                                09CA  2939 ;--
                                09CA  2940
    0000'C2   00000000'8F   C8  09CA  2941 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
    0000'C2   00000000'8F   C8  09D3  2942         BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
    0000'C2   00000000'8F   C8  09DC  2943         BISL    #SIM_ATTN,DR(R2)        ; CAUSE ATTN TO BE SET IN THE SR
    0000'C2   00000000'8F   C8  09E5  2944         BISL    #INVRT_MB_CPAR,DR(R2)   ; SET UP TO INVERT MB CONTROL PARITY
         54   00030000 8F   D0  09EE  2945         MOVL    # X30000,R4             ; LOAD EXPECTED RESULTS INTO R4
         56   00000000'EF   D0  09F5  2946         MOVL    DRIVE0,R6               ; LOAD ADDRESS OF DRIVE 0
              66    00      D0  09FC  2947         MOVL    #0,(R6)                 ; DO A WRITE TO REG 0 TO CAUSE PAR ER
              53    66      D0  09FF  2948         MOVL    (R6),R3                 ; READ DRIVE REGISTER 0
         53   00000000'8F   CA  0A02  2949         BICL2   #BYTE0!BYTE1,R3         ; STORE CMI<31:16>
              54    53      D1  0A09  2950         CMPL    R3,R4                   ; DOES CMI<31:16> EQUAL SR<31:16>?
                    1B      13  0A0C  2951         BEQL    20$                     ; BRANCH IF DATA RESULTS EQUAL
                                0A0E  2952         $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                0A0E  2953                         ARRAY_MSG14,-
                                0A0E  2954                         PRINT_SBE
              00000000'EF       DF  0A0E                       PUSHAL  PRINT_SBE
              00000000'EF       DF  0A14                       PUSHAL  ARRAY_MSG14
              00000000'EF       DD  0A1A                       PUSHL   LUN
                        01      DD  0A20                       PUSHL   #1
         00000000'9F    04      FB  0A22                       CALLS   #$$M, a#DS$ERRHARD
                                0A29  2955 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F    9E AF       FA  0A29                       CALLG   10$, a#DS$CKLOOP
                                0A31  2956 $DS_ENDSUB
                                0A31       T12_S12_X:
00000000'9F   00000288'EF   FA  0A31                       CALLG   $S$, a#DS$ENDSUB
```

```
                                0A3C  2958 $DS_BGNSUB
                                0A3C       T12_S13::
00000000'9F   00000294'EF   FA  0A3C                   CALLG   $S$, a#DS$BGNSUB
                                0A47  2959 ;++
                                0A47  2960 ; MASSBUS EXCEPTION BIT TEST
                                0A47  2961 ;       This subtest performs a dynamic test of SR<07>, MB Exception.
                                0A47  2962 ;       A Block Sending Command is issued and a CMI transaction is
                                0A47  2963 ;       initiated. This should result in MB EXC being set.
                                0A47  2964 ;--
                                0A47  2965
      52   00000000'EF   D0     0A47  2966          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
0000'C2    00000000'8F   C8     0A4E  2967 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
0000'C2    00000000'8F   C8     0A57  2968          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
      57   00000000'EF   DE     0A60  2969          MOVAL   MIOBUFFER,R7            ; GET BUFFER ADDRESS
       59   57   00'8F   78     0A67  2970          ASHL    #-PF_FIELD,R7,R9        ; EXTRACT PFN
      59   00000000'8F   C8     0A6C  2971          BISL    #VALID_BIT,R9           ; SET V-BIT
5A    52   00000000'8F   C1     0A73  2972          ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
               6A   59   D0     0A7B  2973          MOVL    R9,(R10)                ; WRITE MAP REG 0
      57   00000000'8F   CA     0A7E  2974          BICL    #MAP_PTR_MSK,R7         ; CLEAR MAP SEL FIELD IN VAR
           0000'C2   57   D0     0A85  2975          MOVL    R7,VAR(R2)              ; WRITE VAR
           0000'C2   1E   CE     0A8A  2976          MNEGL   #30,BCR(R2)             ; WRITE BYTE COUNT FOR 30 BYTES
      56   00000000'EF   D0     0A8F  2977          MOVL    DRIVE0,R6               ; GET ADDRESS OF DRIVE 0
                    58   D4     0A96  2978          CLRL    R8                      ; CLEAR SCLK COUNTER
           00000000'8F   D0     0A98  2979          MOVL    #DATA_XFER_ABRT!DATA_XFER_DONE!- ; LOAD EXP RESULTS
                    54   0A9E  2980                  MASS_ECP,R4
0000'C2    00000000'8F   C8     0A9F  2981          BISL    #BLKSND_COMD,DR(R2)     ; DO NOT ALLOW CMI TRANSACTIONS
      66   00000000'8F   D0     0AA8  2982          MOVL    #READ,(R6)              ; ATTEMPT CMI WRITE
0000'C2    00000000'8F   C8     0AAF  2983          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
0000'C2    00000000'8F   C8     0AB8  2984 20$:     BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
0000'C2    00000000'8F   CA     0AC1  2985          BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
            EA 58   03   F3     0ACA  2986          AOBLEQ  #3,R8,20$               ; XFER 8 BYTES
0000'C2    00000000'8F   C8     0ACE  2987          BISL    #SIM_EXCEPT,DR(R2)      ; SET SIMULATE EXCEPTION
0000'C2    00000000'8F   CA     0AD7  2988          BICL    #BLKSND_COMD,DR(R2)     ; CLEAR BSC
0000'C2    00000000'8F   C8     0AE0  2989          BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
0000'C2    00000000'8F   CA     0AE9  2990          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
      54   00000000'EF   C8     0AF2  2991          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
           53   0000'C2   D0     0AF9  2992          MOVL    SR(R2),R3               ; READ SR
                54   53   D1     0AFE  2993          CMPL    R3,R4                   ; CHECK FOR EXP RESULTS
                     1B   13     0B01  2994          BEQL    30$                     ; BRANCH IF OK
                                0B03  2995          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                0B03  2996                          ARRAY_MSG34,-
                                0B03  2997                          PRINT_SBE
           00000000'EF   DF     0B03                        PUSHAL  PRINT_SBE
           00000000'EF   DF     0B09                        PUSHAL  ARRAY_MSG34
           00000000'EF   DD     0B0F                        PUSHL   LUN
                    01   DD     0B15                        PUSHL   #1
      00000000'9F   04   FB     0B17                        CALLS   #$$M, a#DS$ERRHARD
                                0B1E  2998 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
00000000'9F   FF2C CF   FA     0B1E                        CALLG   10$, a#DS$CKLOOP
           55   0000'C2   D0     0B27  2999          MOVL    BCR(R2),R5              ; READ BCR
      55   00000000'8F   CA     0B2C  3000          BICL    #BYTE2!BYTE3,R5         ; STORE CMI BYTE COUNT
      55   0000FFEA 8F   D1     0B33  3001          CMPL    # XFFEA,R5             ; CHECK CORRECT COUNT
                    2A   13     0B3A  3002          BEQL    40$                     ; BRANCH IF CORRECT
      00000000'EF   01   D0     0B3C  3003          MOVL    #1,IBC_MSG              ; SET UP TO REPORT ERROR
                                0B43  3004          $DS_ERRHARD_S   #2,LUN,-        ; REPORT ERROR
                                0B43  3005                          ARRAY_MSG15,-
                                0B43  3006                          PRINT_IBCERR,-  ; MAKE USE OF IBC ERR REPORTING
```

```
                                  0B43  3007                          # XFFEA,R5       ; MECHANISM
                        55   DD   0B43                        PUSHL   R5
                0000FFEA 8F   DD   0B45                        PUSHL   # XFFEA
                00000000'EF   DF   0B4B                        PUSHAL  PRINT_IBCERR
                00000000'EF   DF   0B51                        PUSHAL  ARRAY_MSG15
                00000000'EF   DD   0B57                        PUSHL   LUN
                        02   DD   0B5D                        PUSHL   #2
          00000000'9F   06   FB   0B5F                        CALLS   #$$M, a#DS$ERRHARD
                                  0B66  3008 40$:    $DS_CKLOOP      10$              ; SCOPE LOOP?
     00000000'9F   FEE4 CF   FA   0B66                        CALLG   10$, a#DS$CKLOOP
                                  0B6F  3009 $DS_ENDSUB
                                  0B6F       T12_S13_X:
  00000000'9F   00000294'EF   FA   0B6F                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0B7A  3011 $DS_BGNSUB
                                0B7A       T12_S14::
    00000000'9F   000002A0'EF   FA  0B7A                    CALLG   $S$, a#DS$BGNSUB
                                0B85  3012 ;++
                                0B85  3013 ; SPE BIT TEST
                                0B85  3014 ;       This subtest checks the Silo Parity Error bit, SR<14>, by
                                0B85  3015 ;       setting the Invert Silo Parity Generation bit, which is DR<22>,
                                0B85  3016 ;       and then executing a W-T-D function.
                                0B85  3017 ;--
                                0B85  3018
          52   00000000'EF    D0  0B85  3019 10$:    MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
     0000'C2   00000000'8F    C8  0B8C  3020         BISL    #PGM_INIT,CR(R2)        ; INIT THE RH750
     0000'C2   00000000'8F    C8  0B95  3021         BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
     0000'C2   00000000'8F    C8  0B9E  3022         BISL    #INVRT_SILO_PAR,DR(R2)  ; SET ISPG
          57   00000000'EF    DE  0BA7  3023         MOVAL   MIOBUFFER,R7            ; GET BUFFER ADDRESS
00000000'EF    00000100 8F    D0  0BAE  3024         MOVL    # X100,MIOBUFFER        ; WRITE 100 INTO BUFFER
          59   57   00'8F    78  0BB9  3025         ASHL    #-PF_FIELD,R7,R9        ; EXTRACT PFN
          59   00000000'8F    C8  0BBE  3026         BISL    #VALID_BIT,R9           ; SET V-BIT
     5A   52   00000000'8F    C1  0BC5  3027         ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
               6A   59        D0  0BCD  3028         MOVL    R9,(R10)                ; WRITE MAP REG 0
          57   00000000'8F    CA  0BD0  3029         BICL    #MAP_PTR_MSK,R7         ; CLEAR MAP SELECT FIELD IN VAR
          0000'C2   57        D0  0BD7  3030         MOVL    R7,VAR(R2)              ; WRITE VAR
          0000'C2   04        CE  0BDC  3031         MNEGL   #4,BCR(R2)              ; WRITE BYTE COUNT FOR 4
          54   00000000'EF    D0  0BE1  3032         MOVL    DRIVE0,R4               ; GET ADDRESS OF DRIVE 0
          64   00000000'8F    D0  0BE8  3033         MOVL    #WRITE,(R4)             ; EXECUTE W-T-D
     0000'C2   00000000'8F    C8  0BEF  3034         BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
     0000'C2   00000000'8F    C8  0BF8  3035         BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
     0000'C2   00000000'8F    CA  0C01  3036         BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
     0000'C2   00000000'8F    C8  0C0A  3037         BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
     0000'C2   00000000'8F    CA  0C13  3038         BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
     0000'C2   00000000'8F    CA  0C1C  3039         BICL    #INVRT_SILO_PAR,DR(R2)  ; CLEAR ISPG
          54   00000000'8F    D0  0C25  3040         MOVL    #DATA_XFER_ABRT,R4      ; EXPECT ABORT
          54   00000000'8F    C8  0C2C  3041         BISL    #SILO_PE!DATA_XFER_DONE,R4; EXPECT SPE
          54   00000000'EF    C8  0C33  3042         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
          53   0000'C2        D0  0C3A  3043         MOVL    SR(R2),R3               ; READ SR
               54   53        D1  0C3F  3044         CMPL    R3,R4                   ; CHECK FOR SPE
                    1B        13  0C42  3045         BEQL    20$                     ; BRANCH IF SPE SET
                                  0C44  3046         $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                  0C44  3047                         ARRAY_MSG15,-
                                  0C44  3048                         PRINT_SBE    ;
               00000000'EF    DF  0C44                        PUSHAL  PRINT_SBE
               00000000'EF    DF  0C4A                        PUSHAL  ARRAY_MSG15
               00000000'EF    DD  0C50                        PUSHL   LUN
                         01   DD  0C56                        PUSHL   #1
          00000000'9F    04   FB  0C58                        CALLS   #$$M, a#DS$ERRHARD
                                  0C5F  3049 20$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF22 CF    FA  0C5F                        CALLG   10$, a#DS$CKLOOP
                                  0C68  3050 $DS_ENDSUB
                                  0C68       T12_S14_X:
00000000'9F   000002A0'EF    FA  0C68                        CALLG   $S$, a#DS$ENDSUB
```

```
                                  0C73  3052 $DS_ENDTEST
                  50   01   D0    0C73                  MOVL    #1, R0          ; NORMAL EXIT
                                  0C76       TEST_012_X::
        00000000'9F    6E   FA    0C76                  CALLG   (SP), @#DS$BREAK
                             04   0C7D                  RET                     ; RETURN TO TEST SEQUENCER
                                  0C7E  3053 $DS_PAGE
                                  0C7E             .SBTTL  TEST 13:  BLOCK TRANSFER TEST
                             00000000                  .PSECT  TEST_013, PAGE, NOWRT
                                  0018
                                  0018  3055 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  0018       DATA_013:
                        00000000  0018                  .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  001C       TEST_013::
                            0000  001C                  .WORD    M<>            ; ENTRY MASK
                                  001E  3056 ;++
                                  001E  3057 ; TEST DESCRIPTION:
                                  001E  3058 ;
                                  001E  3059 ;       THIS TEST VERIFIES THAT THE RH750 TRANSFERS DATA
                                  001E  3060 ;       FROM THE MASS BUS TO MEMORY WITHOUT ERRORS.
                                  001E  3061 ;       IT ALSO VERIFIES THAT THE RH750 TRANSFERS DATA ACROSS
                                  001E  3062 ;       PAGE BOUNDARIES CORRECTLY.
                                  001E  3063 ;
                                  001E  3064 ; TEST ALGORITHM:
                                  001E  3065 ;
                                  001E  3066 ;       WRITE AN ALL 1'S PATTERN ONTO THE MASS BUS--DONE IN MAINTENANCE MODE
                                  001E  3067 ;       READ THAT PATTERN INTO THE FIRST AND LAST LONG WORDS OF A
                                  001E  3068 ;       1 PAGE BUFFER. THIS IS DONE BY SETTING ALL MAP REGISTERS
                                  001E  3069 ;       ENTRIES TO POINT TO THE PAGE FRAME (PHYSICAL MEMORY) OF THE
                                  001E  3070 ;       BUFFER. THE VIRTUAL ADDRESS IS THEN SET TO POINT TO THE LAST
                                  001E  3071 ;       LONG WORD BOUNDARY IN THE PAGE, THE TRANSFER IS EXECUTED CAUSING
                                  001E  3072 ;       THE VIRTUAL ADDRESS REGISTER TO SELECT THE NEXT MAP REGISTER
                                  001E  3073 ;       AND TRANSFER FOUR BYTES INTO THE FIRST LONG WORD OF THE BUFFER.
                                  001E  3074 ;
                                  001E  3075 ; NOTE: DATA COMPARE ERRORS OCCUR IF THE DATA IS NOT SUCCESSFULLY
                                  001E  3076 ;       TRANSFERRED TO THE APPROPRIATE LOCATIONS.
                                  001E  3077 ;
                                  001E  3078 ; ERROR INDICATORS:
                                  001E  3079 ;
                                  001E  3080 ; THE PHYSICAL MEMORY LOCATION OF THE ERROR
                                  001E  3081 ; THE MAP REGISTER NUMBER THAT WAS INVOKED
                                  001E  3082 ; THE CONTENTS OF THE VIRTUAL ADDRESS REGISTER
                                  001E  3083 ; THE CONTENTS OF THE BYTE COUNT REGISTER
                                  001E  3084 ; THE CONTENTS OF THE MAP REGISTER POINTED AT BY THE VIRTUAL ADDRESS
                                  001E  3085 ;--
                                  001E  3086 $DS_BGNSUB
                                  001E       T13_S1::
00000000'9F   000002AC'EF   FA    001E                  CALLG   $S$, @#DS$BGNSUB
                        FFD4'  30  0029  3087         BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0    002C  3088         MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS TO R2
     0000'C2   00000000'8F   C8    0033  3089         BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
          56   00000000'EF   DE    003C  3090         MOVAL   MIOBUFFER,R6            ; MOVE ADDRESS OF BUFFER TO R6
           57   56   00'8F   78    0043  3091         ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
          57   00000000'8F   C8    0048  3092         BISL    #VALID_BIT,R7           ; SET V BIT IN MAP ENTRY
                         50   D4    004F  3093         CLRL    R0                      ; CLEAR MAP SELECTOR INDEX
     5A   52   00000000'8F   C1    0051  3094         ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
                  6A40   57   D0    0059  3095 10$:    MOVL    R7,(R10) R0             ; WRITE MAP REGISTER R0
       F4 50   000000FF 8F   F3    005D  3096         AOBLEQ  #255,R0,10$             ; DO FOR ALL MAP REGISTERS
```

```
                  0000'C2   D4  0065  3097 20$:    CLRL    VAR(R2)                  ; CLEAR VIRTUAL ADDRESS REGISTER
         00000000'EF   00   D2  0069  3098         MCOML   #0,MIOBUFFER             ; WRITE 1'S PATTERN INTO FIRST
         00000004'EF   00   D2  0070  3099         MCOML   #0,MIOBUFFER+4           ; TWO LONGWORDS OF BUFFER
             0000'C2   06   CE  0077  3100         MNEGL   #6,BCR(R2)               ; SET BYTE COUNT = 6
    0000'C2   00000000'8F   C8  007C  3101         BISL    #MAINT_MODE,CR(R2)       ; PUT RH INTO MAINTENANCE MODE
         50   00000000'EF   D0  0085  3102         MOVL    DRIVE0,R0                ; GET PSEUDO DRIVE ADDRESS
         60   00000000'8F   D0  008C  3103         MOVL    #WRITE,(R0)              ; ISSUE WRITE COMMAND TO DRIVE
    0000'C2   00000000'8F   D0  0093  3104         MOVL    #SIM_OCC,DR(R2)          ; SET OCCUPIED
    0000'C2   00000000'8F   C8  009C  3105         BISL    #SIM_SCLK,DR(R2)         ; ASSERT SCLK
    0000'C2   00000000'8F   CA  00A5  3106         BICL    #SIM_SCLK,DR(R2)         ; DEASSERT SCLK
    0000'C2   00000000'8F   C8  00AE  3107         BISL    #DT_ABORT,CR(R2)         ; SET ABORT
    0000'C2   00000000'8F   C8  00B7  3108         BISL    #SIM_EBL,DR(R2)          ; ASSERT EBL END OF TRANSFER
    0000'C2   00000000'8F   CA  00C0  3109         BICL    #SIM_EBL,DR(R2)          ; DEASSERT EBL
                                00C9  3110         ERRPREP RHSR_MSG,2,FMT_STATUS_REG,UNEXPECTED ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE  00C9                       MOVAL   RHSR_MSG,REG_NAME
         00000000'EF   02   9A  00D4                       MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  00DB                       MOVAL   FMT_STATUS_REG,REG_STRING
              00000000'8F   D0  00E6  3111         MOVL    #DATA_XFER_DONE!-        ; SET UP EXPECTED STATUS
                        54      00EC  3112                 DATA_XFER_ABRT,R4
         54   00000000'EF   C8  00ED  3113         BISL    RH11TB_MSK,R4            ; LOAD EXPECTED MASK BIT
             53   0000'C2   D0  00F4  3114         MOVL    SR(R2),R3                ; READ STATUS REGISTER
                  54   53   D1  00F9  3115         CMPL    R3,R4                    ; IS STATUS AS EXPECTED
                        1B   13  00FC  3116         BEQL    30$                      ; BRANCH IF STATUS IS OK
                                00FE  3117         $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                00FE  3118                         ARRAY_MSG62,-
                                00FE  3119                         PRINT_SBE
              00000000'EF   DF  00FE                       PUSHAL  PRINT_SBE
              00000000'EF   DF  0104                       PUSHAL  ARRAY_MSG62
              00000000'EF   DD  010A                       PUSHL   LUN
                        01   DD  0110                       PUSHL   #1
         00000000'9F   04   FB  0112                       CALLS   #$$M, a#DS$ERRHARD
                                0119  3120 30$:    $DS_CKLOOP      20$              ; SCOPE LOOP
    00000000'9F   FF48 CF   FA  0119                       CALLG   20$, a#DS$CKLOOP
                                0122  3121 $DS_ENDSUB
                                0122       T13_S1_X:
00000000'9F   000002AC'EF   FA  0122                       CALLG   $S$, a#DS$ENDSUB
```

```
                                012D  3123          $DS_BGNSUB
                                012D        T13_S2::
00000000'9F   000002B8'EF   FA  012D                CALLG   $S$, a#DS$BGNSUB
                                0138  3124 ;++
                                0138  3125 ; BEGIN READING ACROSS PAGE BOUNDARIES
                                0138  3126 ;--
    0000'C2   00000000'8F   CA  0138  3127          BICL    #DT_ABORT,CR(R2)        ; CLEAR ABORT
                       58   D4  0141  3128 10$:     CLRL    R8                      ; CLEAR PAGE FRAME POINTER
         5A   00000000'EF   DE  0143  3129          MOVAL   MIOBUFFER,R10           ; GET ADDR OF BUFFER INTO R10
                       6A   D4  014A  3130 15$:     CLRL    (R10)                   ; CLEAR FIRST LONG WORD OF BUF
                  01FC CA   D4  014C  3131          CLRL     X1FC(R10)              ; CLEAR LAST LONG WORD OF BUF
         59   000001FF 8F   D0  0150  3132          MOVL    # X1FF,R9               ; MOVE PAGE BYTE POINTER TO R9
          50    58   00'8F   9C  0157  3133          ROTL    #PF_FIELD,R8,R0         ; PUT INTO PFN FIELD
               59   50   C8  015C  3134          BISL    R0,R9                   ; SET MAP POINTER IN VAR
           0000'C2   59   D0  015F  3135          MOVL    R9,VAR(R2)              ; WRITE VIRTUAL ADDR INTO VAR
           0000'C2   02   CE  0164  3136          MNEGL   #2,BCR(R2)              ; BYTE COUNT = 2
            00000000'8F   DD  0169  3137          PUSHL   #READ                   ; PUSH READ CMD FOR MM_XFER
                      02   DD  016F  3138          PUSHL   #2                      ; PUSH BYTE COUNT
                      52   DD  0171  3139          PUSHL   R2                      ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  0173  3140          CALLS   #3,MM_XFER              ; INITIATE 2 BYTE READ XFER
                                017A  3141          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                017A  3142                  FMT_STATUS_REG,-
                                017A  3143                  UNEXPECTED
00000000'EF   00000000'EF   DE  017A                MOVAL   RHSR_MSG,REG_NAME
         00000000'EF   02   9A  0185                MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  018C                MOVAL   FMT_STATUS_REG,REG_STRING
              FFFFFFFF'8F   CB  0197  3144          BICL3   # CDATA_XFER_DONE,-     ; CHECK FOR DATA TRANSFER DONE
           53   0000'C2        019D  3145                  SR(R2),R3
                      22   12  01A1  3146          BNEQ    20$                     ; IF STATUS IS OK THEN BRANCH
      54   00000000'8F   D0  01A3  3147          MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
                                01AA  3148          $DS_ERRHARD_S   #1,LUN,-
                                01AA  3149                  ARRAY_MSG62,-
                                01AA  3150                  PRINT_SBE
           00000000'EF   DF  01AA                       PUSHAL  PRINT_SBE
           00000000'EF   DF  01B0                       PUSHAL  ARRAY_MSG62
           00000000'EF   DD  01B6                       PUSHL   LUN
                      01   DD  01BC                       PUSHL   #1
      00000000'9F   04   FB  01BE                       CALLS   #$$M, a#DS$ERRHARD
                                01C5  3151 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
00000000'9F   FF78 CF   FA  01C5                       CALLG   10$, a#DS$CKLOOP
              53   6A   D0  01CE  3152          MOVL    (R10),R3                ; READ FIRST LONG WORD OF BUFFER
      53   000000FF 8F   D1  01D1  3153          CMPL    # XFF,R3               ; IS LONG WORD CORRECT?
                      30   13  01D8  3154          BEQL    30$                     ; BRANCH IF NO ERRORS
            54   FF 8F   9A  01DA  3155          MOVZBL  # XFF,R4                ; SET UP EXPECTED RESULTS REGISTER
                                01DE  3156          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                01DE  3157                  ARRAY_MSG1
                      00   DD  01DE                       PUSHL   #0
           00000000'EF   DF  01E0                       PUSHAL  ARRAY_MSG1
           00000000'EF   DD  01E6                       PUSHL   LUN
                      02   DD  01EC                       PUSHL   #2
      00000000'9F   04   FB  01EE                       CALLS   #$$M, a#DS$ERRHARD
                                01F5  3158          $DS_PRINTB_S    FMT_DTERR,R4,-  ; PRINT PARAMETERS
                                01F5  3159                  R3,R8,R9
                      59   DD  01F5                       PUSHL   R9
                      58   DD  01F7                       PUSHL   R8
                      53   DD  01F9                       PUSHL   R3
                      54   DD  01FB                       PUSHL   R4
```

```
                00000000'EF   9F  01FD                          PUSHAB  FMT_DTERR
          00000000'9F   05   FB  0203                          CALLS   #$$N, a#DS$PRINTB
                                 020A  3160 30$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
     00000000'9F   FF33 CF   FA  020A                          CALLG   10$, a#DS$CKLOOP
              53   01FC CA   D0  0213  3161         MOVL     X1FC(R10),R3               ; READ LAST LONG WORD  OF BUFFER
          54   00000000'8F   D0  0218  3162         MOVL     #BYTE3,R4                  ; SET UP EXPECTED VALUE
                    54   53   D1  021F  3163         CMPL     R3,R4                      ; ANY MISSING BITS
                         2C   13  0222  3164         BEQL     40$                        ; BRANCH IF NO ERRORS
                                 0224  3165         $DS_ERRHARD_S   #3,LUN,-            ; PRINT ERROR
                                 0224  3166                         ARRAY_MSG1
                         00   DD  0224                          PUSHL   #0
                00000000'EF   DF  0226                          PUSHAL  ARRAY_MSG1
                00000000'EF   DD  022C                          PUSHL   LUN
                         03   DD  0232                          PUSHL   #3
          00000000'9F   04   FB  0234                          CALLS   #$$M, a#DS$ERRHARD
                                 023B  3167         $DS_PRINTB_S    FMT_DTERR,R4,-      ; AND PARAMETERS
                                 023B  3168                         R3,R8,R9
                         59   DD  023B                          PUSHL   R9
                         58   DD  023D                          PUSHL   R8
                         53   DD  023F                          PUSHL   R3
                         54   DD  0241                          PUSHL   R4
                00000000'EF   9F  0243                          PUSHAB  FMT_DTERR
          00000000'9F   05   FB  0249                          CALLS   #$$N, a#DS$PRINTB
                                 0250  3169 40$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
     00000000'9F   FEED CF   FA  0250                          CALLG   10$, a#DS$CKLOOP
                         58   D6  0259  3170         INCL     R8                         ; INCREMENT MAP SELECTOR
                    58   0A   D1  025B  3171         CMPL     #10,R8                     ; DONE TEN TIMES YET?
                         08   12  025E  3172         BNEQ     45$                        ; IF NOT, BRANCH
                                 0260  3173         $DS_BQUICK      50$                 ; IF QUICK VERIFY, EXIT
                         08   E0  0260                          BBS     #DSA$V_QUICK,-      ; BR IF QUICK
            0D 0000FE00 9F       0262                                 a#DSA$GL_FLAGS, 50$
          58   00000100 8F   D1  0268  3174 45$:    CMPL     #256,R8                    ; HAVE ALL MAP REGS BEEN SELECTED?
                         04   13  026F  3175         BEQL     50$                        ; EXIT IF ALL MAPS HAVE BEEN USED
                    FED5 CF   17  0271  3176         JMP      15$                        ; LOOP USING NEXT MAP REGISTER
                                 0275  3177 50$:
                                 0275  3178         $DS_ENDSUB
                                 0275       T13_S2_X:
  00000000'9F   000002B8'EF   FA  0275                          CALLG   $S$, a#DS$ENDSUB
```

```
                                  0280  3180          $DS_ENDTEST
                  50   01   D0   0280                        MOVL    #1, R0          ; NORMAL EXIT
                                  0283        TEST_013_X::
        00000000'9F   6E   FA   0283                        CALLG   (SP), a#DS$BREAK
                           04   028A                        RET                     ; RETURN TO TEST SEQUENCER
                                  028B  3181 $DS_PAGE
                                  028B                .SBTTL  TEST 14:  BYTE COUNTER TEST
                           00000000                        .PSECT  TEST_014, PAGE, NOWRT
                                  0014
                                  0014  3183 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                                  0014        DATA_014:
                       00000000   0014                        .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                  0018        TEST_014::
                           0000   0018                        .WORD    M<>            ; ENTRY MASK
                                  001A  3184 ;++
                                  001A  3185 ; TEST DESCRIPTION:
                                  001A  3186 ;
                                  001A  3187 ;       THIS TEST VERIFIES THE BYTE COUNTER FUNCTIONALITY
                                  001A  3188 ;
                                  001A  3189 ; TEST ALGORTITHM:
                                  001A  3190 ;
                                  001A  3191 ;       CHECK FOR STUCK AT ONE FAULTS BY FLOATING A ONE THROUGH
                                  001A  3192 ;       EACH BCR BIT POSITION.
                                  001A  3193 ;       ISSUES WRITE TO DRIVE COMMANDS WITH VARYING BYTE COUNTS
                                  001A  3194 ;       AND VERIFYING THAT THE LOWER HALF DECREMENTS FOR EACH
                                  001A  3195 ;       BYTE CLOCKED INTO THE SILO.
                                  001A  3196 ;       THE UPPER HALF OF THE BYTE COUNTER REGISTER IS TESTED
                                  001A  3197 ;       BY ISSUEING A READ FROM DRIVE COMMAND AND VERIFYING THAT
                                  001A  3198 ;       EACH BYTE CLOCKED INTO THE SILO FROM THE MASS BUS
                                  001A  3199 ;       DECREMENTS THE BCR.
                                  001A  3200 ;--
                                  001A  3201 $DS_BGNSUB
                                  001A        T14_S1::
00000000'9F   000002C4'EF   FA   001A                        CALLG   $S$, a#DS$BGNSUB
                      FFD8'   30   0025  3202          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0   0028  3203          MOVL    RH_CUR_ADR,R2           ; MOVE ADDRESS OF RH750 TO R2
                                  002F  3204          ERRPREP RHBCR_MSG,3,FMT_ANY_REG,SA0_MSG ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE   002F                        MOVAL   RHBCR_MSG,REG_NAME
        00000000'EF   03   9A   003A                        MOVZBL  #3,REG_NO
00000000'EF   00000000'EF   DE   0041                        MOVAL   FMT_ANY_REG,REG_STRING
    0000'C2   00000000'8F   C8   004C  3205          BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                  54   01   D0   0055  3206          MOVL    #1,R4                   ; INITIALIZE DATA PATTERN
          54   00010000 8F   C8   0058  3207          BISL    #BIT16,R4               ; SET UPPER HALF TO REFLECT LOWER HALF
                       57   D4   005F  3208          CLRL    R7                      ; INITIALIZE SHIFT COUNT
             0000'C2   54   D0   0061  3209 10$:     MOVL    R4,BCR(R2)              ; WRITE PATTERN INTO BCR
             53   0000'C2   D0   0066  3210          MOVL    BCR(R2),R3              ; READ BCR
                  54   53   D1   006B  3211          CMPL    R3,R4                   ; IS BCR VALUE CORRECT?
                       1B   13   006E  3212          BEQL    20$                     ; BRANCH IF NO ERRORS
                                  0070  3213          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                  0070  3214                          ARRAY_MSG14,-
                                  0070  3215                          PRINT_SBE
             00000000'EF   DF   0070                        PUSHAL  PRINT_SBE
             00000000'EF   DF   0076                        PUSHAL  ARRAY_MSG14
             00000000'EF   DD   007C                        PUSHL   LUN
                       01   DD   0082                        PUSHL   #1
        00000000'9F   04   FB   0084                        CALLS   #$$M, a#DS$ERRHARD
                                  008B  3216 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
```

```
                                                                                CALLG   10$, a#DS$CKLOOP
          00000000'9F    D3 AF    FA  008B
                  54    54   01    9C  0093  3217          ROTL    #1,R4,R4                ; SHIFT PATTERN TO NEXT LEFT POSITION
                    C6 57    0F    F3  0097  3218          AOBLEQ  #15,R7,10$              ; INDEX AND LOOP FOR ALL BITS
        5A    52    00000000'8F    C1  009B  3219          ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
                                 50    D4  00A3  3220          CLRL    R0                      ; CLEAR MAP INDEX
              57    00000000'EF    DE  00A5  3221          MOVAL   DATA_IBUFFER,R7         ; PREPARE TO CREATE MAP ENTRIES
               59    57    00'8F    78  00AC  3222          ASHL    #-PF_FIELD,R7,R9        ; PUT INTO PFN FIELD
              59    00000000'8F    C8  00B1  3223          BISL    #VALID_BIT,R9           ; VALIDATE THE MAP ENTRY
                      6A40    59    D0  00B8  3224          MOVL    R9,(R10) R0             ; WRITE MAP REGISTER 0
                                 50    D6  00BC  3225          INCL    R0                      ; INCREMENT MAP INDEX
              57    00000000'EF    DE  00BE  3226          MOVAL   MIOBUFFER,R7            ; GET ADDRESS OF MIDDLE BUFFER
               59    57    00'8F    78  00C5  3227          ASHL    #-PF_FIELD,R7,R9        ; PUT INTO PFN FIELD
              59    00000000'8F    C8  00CA  3228          BISL    #VALID_BIT,R9           ; V IT
                      6A40    59    D0  00D1  3229          MOVL    R9,(R10) R0             ; WRITE MAP REG 1
              57    00000000'EF    DE  00D5  3230          MOVAL   DATA_OBUFFER,R7         ; GET ADDRESS OF OUT PUT BUFFER
               59    57    00'8F    78  00DC  3231          ASHL    #-PF_FIELD,R7,R9        ; PUT INTO PFN
                                 50    D6  00E1  3232          INCL    R0                      ; INCREMENT MAP INDEX
              59    00000000'8F    C8  00E3  3233          BISL    #VALID_BIT,R9           ; V IT
                      6A40    59    D0  00EA  3234          MOVL    R9,(R10) R0             ; WRITE MAP REG 2
                                       00EE  3235
                                       00EE  3236 $DS_ENDSUB
                                       00EE       T14_S1_X:
    00000000'9F    000002C4'EF    FA  00EE                   CALLG   $S$, a#DS$ENDSUB
```

```
                                00F9  3238          $DS_BGNSUB                          ; V IT
                                00F9          T14_S2::
00000000'9F   000002D0'EF   FA  00F9                      CALLG   $S$, a#DS$BGNSUB
                 58   01    D0  0104  3239          MOVL    #1,R8                       ; INITIALIZE BYTE COUNT
     0000'C2   00000000'8F   C8  0107  3240 10$:     BISL    #PGM_INIT,CR(R2)            ; INIT THE MUT
     0000'C2   00000000'8F   C8  0110  3241          BISL    #MAINT_MODE,CR(R2)          ; PUT MUT INTO MAINTENANCE MODE
               0000'C2       D4  0119  3242          CLRL    VAR(R2)                     ; CLEAR VIRTUAL ADDRESS REG
          0000'C2   58       CE  011D  3243          MNEGL   R8,BCR(R2)                  ; WRITE BYTE COUNT INTO BCR
          50   00000000'EF   D0  0122  3244          MOVL    DRIVE0,R0                   ; GET DRIVE ADDRESS
          60   00000000'8F   D0  0129  3245          MOVL    #WRITE,(R0)                 ; ISSUE WRITE CMD
     0000'C2   00000000'8F   D0  0130  3246          MOVL    #SIM_OCC,DR(R2)             ; SET OCCUPIED--PREVENT MISSED XFER
                             0139  3247          $DS_WAITUS_S    #10                     ; WAIT 100 US
                          00 DD  0139                      PUSHL   #0
                          0A DD  013B                      PUSHL   #10
          00000000'9F     02 FB  013D                      CALLS   #2, a#DS$WAITUS
             53   0000'C2   D0  0144  3248          MOVL    BCR(R2),R3                  ; READ BCR
          53   00000000'8F   CA  0149  3249          BICL    #BYTE3!BYTE2,R3             ; CLEAR UPPER WORD OF BCR COPY
                       1D   13  0150  3250          BEQL    20$                         ; BCR SHOULD BE ZERO IF BYTE COUNT WAS
                             0152  3251                                                  ; LESS THAN 35
                       54   D4  0152  3252          CLRL    R4                          ; EXPECT ZERO
                             0154  3253          $DS_ERRHARD_S #1,LUN,-                  ; PRINT ERROR
                             0154  3254                        ARRAY_MSG24,-
                             0154  3255                        PRINT_SBE
               00000000'EF   DF  0154                      PUSHAL  PRINT_SBE
               00000000'EF   DF  015A                      PUSHAL  ARRAY_MSG24
               00000000'EF   DD  0160                      PUSHL   LUN
                       01    DD  0166                      PUSHL   #1
          00000000'9F     04 FB  0168                      CALLS   #$$M, a#DS$ERRHARD
                             016F  3256 20$:     $DS_CKLOOP      10$                     ; SCOPE LOOP?
     00000000'9F   95 AF   FA  016F                      CALLG   10$, a#DS$CKLOOP
           8C 58   22      F3  0177  3257          AOBLEQ  #34,R8,10$                  ; THE SILO IS 34 BYTES DEEP AND FOR A
                             017B  3258                                                  ; WRITE COMMAND THAT MANY BYTES SHOULD
                             017B  3259                                                  ; BE CLOCKED INTO THE SILO
                             017B  3260          $DS_ENDSUB
                             017B          T14_S2_X:
00000000'9F   000002D0'EF   FA  017B                      CALLG   $S$, a#DS$ENDSUB
```

```
                                0186  3262          $DS_BGNSUB
                                0186          T14_S3::
00000000'9F   000002DC'EF   FA  0186                CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0191  3263          BISL    #PGM_INIT,CR(R2)          ; INIT THE MUT
          50  00000000'EF   DE  019A  3264          MOVAL   MIOBUFFER,R0              ; MOVE BUFFER ADDRESS TO R0
           50   50    F7 8F   78  01A1  3265          ASHL    #-9,R0,R0                 ; PUT PAGE FRAME NUMBER INTO R0
          50   00000000'8F   C8  01A6  3266          BISL    #VALID_BIT,R0             ; CREATE VALID MAP ENTRY
                         51   D4  01AD  3267          CLRL    R1                        ; CLEAR MAP SELECTOR INDEX
            0800 C241    50   D0  01AF  3268 10$:     MOVL    R0, X800(R2) R1           ; WRITE ALL MAPS THEY NOW ALL POINT
                                01B5  3269                                            ; TO BUFFER'S PAGE FRAME
       F2 51    000000FF 8F   F3  01B5  3270          AOBLEQ  #255,R1,10$               ; IF ALL MAPS NOT WRITTEN LOOP
                                01BD  3271          ERRPREP RHBCR_MSG,4,FMT_ANY_REG ; PREPARE TO HANDLE ERROR
00000000'EF   00000000'EF   DE  01BD                MOVAL   RHBCR_MSG,REG_NAME
         00000000'EF   04   9A  01C8                MOVZBL  #4,REG_NO
00000000'EF   00000000'EF   DE  01CF                MOVAL   FMT_ANY_REG,REG_STRING
                   58   24   9A  01DA  3272 20$:     MOVZBL  #36,R8                    ; SET UP INITIAL VALUE OF BCR AFTER
                                01DD  3273                                            ; WRITE COMMAND ISSUED
          59   00003000'EF   D0  01DD  3274          MOVL    DRIVE0,R9                 ; PUT ADDRESS OF DRIVE 0 INTO R9
    0000'C2   00000000'8F   C8  01E4  3275          BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8  01ED  3276          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
                   0000'C2  D4  01F6  3277          CLRL    VAR(R2)                   ; CLEAR VAR
                   0000'C2  D4  01FA  3278          CLRL    BCR(R2)                   ; SET BCR TO MAXIMUM COUNT
          69   00000000'8F   D0  01FE  3279          MOVL    #WRITE,(R9)               ; ISSUE WRITE COMMAND TO DUMMY DRIVE
    0000'C2   00000000'8F   D0  0205  3280          MOVL    #SIM_OCC,DR(R2)           ; ASSERT OCCUPIED
                                020E  3281          $DS_WAITUS_S    #10               ; WAIT 100 US
                         00   DD  020E                PUSHL   #0
                         0A   DD  0210                PUSHL   #10
         00000000'9F     02   FB  0212                CALLS   #2, a#DS$WAITUS
                         50   D4  0219  3282 30$:     CLRL    R0                        ; CLEAR SCLK COUNTER
0000'C2   00000000'8F   C8  021B  3283 35$:     BISL    #SIM_SCLK,DR(R2)          ; SET SCLK
0000'C2   00000000'8F   CA  0224  3284          BICL    #SIM_SCLK,DR(R2)          ;
           EA 50        03   F3  022D  3285          AOBLEQ  #3,R0,35$                 ; DO FOR 4 SCLKS
              58        08   C0  0231  3286          ADDL    #8,R8                     ; BYTE CTR SHOULD HAVE INCREASED BY 8
              58        04   B1  0234  3287          CMPW    #4,R8                     ; DONE YET?
                        02   12  0237  3288          BNEQ    36$                       ; BRANCH IF NO
                        58   D4  0239  3289          CLRL    R8                        ; BYTE COUNTER STOPS AT 0
                 54     58   3C  023B  3290 36$:     MOVZWL  R8,R4                     ; SET UP EXPECTED BYTE COUNT
                        50   D4  023E  3291          CLRL    R0                        ; CLEAR DELAY COUNTER
             FC 50      0A   F3  0240  3292 37$:     AOBLEQ  #10,R0,37$                ; WAIT FOR DATA TO XFER FROM MEMORY
           51   0000'C2     D0  0244  3293          MOVL    BCR(R2),R1                ; READ BYTE COUNTER
                53     51   3C  0249  3294          MOVZWL  R1,R3                     ; CLEAR MASS BUS BYTE COUNT
                54     53   D1  024C  3295          CMPL    R3,R4                     ; IS CMI BYTE COUNT ACCURATE?
                       1B   13  024F  3296          BEQL    40$                       ; BRANCH IF BYTE COUNT IS OK
                                0251  3297          $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                0251  3298                          ARRAY_MSG24,-
                                0251  3299                          PRINT_SBE
             00000000'EF   DF   0251                PUSHAL  PRINT_SBE
             00000000'EF   DF   0257                PUSHAL  ARRAY_MSG24
             00000000'EF   DD   025D                PUSHL   LUN
                           01   DD  0263                PUSHL   #1
         00000000'9F       04   FB  0265                CALLS   #$$M, a#DS$ERRHARD
                                026C  3300 40$:     $DS_CKLOOP      20$               ; SCOPE LOOP?
00000000'9F   FF6A CF   FA  026C                CALLG   20$, a#DS$CKLOOP
              58   00   B1  0275  3301          CMPW    #0,R8                     ; CHECK FOR ZERO BYTE COUNT
                   9F   12  0278  3302          BNEQ    30$                       ; CONTINUE TRANSFER IF BYTE COUNT IS
                                027A  3303                                            ; NOT ZERO
0000'C2   00000000'8F   C8  027A  3304          BISL    #SIM_EBL,DR(R2)           ; SET END OF BLOCK
```

```
        0000'C2   00000000'8F   CA  0283  3305          BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                    028C  3306          $DS_ENDSUB
                                    028C        T14_S3_X:
    00000000'9F   000002DC'EF   FA  028C                CALLG   $S$, a#DS$ENDSUB
```

```
                                        0297  3308          $DS_BGNSUB
                                        0297          T14_S4::
00000000'9F   000002E8'EF    FA         0297                        CALLG   $S$, a#DS$BGNSUB
                        58   D4         02A2  3309          CLRL    R8                      ; CLEAR MASS BUS BYTE COUNT
     0000'C2   00000000'8F   C8         02A4  3310 10$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE THE RH
     0000'C2   00000000'8F   C8         02AD  3311          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINT_MODE
                        58   D4         02B6  3312          CLRL    R8                      ; CLEAR MASS BUS BYTE COUNT
                  0000'C2   D4         02B8  3313          CLRL    VAR(R2)                 ; CLEAR THE VIRTUAL ADDRESS REGISTER
                  0000'C2   D4         02BC  3314          CLRL    BCR(R2)                 ; CLEAR BYTE COUNTER
           69   00000000'8F   D0         02C0  3315          MOVL    #WRITE,(R9)             ; ISSUE WRITE COMMAND
     0000'C2   00000000'8F   D0         02C7  3316          MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                        02D0  3317          $DS_WAITUS_S    #10             ; WAIT FOR SILO TO FILL
                        00   DD         02D0                        PUSHL   #0
                        0A   DD         02D2                        PUSHL   #10
           00000000'9F   02   FB         02D4                        CALLS   #2, a#DS$WAITUS
     0000'C2   00000000'8F   C8         02DB  3318 20$:     BISL    #SIM_SCLK,DR(R2)        ; SET SCLK
     0000'C2   00000000'8F   CA         02E4  3319          BICL    #SIM_SCLK,DR(R2)        ; DE-ASSERT SCLK
                        58   02   A0    02ED  3320          ADDW    #2,R8                   ; INCREASE EXPECTED MB BYTE COUNT
                  53   0000'C2   D0     02F0  3321          MOVL    BCR(R2),R3              ; READ BCR
              58  53   10   10   ED     02F5  3322          CMPZV   #16,#16,R3,R8           ; COMPARE MB BYTE COUNTS
                        25   13         02FA  3323          BEQL    30$                     ; BRANCH IF BYTE COUNT IS CORRECT
                  54   53   3C         02FC  3324          MOVZWL  R3,R4                   ; MOVE SBI BYTE CNT TO EXP VALUE REG
                  50   58   10   9C     02FF  3325          ROTL    #16,R8,R0               ; SET UP EXPECTED VALUE FOR ER DISPLAY
                  54   50   C8         0303  3326          BISL    R0,R4                   ; SET UP EXPECTED BCR VALUE
                                        0306  3327          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                        0306  3328                          ARRAY_MSG24,-
                                        0306  3329                          PRINT_SBE
                  00000000'EF   DF      0306                        PUSHAL  PRINT_SBE
                  00000000'EF   DF      030C                        PUSHAL  ARRAY_MSG24
                  00000000'EF   DD      0312                        PUSHL   LUN
                        01   DD         0318                        PUSHL   #1
              00000000'9F   04   FB     031A                        CALLS   #$$M, a#DS$ERRHARD
                                        0321  3330 30$:     $DS_CKLOOP 10$                  ; SCOPE LOOP?
           00000000'9F   80  AF   FA     0321                        CALLG   10$, a#DS$CKLOOP
                        58   B5         0329  3331          TSTW    R8                      ; SEE IF TRANSFER HAS COMPLETED
                        AE   12         032B  3332          BNEQ    20$                     ; IF TRANSFER IS NOT COMPLETED LOOP
     0000'C2   00000000'8F   C8         032D  3333          BISL    #SIM_EBL,DR(R2)         ; SET END OF BLOCK
     0000'C2   00000000'8F   CA         0336  3334          BICL    #SIM_EBL,DR(R2)         ; CLEAR END OF BLOCK
                                        033F  3335          $DS_ENDSUB
                                        033F          T14_S4_X:
00000000'9F   000002E8'EF    FA         033F                        CALLG   $S$, a#DS$ENDSUB
```

```
                                034A  3337 $DS_ENDTEST
                  50   01   D0   034A                  MOVL    #1, R0          ; NORMAL EXIT
                                034D        TEST_014_X::
        00000000'9F    6E   FA   034D                  CALLG   (SP), a#DS$BREAK
                             04   0354                  RET                     ; RETURN TO TEST SEQUENCER
                                0355  3338 $DS_PAGE
                                0355              .SBTTL  TEST 15:  FUNCTIONAL READ/WRITE TESTS
                             00000000                  .PSECT  TEST_015, PAGE, NOWRT
                                0020
                                0020  3340 $DS_BGNTEST    <DEFAULT,ALL>,,LONG
                                0020        DATA_015:
                       00000000 0020                  .LONG   0               ; TEST ARGUMENT TABLE TERMINATOR
                                0024        TEST_015::
                           0000 0024                  .WORD    M<>            ; ENTRY MASK
                                0026  3341 ;++
                                0026  3342 ; TEST DESCRIPTION:
                                0026  3343 ;
                                0026  3344 ;       THIS TEST VERIFIES THAT THE RH750 CAN EXECUTE PAGED TRANSFERS
                                0026  3345 ;       ALONG WITH THE PROPER EXECUTION OF READ,READ REVERSE,
                                0026  3346 ;       WRITECHECK, WRITECHECK REVERSE.
                                0026  3347 ;
                                0026  3348 ; TEST ALGORITHM:
                                0026  3349 ;
                                0026  3350 ;       SET UP A 1 PAGE BUFFER WITH AN ALL ONE'S PATTERN AND
                                0026  3351 ;       WRITE THAT PATTERN ONTO THE MASS BUSS. THEN EXECUTE PAGED
                                0026  3352 ;       TRANSFERS USING ALL MAP REGISTERS FOR EACH OF THE FOLLOWING
                                0026  3353 ;       DRIVE COMMANDS: READ,WRITE CHECK,READ REVERSE,WRITE CHECK
                                0026  3354 ;       REVERSE.
                                0026  3355 ;--
                                0026  3356 $DS_BGNSUB
                                0026        T15_S1::
00000000'9F    000002F4'EF   FA   0026                  CALLG   $S$, a#DS$BGNSUB
                       FFCC'  30   0031  3357         BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0   0034  3358         MOVL    RH_CUR_ADR,R2           ; MOVE ADDRESS OF RH750 INTO R2
                                003B  3359         ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                003B  3360                 FMT_STATUS_REG,-
                                003B  3361                 UNEXPECTED
00000000'EF    00000000'EF   DE   003B                  MOVAL   RHSR_MSG,REG_NAME
        00000000'EF    02   9A   0046                  MOVZBL  #2,REG_NO
00000000'EF    00000000'EF   DE   004D                  MOVAL   FMT_STATUS_REG,REG_STRING
    0000'C2    00000000'8F   C8   0058  3362 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT RH
    0000'C2    00000000'8F   C8   0061  3363         BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
          57   00000000'EF   DE   006A  3364         MOVAL   MIOBUFFER,R7            ; PICK UP  ADDRESS OF DATA BUFFER
                    55   57   D0   0071  3365         MOVL    R7,R5                   ; MAKE COPY OF DATA BUFFER ADDRESS
           58   57   00'8F   78   0074  3366         ASHL    #-PF_FIELD,R7,R8        ; PUT INTO PFN FIELD
          58   00000000'8F   C8   0079  3367         BISL    #VALID_BIT,R8           ; SET V BIT IN MAP ENTRY
                         59   D4   0080  3368         CLRL    R9                      ; CLEAR MAP INDEX
    5A    52   00000000'8F   C1   0082  3369         ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
                  6A49   58   D0   008A  3370 20$:    MOVL    R8,(R10) R9             ; WRITE A MAP ENTRY
      F4 59   000000FF 8F   F3   008E  3371         AOBLEQ  #255,R9,20$             ; DO FOR ALL MAPS
                    0000'C2   D4   0096  3372         CLRL    VAR(R2)                 ; CLEAR VIRTUAL ADDRESS REGISTER
                    85   00   D2   009A  3373         MCOML   #0,(R5)+                ; SET UP TO PUT 1'S PATTERN ONTO MB
                    65   00   D2   009D  3374         MCOML   #0,(R5)                 ; BY WRITING 8 BYTES OF 1'S
                0000'C2   06   CE   00A0  3375         MNEGL   #6,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
          50   00000000'EF   D0   00A5  3376         MOVL    DRIVE0,R0               ; GET PSEUDO DRIVE ADDRESS
          60   00000000'8F   D0   00AC  3377         MOVL    #WRITE,(R0)             ; ISSUE WRITE TO DRIVE
    0000'C2    00000000'8F   D0   00B3  3378         MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
```

```
                0000'C2   00000000'8F   C8   00BC  3379          BISL    #SIM_SCLK,DR(R2)          ; ASSERT SCLK
                0000'C2   00000000'8F   CA   00C5  3380          BICL    #SIM_SCLK,DR(R2)          ; DROP SCLK
                0000'C2   00000000'8F   C8   00CE  3381          BISL    #DT_ABORT,CR(R2)          ; SET ABORT
                0000'C2   00000000'8F   C8   00D7  3382          BISL    #SIM_EBL,DR(R2)           ; ASSERT END OF BLOCK
                0000'C2   00000000'8F   CA   00E0  3383          BICL    #SIM_EBL,DR(R2)           ; NEGATE END OF BLOCK
                      53     0000'C2   D0   00E9  3384          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                       00000000'8F      D0   00EE  3385          MOVL    #DATA_XFER_DONE!-         ; SET UP EXPECTED STATUS
                                   54   00F4  3386                      DATA_XFER_ABRT,R4
                   54  00000000'EF      C8   00F5  3387          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                          54   53       D1   00FC  3388          CMPL    R3,R4                     ; CHECK IF OK
                               1B       13   00FF  3389          BEQL    40$                       ; IF DT COMPLETE BRANCH
                                        0101  3390          $DS_ERRHARD_S   #1,LUN,-              ; PRINT ERROR
                                        0101  3391                          ARRAY_MSG62,-
                                        0101  3392                          PRINT_SBE
                       00000000'EF      DF   0101                      PUSHAL  PRINT_SBE
                       00000000'EF      DF   0107                      PUSHAL  ARRAY_MSG62
                       00000000'EF      DD   010D                      PUSHL   LUN
                                01      DD   0113                      PUSHL   #1
                00000000'9F     04      FB   0115                      CALLS   #$$M, a#DS$ERRHARD
                                        011C  3393 40$:     $DS_CKLOOP      10$                   ; SCOPE LOOP
           00000000'9F   FF38 CF        FA   011C                      CALLG   10$, a#DS$CKLOOP
                                        0125  3394 $DS_ENDSUB
                                        0125       T15_S1_X:
        00000000'9F   000002F4'EF       FA   0125                      CALLG   $S$, a#DS$ENDSUB
```

```
                                0130  3396          $DS_BGNSUB
                                0130          T15_S2::
00000000'9F   00000300'EF   FA  0130                CALLG   $S$, a#DS$BGNSUB
                                013B  3397 ;++
                                013B  3398 ; BEGIN READ FORWARD COMMAND.
                                013B  3399 ;--
     0000'C2   00000000'8F  CA  013B  3400          BICL    #DT_ABORT,CR(R2)        ; CLEAR ABORT
          0000'C2           D4  0144  3401          CLRL    VAR(R2)                 ; VIRTUAL ADDRESS REG
          5B   01           D0  0148  3402          MOVL    #1,R11                  ; INTIALIZE MAP POINTER TO VALUE
                                014B  3403                                          ; EXPECTED AFTER TRANSFER
          0000'C2   00      D2  014B  3404 50$:     MCOML   #0,SR(R2)               ; CLEAR STATUS REGISTER
                    56      D4  0150  3405          CLRL    R6                      ; CLEAR BUFFER INDEX
        00000000'EF46       D4  0152  3406 60$:     CLRL    MIOBUFFER R6            ; WRITE 0'S PATTERN INTO BUFFER
  F1 56   0000007F 8F       F3  0159  3407          AOBLEQ  #127,R6,60$             ; INDEX AND LOOP
0000'C2   00000200 8F       CE  0161  3408          MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT
        53   0000'C2        D0  016A  3409          MOVL    VAR(R2),R3              ; READ VAR
                    53      94  016F  3410          CLRB    R3                      ; CLEAR BYTE POINTER
          53    00000100 8F  CA  0171  3411          BICL    #BIT8,R3                ; CLEAR MSB
              0000'C2   53  D0  0178  3412          MOVL    R3,VAR(R2)              ; REWRITE VAR
               00000000'8F  DD  017D  3413          PUSHL   #READ                   ; PUSH READ COMMAND FOR MM_XFER
               00000200 8F  DD  0183  3414          PUSHL   #512                    ; PUSH BYTE COUNT
                    52      DD  0189  3415          PUSHL   R2                      ; PUSH RH750 ADDRESS
          00000000'EF   03  FB  018B  3416          CALLS   #3,MM_XFER              ; TRANSFER 1 PAGE
          54   00000000'8F  D0  0192  3417          MOVL    #DATA_XFER_DONE,R4      ; EXPECT DATA TRANSFER DONE SET
          54   00000000'EF  C8  0199  3418          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  01A0  3419          MOVL    SR(R2),R3               ; GET STATUS REGISTER
                   54   53   D1  01A5  3420          CMPL    R3,R4                   ; CHECK STATUS
                   1B   13   BEQL  65$               ; BRANCH IF STATUS IS OK
                                01AA  3422          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                01AA  3423                          ARRAY_MSG62,-
                                01AA  3424                          PRINT_SBE
     00000000'EF    DF  01AA                        PUSHAL  PRINT_SBE
     00000000'EF    DF  01B0                        PUSHAL  ARRAY_MSG62
     00000000'EF    DD  01B6                        PUSHL   LUN
                    01  DD  01BC                    PUSHL   #1
00000000'9F         04  FB  01BE                    CALLS   #$$M, a#DS$ERRHARD
   50    0000'C2    D0  01C5  3425 65$:     MOVL    VAR(R2),R0                      ; READ VAR
53   50     00'8F   78  01CA  3426          ASHL    #-PF_FIELD,R0,R3                ; PUT INTO PFN FIELD
         53    5B   91  01CF  3427          CMPB    R11,R3                          ; IS MAP POINTER CORRECT?
               28   13  01D2  3428          BEQL    70$                             ; BRANCH IF OK
                          01D4  3429          $DS_ERRHARD_S   #2,LUN,-            ; PRINT ERROR
                          01D4  3430                          ARRAY_MSG4
                    00    DD  01D4                        PUSHL   #0
     00000000'EF    DF  01D6                        PUSHAL  ARRAY_MSG4
     00000000'EF    DD  01DC                        PUSHL   LUN
                    02  DD  01E2                    PUSHL   #2
00000000'9F         04  FB  01E4                    CALLS   #$$M, a#DS$ERRHARD
                          01EB  3431          $DS_PRINTX_S    FMT_VARER,-         ; WITH MAP SELECTOR
                          01EB  3432                          R3,R11
                    5B    DD  01EB                        PUSHL   R11
                    53    DD  01ED                        PUSHL   R3
     00000000'EF    9F  01EF                        PUSHAB  FMT_VARER
00000000'9F         03  FB  01F5                    CALLS   #$$N, a#DS$PRINTX
     50    5B    01   C3  01FC  3433 70$:     SUBL3   #1,R11,R0                   ; SET UP TO LOOP ON ERROR
     50    50    09   78  0200  3434          ASHL    #9,R0,R0                    ; PUT MAP SELECTOR VALUE INTO PF FIELD
     0000'C2   50   D0  0204  3435          MOVL    R0,VAR(R2)                  ; REWRITE VAR ENTRY FOR LOOP
                          0209  3436          $DS_CKLOOP      50$                 ; SCOPE LOOP?
```

```
                                                                                CALLG   50$, a#DS$CKLOOP
                                                        CLRL    R6                              ; SET UP TO TEST DATA TRANSFERRED
                                                                                                ; INTO BUFFER
                                                80$:     MCOML   MIOBUFFER R6 ,R3                ; READ DATA BUFFER
                                                        BEQL    90$                             ; BRANCH IF OK
                                                        $DS_ERRHARD_S   #3,LUN,-                ; PRINT ERROR
                                                                        ARRAY_MSG1
                                                                                PUSHL   #0
                                                                                PUSHAL  ARRAY_MSG1
                                                                                PUSHL   LUN
                                                                                PUSHL   #3
                                                                                CALLS   #$$M, a#DS$ERRHARD
                                                        MOVAL   MIOBUFFER R6 ,R0                ; GET ADDRESS OF ERROR
                                                        MCOML   R3,R3                           ; SET UP ERROR DATA FOR DISPLAY
                                                        $DS_PRINTX_S    FMT_XFERERR,-           ; PRINT ERROR PARAMETERS
                                                                        R0,-
                                                                        # XFFFFFFFF,R3
                                                                                PUSHL   R3
                                                                                PUSHL   # XFFFFFFFF
                                                                                PUSHL   R0
                                                                                PUSHAB  FMT_XFERERR
                                                                                CALLS   #$$N, a#DS$PRINTX
                                                90$:     $DS_CKLOOP      50$                     ; SCOPE LOOP?
                                                                                CALLG   50$, a#DS$CKLOOP
                                                        AOBLEQ  #127,R6,80$                     ; IF ALL THE BUFFER NOT INSPECTED LOOP
                                                        ASHL    #9,R11,R0                       ; SET UP VAR MAP POINTER FOR NEXT XFER
                                                        MOVL    R0,VAR(R2)                      ; INITIALIZE VAR FOR NEXT XFER
                                                        INCL    R11                             ; TRANSFER USING ALL MAP REGISTERS
                                                        CMPL    R11,#256                        ; ALL MAP REGISTERS USED?
                                                        BGTR    100$                            ; BRANCH IF SO
                                                        JMP     50$                             ; LOOP IF NOT
                                                100$:
                                                        $DS_ENDSUB
                                                T15_S2_X:
                                                                                CALLG   $S$, a#DS$ENDSUB
```

```
                                028B  3459          $DS_BGNSUB
                                028B        T15_S3::
00000000'9F   0000030C'EF   FA  028B                CALLG   $S$, a#DS$BGNSUB
                                0296  3460 ;++
                                0296  3461 ; BEGIN READ REVERSE BLOCK TRANSFER TEST
                                0296  3462 ;--
     0000'C2   000001FF 8F   D0  0296  3463          MOVL    # X1FF,VAR(R2)          ; INITIALIZE THE VAR
          5B   000000FF 8F   D0  029F  3464          MOVL    #255,R11                ; INTIALIZE MAP POINTER TO VALUE
                                02A6  3465                                          ; EXPECTED AFTER TRANSFER
               0000'C2   00   D2  02A6  3466 50$:     MCOML   #0,SR(R2)               ; CLEAR STATUS REGISTER
                         56   D4  02AB  3467          CLRL    R6                      ; CLEAR BUFFER INDEX
              00000000'EF46   D4  02AD  3468 60$:     CLRL    MIOBUFFER R6            ; WRITE 0'S PATTERN INTO BUFFER
       F1 56    0000007F 8F   F3  02B4  3469          AOBLEQ  #127,R6,60$             ; INDEX AND LOOP
     0000'C2    00000200 8F   CE  02BC  3470          MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT
               53   0000'C2   D0  02C5  3471          MOVL    VAR(R2),R3              ; READ VAR
          53    000001FF 8F   C8  02CA  3472          BISL    # X1FF,R3              ; VAR POINTS TO LAST BYTE IN PAGE
               0000'C2   53   D0  02D1  3473          MOVL    R3,VAR(R2)              ; REWRITE VAR
                00000000'8F   DD  02D6  3474          PUSHL   #READ_REV               ; PUSH READ COMMAND FOR MM_XFER
                00000200 8F   DD  02DC  3475          PUSHL   #512                    ; PUSH BYTE COUNT
                         52   DD  02E2  3476          PUSHL   R2                      ; PUSH RH750 ADDRESS
          00000000'EF    03   FB  02E4  3477          CALLS   #3,MM_XFER              ; TRANSFER 1 PAGE
          54    00000000'8F   D0  02EB  3478          MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
          54    00000000'EF   C8  02F2  3479          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
               53   0000'C2   D0  02F9  3480          MOVL    SR(R2),R3               ; READ STATUS
                    54   53   D1  02FE  3481          CMPL    R3,R4                   ; IS STATUS OK
                         1B   13  0301  3482          BEQL    65$                     ; BRANCH IF STATUS IS OK
                                0303  3483          $DS_ERRHARD_S   #1,LUN,-         ; PRINT ERROR
                                0303  3484                          ARRAY_MSG62,-
                                0303  3485                          PRINT_SBE
          00000000'EF   DF  0303                    PUSHAL  PRINT_SBE
          00000000'EF   DF  0309                    PUSHAL  ARRAY_MSG62
          00000000'EF   DD  030F                    PUSHL   LUN
                    01  DD  0315                    PUSHL   #1
00000000'9F       04  FB  0317                    CALLS   #$$M, a#DS$ERRHARD
     50   0000'C2   D0  031E  3486 65$:     MOVL    VAR(R2),R0                      ; READ VAR
 53   50    00'8F   78  0323  3487          ASHL    #-PF_FIELD,R0,R3                ; PUT INTO PFN FIELD
          53   5B   91  0328  3488          CMPB    R11,R3                          ; IS MAP POINTER CORRECT?
               28   13  032B  3489          BEQL    70$                             ; BRANCH IF OK
                                032D  3490          $DS_ERRHARD_S   #2,LUN,-         ; PRINT ERROR
                                032D  3491                          ARRAY_MSG4
                         00   DD  032D                    PUSHL   #0
          00000000'EF   DF  032F                    PUSHAL  ARRAY_MSG4
          00000000'EF   DD  0335                    PUSHL   LUN
                    02  DD  033B                    PUSHL   #2
00000000'9F       04  FB  033D                    CALLS   #$$M, a#DS$ERRHARD
                                0344  3492          $DS_PRINTX_S    FMT_VARER,-     ; AND PARAMETERS
                                0344  3493                          R3,R11
                         5B   DD  0344                    PUSHL   R11
                         53   DD  0346                    PUSHL   R3
          00000000'EF   9F  0348                    PUSHAB  FMT_VARER
00000000'9F       03  FB  034E                    CALLS   #$$N, a#DS$PRINTX
                         50   D4  0355  3494 70$:     CLRL    R0                      ; CLEAR VIRTUAL ADDRESS ENTRY FOR LOOP
     50   5B     01   C1  0357  3495          ADDL3   #1,R11,R0                      ; SET UP MAP SELECTOR FOR LOOP
     50   50     09   78  035B  3496          ASHL    #9,R0,R0                        ; PUT INTO PFN FIELD
50    000001FF 8F   C8  035F  3497          BISL    # X1FF,R0                      ; POINT TO LAST BYTE IN PAGE
     0000'C2   50   D0  0366  3498          MOVL    R0,VAR(R2)                      ; REWITE VAR ENTRY FOR LOOP
                         036B  3499          $DS_CKLOOP      50$                     ; SCOPE LOOP?
```

```
      00000000'9F   FF37 CF   FA  036B                          CALLG   50$, a#DS$CKLOOP
           56   0000007F 8F   D0  0374  3500          MOVL    #127,R6                  ; INIT INDEX FOR GETTING BUFFER DATA
         53   00000000'EF46   D2  037B  3501 80$:     MCOML   MIOBUFFER R6 ,R3         ; READ DATA BUFFER
                        42   13  0383  3502          BEQL    90$                      ; BRANCH IF CONTAINED ALL ONES
                                0385  3503          $DS_ERRHARD_S   #3,LUN,-         ; PRINT ERROR
                                0385  3504                          ARRAY_MSG1
                        00   DD  0385                          PUSHL   #0
                 00000000'EF   DF  0387                          PUSHAL  ARRAY_MSG1
                 00000000'EF   DD  038D                          PUSHL   LUN
                        03   DD  0393                          PUSHL   #3
           00000000'9F   04   FB  0395                          CALLS   #$$M, a#DS$ERRHARD
         50   00000000'EF46   DE  039C  3505          MOVAL   MIOBUFFER R6 ,R0         ; GET PHYSICAL ADDRESS OF ERROR
                     53   53   D2  03A4  3506          MCOML   R3,R3                    ; COMPLEMENT CONTENTS OF R3
                                03A7  3507          $DS_PRINTX_S    FMT_XFERERR,-    ; PRINT PARAMETERS
                                03A7  3508                          R0,-
                                03A7  3509                          # XFFFFFFFF,R3
                        53   DD  03A7                          PUSHL   R3
                 FFFFFFFF 8F   DD  03A9                          PUSHL   # XFFFFFFFF
                        50   DD  03AF                          PUSHL   R0
                 00000000'EF   9F  03B1                          PUSHAB  FMT_XFERERR
           00000000'9F   04   FB  03B7                          CALLS   #$$N, a#DS$PRINTX
                                03BE  3510          $DS_CKLOOP      50$              ; SCOPE LOOP?
      00000000'9F   FEE4 CF   FA  03BE                          CALLG   50$, a#DS$CKLOOP
                      B1 56   F4  03C7  3511 90$:     SOBGEQ  R6,80$                   ; DO FOR ENTIRE BUFFER
              50   5B   09   78  03CA  3512          ASHL    #9,R11,R0                ; PUT PF VALUE INTO R0
         50   000001FF 8F   C8  03CE  3513          BISL    # X1FF,R0                ; POINT TO LAST BYTE IN PAGE
              0000'C2   50   D0  03D5  3514          MOVL    R0,VAR(R2)               ; WRITE VAR
                        5B   D7  03DA  3515          DECL    R11                      ; POINT TO NEXT PAGE
                     00   5B   D1  03DC  3516          CMPL    R11,#0                   ; GOTTEN TO ZERO YET?
                        04   19  03DF  3517          BLSS    100$                     ; BRANCH IF SO
                   FEC1 CF   17  03E1  3518          JMP     50$                      ; LOOP IF NOT
                                03E5  3519 100$:
                                03E5  3520          $DS_ENDSUB
                                03E5        T15_S3_X:
   00000000'9F   0000030C'EF   FA  03E5                          CALLG   $S$, a#DS$ENDSUB
```

```
                                03F0  3522          $DS_BGNSUB
                                03F0          T15_S4::
00000000'9F   00000318'EF   FA  03F0                CALLG   $S$, a#DS$BGNSUB
                                03FB  3523 ;++
                                03FB  3524 ; BEGIN WRITE CHECK REVERSE BLOCK TRANSFER TEST
                                03FB  3525 ;--
     0000'C2   000001FF 8F  D0  03FB  3526          MOVL    # X1FF,VAR(R2)          ; INITIALIZE THE VAR
          5B   000000FF 8F  D0  0404  3527          MOVL    #255,R11                ; INTIALIZE MAP POINTER TO VALUE
                                040B  3528                                          ; EXPECTED AFTER TRANSFER
               0000'C2   00  D2  040B  3529 50$:    MCOML   #0,SR(R2)               ; CLEAR STATUS REGISTER
     0000'C2    00000200 8F  CE  0410  3530          MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT
               53   0000'C2  D0  0419  3531          MOVL    VAR(R2),R3              ; READ VAR
          53    000001FF 8F  C8  041E  3532          BISL    # X1FF,R3              ; POINT TO LAST BYTE IN PAGE
               0000'C2   53  D0  0425  3533          MOVL    R3,VAR(R2)              ; REWRITE VAR
                00000000'8F  DD  042A  3534          PUSHL   #WRITE_CHECKREV         ; PUSH READ COMMAND FOR MM_XFER
                00000200 8F  DD  0430  3535          PUSHL   #512                    ; PUSH BYTE COUNT
                         52  DD  0436  3536          PUSHL   R2                      ; PUSH RH750 ADDRESS
          00000000'EF   03  FB  0438  3537          CALLS   #3,MM_XFER              ; TRANSFER 1 PAGE
          54   00000000'8F  D0  043F  3538          MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
          54   00000000'EF  C8  0446  3539          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2  D0  044D  3540          MOVL    SR(R2),R3               ; READ STATUS
                   54   53  D1  0452  3541          CMPL    R3,R4                   ; IS STATUS OK
                        1B  13  0455  3542          BEQL    65$                     ; BRANCH IF STATUS IS OK
                                0457  3543          $DS_ERRHARD_S   #1,LUN,-        ; PRINT ERROR
                                0457  3544                          ARRAY_MSG62,-
                                0457  3545                          PRINT_SBE
                00000000'EF   DF  0457                        PUSHAL  PRINT_SBE
                00000000'EF   DF  045D                        PUSHAL  ARRAY_MSG62
                00000000'EF   DD  0463                        PUSHL   LUN
                         01   DD  0469                        PUSHL   #1
          00000000'9F   04   FB  046B                        CALLS   #$$M, a#DS$ERRHARD
              50   0000'C2   D0  0472  3546 65$:     MOVL    VAR(R2),R0              ; READ VAR
           53   50    00'8F   78  0477  3547          ASHL    #-PF_FIELD,R0,R3        ; PUT PFN INTO R3
                53   5B   91  047C  3548          CMPB    R11,R3                  ; IS MAP POINTER CORRECT?
                         28  13  047F  3549          BEQL    70$                     ; BRANCH IF OK
                                0481  3550          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                0481  3551                          ARRAY_MSG4
                         00  DD  0481                        PUSHL   #0
                00000000'EF   DF  0483                        PUSHAL  ARRAY_MSG4
                00000000'EF   DD  0489                        PUSHL   LUN
                         02   DD  048F                        PUSHL   #2
          00000000'9F   04   FB  0491                        CALLS   #$$M, a#DS$ERRHARD
                                0498  3552          $DS_PRINTX_S    FMT_VARER,-     ; PRINT PARAMETERS
                                0498  3553                          R3,R11
                         5B  DD  0498                        PUSHL   R11
                         53  DD  049A                        PUSHL   R3
                00000000'EF   9F  049C                        PUSHAB  FMT_VARER
          00000000'9F   03   FB  04A2                        CALLS   #$$N, a#DS$PRINTX
                         50  D4  04A9  3554 70$:     CLRL    R0                      ; CLEAR VIRTUAL ADDRESS ENTRY FOR LOOP
              50   5B   01  C1  04AB  3555          ADDL3   #1,R11,R0               ; SET UP MAP SELECTOR FOR LOOP
              50   50   09  78  04AF  3556          ASHL    #9,R0,R0                ; ADJUST TO PFN FIELD
          50   000001FF 8F  C8  04B3  3557          BISL    # X1FF,R0              ; POINT TO LAST BYTE IN PAGE
              0000'C2   50  D0  04BA  3558          MOVL    R0,VAR(R2)              ; REWITE VAR ENTRY FOR LOOP
                                04BF  3559          $DS_CKLOOP      50$             ; SCOPE LOOP?
00000000'9F   FF48 CF   FA  04BF                        CALLG   50$, a#DS$CKLOOP
     56   0000007F 8F  D0  04C8  3560          MOVL    #127,R6                 ; SET UP INDEX TO GET BUFFER DATA
   53   00000000'EF46   D2  04CF  3561 80$:     MCOML   MIOBUFFER R6 ,R3        ; READ DATA BUFFER
```

```
                    42   13  04D7  3562          BEQL    90$                     ; BRANCH IF CONTAINS ALL ONES
                              04D9  3563          $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                              04D9  3564                          ARRAY_MSG1
                    00   DD  04D9                          PUSHL   #0
          00000000'EF   DF  04DB                          PUSHAL  ARRAY_MSG1
          00000000'EF   DD  04E1                          PUSHL   LUN
                    03   DD  04E7                          PUSHL   #3
     00000000'9F    04   FB  04E9                          CALLS   #$$M, a#DS$ERRHARD
   50   C0000000'EF46   DE  04F0  3565          MOVAL   MIOBUFFER R6 ,R0        ; GET PHYSICAL ADDRESS OF ERROR
              53    53   D2  04F8  3566          MCOML   R3,R3                   ; COMPLEMENT CONTENTS OF R3
                              04FB  3567          $DS_PRINTX_S    FMT_XFERERR,-   ; PRINT PARAMETERS
                              04FB  3568                          R0,-
                              04FB  3569                          # XFFFFFFFF,R3
                    53   DD  04FB                          PUSHL   R3
          FFFFFFFF 8F   DD  04FD                          PUSHL   # XFFFFFFFF
                    50   DD  0503                          PUSHL   R0
          00000000'EF   9F  0505                          PUSHAB  FMT_XFERERR
     00000000'9F    04   FB  050B                          CALLS   #$$N, a#DS$PRINTX
                              0512  3570          $DS_CKLOOP      50$             ; SCOPE LOOP?
  00000000'9F   FEF5 CF   FA  0512                          CALLG   50$, a#DS$CKLOOP
                 B1 56   F4  051B  3571 90$:     SOBGEQ  R6,80$                  ; DO FOR ENTIRE BUFFER
          50    5B   09   78  051E  3572          ASHL    #9,R11,R0               ; PUT PF VALUE INTO R0
     50   000001FF 8F   C8  0522  3573          BISL    # X1FF,R0               ; POINT TO LAST BYTE IN PAGE
          0000'C2   50   D0  0529  3574          MOVL    R0,VAR(R2)              ; WRITE VAR
                    5B   D7  052E  3575          DECL    R11                     ; POINT TO NEXT BUFFER LOCATION
              00    5B   D1  0530  3576          CMPL    R11,#0                  ; GOTTEN TO LAST BUFFER LOCATION?
                    04   19  0533  3577          BLSS    100$                    ; BRANCH IF SO
              FED2 CF   17  0535  3578          JMP     50$                     ; LOOP IF NOT
                              0539  3579 100$:
                              0539  3580          $DS_ENDSUB
                              0539        T15_S4_X:
00000000'9F   00000318'EF   FA  0539                          CALLG   $S$, a#DS$ENDSUB
```

```
                        0544  3582          $DS_BGNSUB
                        0544          T15_S5::
00000000'9F  00000324'EF  FA  0544                CALLG   $S$, a#DS$BGNSUB
                        054F  3583          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                        054F  3584                  FMT_STATUS_REG,-
                        054F  3585                  UNEXPECTED
00000000'EF  00000000'EF  DE  054F                MOVAL   RHSR_MSG,REG_NAME
      00000000'EF  02     9A  055A                MOVZBL  #2,REG_NO
00000000'EF  00000000'EF  DE  0561                MOVAL   FMT_STATUS_REG,REG_STRING
          0000'C2  00     D2  056C  3586          MCOML   #0,SR(R2)               ; CLEAR STATUS REGISTER
                  56      D4  0571  3587          CLRL    R6                      ; CLEAR BUFFER SELECTOR INDEX
    00000000'EF46  00     D2  0573  3588 10$:     MCOML   #0,MIOBUFFER R6         ; WRITE 1'S PATTERN INTO BUFFER
  F0 56  0000007F  8F     F3  057B  3589          AOBLEQ  #127,R6,10$             ; DO FOR ENTIRE BUFFER
0000'C2  00000200  8F     CE  0583  3590          MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT INTO BCR
            0000'C2  D4     058C  3591          CLRL    VAR(R2)                 ; CLEAR VIRTUAL ADDRESS REG
          00000000'8F  DD     0590  3592          PUSHL   #WRITE_CHECK            ; PUSH WRITE CHECK COMMAND
          00000200 8F  DD     0596  3593          PUSHL   #512                    ; PUSH BYTE COUNT
                  52   DD     059C  3594          PUSHL   R2                      ; PUSH RH750 ADDRESS
      00000000'EF  03     FB  059E  3595          CALLS   #3,MM_XFER              ; INITIATE WRITE CHECK
          53  0000'C2  D0     05A5  3596          MOVL    SR(R2),R3               ; GET STATUS
      54  00000000'8F  D0     05AA  3597          MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
      54  00000000'EF  C8     05B1  3598          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
            54  53     D1     05B8  3599          CMPL    R3,R4                   ; CHECK IF OK
                  1B   13     05BB  3600          BEQL    30$                     ; BRANCH IF NO ERROR
                        05BD  3601          $DS_ERRHARD_S   #1,LUN,-            ; PRINT ERROR
                        05BD  3602                  ARRAY_MSG62,-
                        05BD  3603                  PRINT_SBE
      00000000'EF  DF     05BD                    PUSHAL  PRINT_SBE
      00000000'EF  DF     05C3                    PUSHAL  ARRAY_MSG62
      00000000'EF  DD     05C9                    PUSHL   LUN
                  01  DD     05CF                PUSHL   #1
   00000000'9F  04     FB  05D1                CALLS   #$$M, a#DS$ERRHARD
                        05D8  3604 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
 00000000'9F  98 AF     FA  05D8                CALLG   10$, a#DS$CKLOOP
0000'C2  00000200 8F     CE  05E0  3605 40$:     MNEGL   #512,BCR(R2)            ; WRITE BYTE COUNT
0000'C2  000001FF 8F     D0  05E9  3606          MOVL    #511,VAR(R2)            ; WRITE VIRTUAL ADDRESS REGISTER
      00000000'8F  DD     05F2  3607          PUSHL   #WRITE_CHECKREV         ; PUSH COMMAND FOR MM_XFER
      00000200 8F  DD     05F8  3608          PUSHL   #512                    ; PUSH BYTE COUNT--SCLK MOD 2
                  52   DD     05FE  3609          PUSHL   R2                      ; PUSH RH750 ADDRESS
   00000000'EF  03     FB  0600  3610          CALLS   #3,MM_XFER              ; INITATE WRITE CHECK REVERSE
   54  00000000'8F  D0     0607  3611          MOVL    #DATA_XFER_DONE,R4      ; SET UP EXPECTED STATUS
   54  00000000'EF  C8     060E  3612          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
       53  0000'C2  D0     0615  3613          MOVL    SR(R2),R3               ; READ STATUS REGISTER
            54  53     D1     061A  3614          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                  1B   13     061D  3615          BEQL    50$                     ; IF STATUS IS AS EXPECTED BRANCH
                        061F  3616          $DS_ERRHARD_S   #2,LUN,-            ; PRINT ERROR
                        061F  3617                  ARRAY_MSG61,-
                        061F  3618                  PRINT_SBE
      00000000'EF  DF     061F                    PUSHAL  PRINT_SBE
      00000000'EF  DF     0625                    PUSHAL  ARRAY_MSG61
      00000000'EF  DD     062B                    PUSHL   LUN
                  02  DD     0631                PUSHL   #2
   00000000'9F  04     FB  0633                CALLS   #$$M, a#DS$ERRHARD
                        063A  3619 50$:     $DS_CKLOOP      40$             ; SCOPE LOOP?
 00000000'9F  A3 AF     FA  063A                CALLG   40$, a#DS$CKLOOP
                        0642  3620          $DS_ENDSUB
                        0642          T15_S5_X:
```

```
00000000'9F   00000324'EF   FA  0642                          CALLG   $S$, a#DS$ENDSUB
                            064D  3621 $DS_ENDTEST
                      50    01  D0  064D                      MOVL    #1, R0          ; NORMAL EXIT
                            0650       TEST_015_X::
              00000000'9F   6E  FA  0650                      CALLG   (SP), a#DS$BREAK
                            04  0657                         RET                     ; RETURN TO TEST SEQUENCER
                                0658  3622 $DS_ENDMOD
                            00000000                        .PSECT  $TSTCNT, NOEXE, NOWRT, OVR, LONG
                       0000000F 0000                        .LONG   $TN
                                0004  3623 .END
```

```
$$$                     = 00000324 R      04
$$E                     = 00000001
$$M                     = 00000004
$$N                     = 00000000
$$S                     = FFFFFFFF
$ENV                    = 00000001  G
$ER                     = FFFFFFFF
$MO                     = 00000001  G
$S$                     = 00000324 R      04
$ST                     = 00000000
$TN                     = 0000000F
ABORT_IF_FAIL             ********    X   07
ALL                     = 00000001  G
ARRAY_MSG1                ********    X   02
ARRAY_MSG10               ********    X   07
ARRAY_MSG11               ********    X   08
ARRAY_MSG12               ********    X   08
ARRAY_MSG13               ********    X   08
ARRAY_MSG14               ********    X   02
ARRAY_MSG15               ********    X   0F
ARRAY_MSG16               ********    X   08
ARRAY_MSG17               ********    X   08
ARRAY_MSG18               ********    X   08
ARRAY_MSG19               ********    X   08
ARRAY_MSG2                ********    X   07
ARRAY_MSG20               ********    X   07
ARRAY_MSG21               ********    X   07
ARRAY_MSG22               ********    X   02
ARRAY_MSG23               ********    X   05
ARRAY_MSG24               ********    X   0C
ARRAY_MSG26               ********    X   08
ARRAY_MSG27               ********    X   08
ARRAY_MSG29               ********    X   0B
ARRAY_MSG3                ********    X   0D
ARRAY_MSG31               ********    X   06
ARRAY_MSG32               ********    X   0A
ARRAY_MSG33               ********    X   07
ARRAY_MSG34               ********    X   0F
ARRAY_MSG35               ********    X   06
ARRAY_MSG36               ********    X   0D
ARRAY_MSG37               ********    X   06
ARRAY_MSG38               ********    X   08
ARRAY_MSG39               ********    X   07
ARRAY_MSG4                ********    X   0D
ARRAY_MSG40               ********    X   07
ARRAY_MSG41               ********    X   08
ARRAY_MSG42               ********    X   07
ARRAY_MSG43               ********    X   07
ARRAY_MSG44               ********    X   07
ARRAY_MSG45               ********    X   07
ARRAY_MSG46               ********    X   0D
ARRAY_MSG47               ********    X   0F
ARRAY_MSG48               ********    X   0F
ARRAY_MSG49               ********    X   0F
ARRAY_MSG5                ********    X   0D
ARRAY_MSG50               ********    X   0D
ARRAY_MSG51               ********    X   0D
ARRAY_MSG52               ********    X   0D
ARRAY_MSG53               ********    X   0B
ARRAY_MSG56               ********    X   05
ARRAY_MSG57               ********    X   06
ARRAY_MSG6                ********    X   0D
ARRAY_MSG60               ********    X   02
ARRAY_MSG61               ********    X   12
ARRAY_MSG62               ********    X   0C
ARRAY_MSG7                ********    X   07
ARRAY_MSG8                ********    X   07
ARRAY_MSG9                ********    X   07
ASR_OFFSET                ********    X   0F
ATTENTION                 ********    X   0D
BASE_001                  00000000 R      02
BASE_002                  00000000 R      05
BASE_003                  00000000 R      06
BASE_004                  00000000 R      07
BASE_005                  00000000 R      08
BASE_006                  00000000 R      09
BASE_007                  00000000 R      0A
BASE_008                  00000000 R      0B
BASE_009                  00000000 R      0C
BASE_010                  00000000 R      0D
BASE_011                  00000000 R      0E
BASE_012                  00000000 R      0F
BASE_013                  00000000 R      10
BASE_014                  00000000 R      11
BASE_015                  00000000 R      12
BCR                       ********    X   05
BIT...                  = 00000002
BIT0                    = 00000001
BIT1                    = 00000002
BIT10                   = 00000400
BIT11                   = 00000800
BIT12                   = 00001000
BIT13                   = 00002000
BIT14                   = 00004000
BIT15                   = 00008000
BIT16                   = 00010000
BIT17                   = 00020000
BIT18                   = 00040000
BIT19                   = 00080000
BIT2                    = 00000004
BIT20                   = 00100000
BIT21                   = 00200000
BIT22                   = 00400000
BIT23                   = 00800000
BIT24                   = 01000000
BIT25                   = 02000000
BIT26                   = 04000000
BIT27                   = 08000000
BIT28                   = 10000000
BIT29                   = 20000000
BIT3                    = 00000008
BIT30                   = 40000000
BIT31                   = 80000000
BIT4                    = 00000010
```

```
BIT5                            = 00000020
BIT6                            = 00000040
BIT7                            = 00000080
BIT8                            = 00000100
BIT9                            = 00000200
BLKSND_COMD                       ********    X   07
BYTE0                             ********    X   05
BYTE1                             ********    X   05
BYTE2                             ********    X   0B
BYTE3                             ********    X   0B
CAR                               ********    X   0B
CEP_FUNCTIONAL                  = 00000000
CEP_REPAIR                      = 00000001
COMP_MSG1                         ********    X   0D
COMP_MSG2                         ********    X   0D
CONT_PC                           ********    X   02
CR                                ********    X   02
CRTED_READ_DATA                   ********    X   0F
CSR                               ********    X   02
DATA_001                          00000030 R      02
DATA_002                          00000018 R      05
DATA_003                          0000001C R      06
DATA_004                          0000001C R      07
DATA_005                          00000030 R      08
DATA_006                          00000018 R      09
DATA_007                          00000024 R      0A
DATA_008                          00000018 R      0B
DATA_009                          00000024 R      0C
DATA_010                          00000024 R      0D
DATA_011                          00000018 R      0E
DATA_012                          00000024 R      0F
DATA_013                          00000018 R      10
DATA_014                          00000014 R      11
DATA_015                          00000020 R      12
DATA_IBUFFER                      ********      X   0C
DATA_OBUFFER                      ********      X   0C
DATA_XFER_ABRT                    ********      X   0D
DATA_XFER_DONE                    ********      X   0C
DATA_XFER_LATE                    ********      X   0D
DCM                               ********      X   0F
DEFAULT                         = 00000000  G
DR                                ********    X   02
DRIVE0                            ********    X   02
DRIVE_ADR_TBL                     ********    X   07
DRIVE_OFFSET                      ********    X   0D
DS$BGNSUB                         ********    X   02
DS$BREAK                          ********    X   02
DS$CKLOOP                         ********    X   02
DS$CLRVEC                         ********    X   02
DS$CVTREG                         ********    X   02
DS$ENDSUB                         ********    X   02
DS$ERRHARD                        ********    X   02
DS$ESCAPE                         ********    X   07
DS$PRINTB                         ********    X   0C
DS$PRINTX                         ********    X   02
DS$SETIPL                         ********    X   0D
DS$SETVEC                         ********    X   02
DS$WAITUS                         ********    X   0C
DSA$AL_APTMAIL                    0000FE00
DSA$AT_APTTXT                     0000FA00
DSA$GL_APTCOM                     0000FE04
DSA$GL_DEVLEN                     0000FE58
DSA$GL_ERRNO                      0000FE44
DSA$GL_EVENT                      0000FE48
DSA$GL_FLAGS                      0000FE00
DSA$GL_MSGTYP                     0000FE40
DSA$GL_PASSES                     0000FE08
DSA$GL_PASSNO                     0000FE54
DSA$GL_SECTNO                     0000FE10
DSA$GL_SID                        0000FE14
DSA$GL_SUBTNO                     0000FE4C
DSA$GL_TESTNO                     0000FE50
DSA$GL_UNITS                      0000FE0C
DSA$GQ_MSGPTR                     0000FE68
DSA$GT_DEVNAM                     0000FE5C
DSA$M_TRACE                     = 00000400
DSA$V_OPER                      = 0000000C
DSA$V_QUICK                     = 00000008
DT_ABORT                          ********    X   06
DT_BUSY                           ********    X   0D
EDM                               ********    X   0F
ENV$M_DOMAIN                    = 00000002
ENV$M_LEVEL                     = 00000001
ENV$M_SUPER                     = 000003FC
ENV$S_DOMAIN                    = 00000001
ENV$S_LEVEL                     = 00000001
ENV$S_SUPER                     = 00000008
ENV$V_DOMAIN                    = 00000001
ENV$V_LEVEL                     = 00000000
ENV$V_SUPER                     = 00000002
ENV$_CPU                        = 00000000
ENV$_FUNCTIONAL                 = 00000000
ENV$_REPAIR                     = 00000001
ENV$_SUPER                      = 00000001
ENV$_SYSTEM                     = 00000001
ERCE                              ********    X   0F
ERR_STAT                          ********    X   0F
EXT_REG_MSG                       ********    X   0A
FMT_ANY_REG                       ********    X   02
FMT_BADNEXUS                      ********    X   0F
FMT_CONTRL_REG                    ********    X   05
FMT_DIAG_REG                      ********    X   05
FMT_DRIVENORES                    ********    X   0D
FMT_DRIVERES                      ********    X   0D
FMT_DTERR                         ********    X   10
FMT_DUADR_ERR                     ********    X   09
FMT_DUALRS_ERR                    ********    X   02
FMT_DUALRS_MSG                    ********    X   02
FMT_DUALWT_ERR                    ********    X   09
FMT_DUALWT_MSG                    ********    X   09
FMT_ERRSUM                        ********    X   0C
FMT_IBUS0                         ********    X   02
FMT_IBUS1                         ********    X   02
FMT_MAP_REG                       ********    X   05
```

FMT_STATUS_REG	********	X	05	MSG_012	00000002	R	0F
FMT_VARER	********	X	12	MSG_013	00000002	R	10
FMT_XFERERR	********	X	12	MSG_014	00000002	R	11
HP$A_DEPENDENT	00000032			MSG_015	00000002	R	12
HP$A_DEVICE	00000018			NIBBLE1	********	X	0B
HP$A_DVA	0000001C			NIBBLE4	********	X	07
HP$A_LINK	00000020			NODRIVE	********	X	0D
HP$B_DRIVE	0000000B			NOOP	********	X	07
HP$B_FLAGS	0000000A			NO_RESPONSE	********	X	0F
HP$B_RH750_BR	00000032			NXM_ADR	********	X	02
HP$K_RH750_LEN	00000033			NXM_RD_FLAG	********	X	02
HP$Q_DEVICE	00000000			PF_FIELD	********	X	0B
HP$T_DEVICE	0000000C			PGM_INIT	********	X	02
HP$T_TYPE	00000026			PM	********	X	0F
HP$W_SIZE	00000008			PRINT_CMDER	********	X	0B
HP$W_VECTOR	00000024			PRINT_DTERR	********	X	0C
IBC_MSG	********	X	0F	PRINT_IBCERR	********	X	0F
INH_BYTE_CNT	********	X	06	PRINT_MAPERR	********	X	0E
INTERRUPT_EN	********	X	06	PRINT_NXADR	********	X	02
INVRT_MAP_PAR	********	X	07	PRINT_SBE	********	X	05
INVRT_MB_CPAR	********	X	07	PROGRAM_ERROR	********	X	0D
INVRT_MB_DPAR	********	X	07	READ	********	X	0B
INVRT_SILO_PAR	********	X	07	READ_REV	********	X	12
ISSUE_MMREAD	********	X	0E	REG_NAME	********	X	05
LUN	********	X	02	REG_NO	********	X	05
MACH_CHK_SRV	********	X	02	REG_STRING	********	X	05
MAINT_MODE	********	X	02	REPORT_BUFFER	********	X	02
MAP_1_PATTRN	********	X	09	RH11TB_MSK	********	X	0A
MAP_INVALID	********	X	0F	RH11TB_PRES	********	X	0A
MAP_OFFSET	********	X	02	RH750$B_BR	00000032		
MAP_PE	********	X	0F	RH750$K_LEN	00000033		
MAP_PTR_MSK	********	X	0C	RHBCR_MSG	********	X	05
MASS_CNTRL_PE	********	X	0F	RHCR_MSG	********	X	05
MASS_CTOD	********	X	05	RHDR_MSG	********	X	05
MASS_DATA_PE	********	X	0F	RHMAPR_MSG	********	X	05
MASS_ECP	********	X	0F	RHSR_MSG	********	X	05
MASS_FAIL	********	X	07	RHVAR_MSG	********	X	05
MASS_RUN	********	X	0B	RH_CUR_ADR	********	X	02
MASS_WCLK	********	X	07	SEP_FUNCTIONAL	= 00000002		
MBDIB_SEL	********	X	07	SEP_REPAIR	= 00000003		
MBE	= 00000002	G		SILO_PE	********	X	0F
MEM_CSR0	********	X	0F	SIM_ATTN	********	X	07
MEM_CSR1	********	X	0F	SIM_EBL	********	X	07
MIOBUFFER	********	X	0B	SIM_EXCEPT	********	X	07
MISSED_XFER	********	X	0D	SIM_OCC	********	X	07
MM_XFER	********	X	0F	SIM_SCLK	********	X	07
MSG_001	00000002	R	02	SIZ...	= 00000008		
MSG_002	00000002	R	05	SR	********	X	05
MSG_003	00000002	R	06	T10_S1	0000006C	RG	0D
MSG_004	00000002	R	07	T10_S10	00000804	RG	0D
MSG_005	00000002	R	08	T10_S10_X	000008B0	R	0D
MSG_006	00000002	R	09	T10_S11	000008BB	RG	0D
MSG_007	00000002	R	0A	T10_S11_X	00000942	R	0D
MSG_008	00000002	R	0B	T10_S12	0000094D	RG	0D
MSG_009	00000002	R	0C	T10_S12_X	000009E5	R	0D
MSG_010	00000002	R	0D	T10_S13	000009F0	RG	0D
MSG_011	00000002	R	0E	T10_S13_X	00000A75	R	0D

T10_S1_X	00000170 R	0D	T14_S3	00000186 RG	11		
T10_S2	0000017B RG	0D	T14_S3_X	0000028C R	11		
T10_S2_X	00000219 R	0D	T14_S4	00000297 RG	11		
T10_S3	00000224 RG	0D	T14_S4_X	0000033F R	11		
T10_S3_X	00000316 R	0D	T15_S1	00000026 RG	12		
T10_S4	00000321 RG	0D	T15_S1_X	00000125 R	12		
T10_S4_X	000003F6 R	0D	T15_S2	00000130 RG	12		
T10_S5	00000401 RG	0D	T15_S2_X	00000280 R	12		
T10_S5_X	00000506 R	0D	T15_S3	0000028B RG	12		
T10_S6	00000511 RG	0D	T15_S3_X	000003E5 R	12		
T10_S6_X	000005C5 R	0D	T15_S4	000003F0 RG	12		
T10_S7	000005D0 RG	0D	T15_S4_X	00000539 R	12		
T10_S7_X	00000681 R	0D	T15_S5	00000544 RG	12		
T10_S8	0000068C RG	0D	T15_S5_X	00000642 R	12		
T10_S8_X	0000073D R	0D	T1_S1	00000036 RG	02		
T10_S9	00000748 RG	0D	T1_S1_X	000000C1 R	02		
T10_S9_X	000007F9 R	0D	T1_S2	000000CC RG	02		
T11_S1	00000031 RG	0E	T1_S2_X	0000015A R	02		
T11_S1_X	000000EE R	0E	T1_S3	00000165 RG	02		
T11_S2	000000F9 RG	0E	T1_S3_X	000001F4 R	02		
T11_S2_X	000001C2 R	0E	T1_S4	000001FF RG	02		
T12_S1	0000002A RG	0F	T1_S4_X	00000308 R	02		
T12_S10	00000838 RG	0F	T1_S5	00000313 RG	02		
T12_S10_X	0000094B R	0F	T1_S5_X	00000423 R	02		
T12_S11	00000956 RG	0F	T3_S1	00000046 RG	06		
T12_S11_X	000009B4 R	0F	T3_S1_X	0000008F R	06		
T12_S12	000009BF RG	0F	T3_S2	0000009A RG	06		
T12_S12_X	00000A31 R	0F	T3_S2_X	000001BA R	06		
T12_S13	00000A3C RG	0F	T4_S1	00000046 RG	07		
T12_S13_X	00000B6F R	0F	T4_S1_X	00000090 R	07		
T12_S14	00000B7A RG	0F	T4_S2	0000009B RG	07		
T12_S14_X	00000C68 R	0F	T4_S2_X	00000104 R	07		
T12_S1_X	000000C9 R	0F	T4_S3	0000010F RG	07		
T12_S2	000000D4 RG	0F	T4_S3_X	0000016E R	07		
T12_S2_X	000001DD R	0F	T4_S4	00000179 RG	07		
T12_S3	000001E8 RG	0F	T4_S4_X	000006A8 R	07		
T12_S3_X	00000296 R	0F	T5_S1	00000036 RG	08		
T12_S4	000002A1 RG	0F	T5_S1_X	000000B4 R	08		
T12_S4_X	00000379 R	0F	T5_S2	000000BF RG	08		
T12_S5	00000384 RG	0F	T5_S2_X	00000668 R	08		
T12_S5_X	0000045F R	0F	T6_S1	00000052 RG	09		
T12_S6	0000046A RG	0F	T6_S1_X	000000D1 R	09		
T12_S6_X	00000555 R	0F	T6_S2	000000DC RG	09		
T12_S7	00000560 RG	0F	T6_S2_X	00000129 R	09		
T12_S7_X	0000063F R	0F	T6_S3	00000134 RG	09		
T12_S8	0000064A RG	0F	T6_S3_X	000001D9 R	09		
T12_S8_X	00000720 R	0F	T6_S4	000001E4 RG	09		
T12_S9	0000072B RG	0F	T6_S4_X	00000256 R	09		
T12_S9_X	0000082D R	0F	T8_S1	0000001E RG	0B		
T13_S1	0000001E RG	10	T8_S1_X	000000F5 R	0B		
T13_S1_X	00000122 R	10	T8_S2	00000100 RG	0B		
T13_S2	0000012D RG	10	T8_S2_X	000001D6 R	0B		
T13_S2_X	00000275 R	10	T8_S3	000001E1 RG	0B		
T14_S1	0000001A RG	11	T8_S3_X	000002B7 R	0B		
T14_S1_X	000000EE R	11	T8_S4	000002C2 RG	0B		
T14_S2	000000F9 RG	11	T8_S4_X	00000398 R	0B		
T14_S2_X	0000017B R	11	T9_S1	0000002A RG	0C		

T9_S1_X	000001F2	R	0C	VAR_PAT20	********	X	08
T9_S2	000001FD	RG	0C	VAR_PAT21	********	X	08
T9_S2_X	000003AC	R	0C	VAR_PAT22	********	X	08
T9_S3	000003B7	RG	0C	VAR_PAT23	********	X	08
T9_S3_X	00000550	R	0C	VAR_PAT24	********	X	08
T9_S4	0000055B	RG	0C	VAR_PAT3	********	X	08
T9_S4_X	000006DE	R	0C	VAR_PAT4	********	X	08
T9_S5	000006E9	RG	0C	VAR_PAT5	********	X	08
T9_S5_X	0000086C	R	0C	VAR_PAT6	********	X	08
T9_S6	00000877	RG	0C	VAR_PAT7	********	X	08
T9_S6_X	000009FA	R	0C	VAR_PAT8	********	X	08
T9_S7	00000A05	RG	0C	VAR_PAT9	********	X	08
T9_S7_X	00000B88	R	0C	WRITE	********	X	0B
TEST_001	00000034	RG	02	WRITE_CHECK	********	X	0B
TEST_001_X	00000431	RG	02	WRITE_CHECKREV	********	X	12
TEST_002	0000001C	RG	05	WRITE_CHK_HIGH	********	X	0F
TEST_002_X	00000210	RG	05	WRITE_CHK_LOW	********	X	0F
TEST_003	00000020	RG	06				
TEST_003_X	000001C8	RG	06				
TEST_004	00000020	RG	07				
TEST_004_X	000006B6	RG	07				
TEST_005	00000034	RG	08				
TEST_005_X	00000676	RG	08				
TEST_006	0000001C	RG	09				
TEST_006_X	00000264	RG	09				
TEST_007	00000028	RG	0A				
TEST_007_X	00000181	RG	0A				
TEST_008	0000001C	RG	0B				
TEST_008_X	000003A6	RG	0B				
TEST_009	00000028	RG	0C				
TEST_009_X	00000B96	RG	0C				
TEST_010	00000028	RG	0D				
TEST_010_X	00000A83	RG	0D				
TEST_011	0000001C	RG	0E				
TEST_011_X	000001D0	RG	0E				
TEST_012	00000028	RG	0F				
TEST_012_X	00000C76	RG	0F				
TEST_013	0000001C	RG	10				
TEST_013_X	00000283	RG	10				
TEST_014	00000018	RG	11				
TEST_014_X	0000034D	RG	11				
TEST_015	00000024	RG	12				
TEST_015_X	00000650	RG	12				
VALID_BIT	********		X	09			
VAR	********		X	02			
VAR_PAT1	********		X	08			
VAR_PAT10	********		X	08			
VAR_PAT11	********		X	08			
VAR_PAT12	********		X	08			
VAR_PAT13	********		X	08			
VAR_PAT14	********		X	08			
VAR_PAT15	********		X	08			
VAR_PAT16	********		X	08			
VAR_PAT17	********		X	08			
VAR_PAT18	********		X	08			
VAR_PAT19	********		X	08			
VAR_PAT2	********		X	08			

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	0000FE70 (65136.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
TEST_001	00000439 (1081.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
DISPATCH	00000168 (360.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
ARGLIST	00000330 (816.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG
TEST_002	00000218 (536.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_003	000001D0 (464.)	06 (6.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_004	000006BE (1726.)	07 (7.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_005	0000067E (1662.)	08 (8.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_006	0000026C (620.)	09 (9.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_007	00000189 (393.)	0A (10.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_008	000003AE (942.)	0B (11.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_009	00000B9E (2974.)	0C (12.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_010	00000A8B (2699.)	0D (13.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_011	000001D8 (472.)	0E (14.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_012	00000C7E (3198.)	0F (15.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_013	0000028B (651.)	10 (16.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_014	00000355 (853.)	11 (17.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_015	00000658 (1624.)	12 (18.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
$TSTCNT	00000004 (4.)	13 (19.)	NOPIC USR OVR REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG

Symbol Cross Reference

SYMBOL	VALUE	DEFINITION	REFERENCES...
$$$	=00000324-R	3582 (71)	1078 (16) 1105 (17) 1119 (18) 1153 (19) 1258 (21) 1295 (22) 1333 (23) 1370 (24) 1424 (25) 1498 (26) 152 (2) 1568 (27) 1631 (28) 1696 (29) 1760 (30) 1824 (31) 185 (3) 1922 (32) 1967 (33) 1992 (34) 2044 (35) 2078 (36) 211 (4) 2124 (37) 2155 (38) 2187 (39) 2218 (40) 2249 (41) 2278 (42) 2302 (43) 2327 (44) 2378 (45) 241 (5) 2419 (46) 2474 (47) 2499 (48) 2551 (49) 2587 (50) 2629 (51) 2672 (52) 2725 (53) 277 (6) 2771 (54) 2811 (55) 2858 (56) 2909 (57) 2931 (58) 2958 (59) 3011 (60) 3086 (61) 3123 (62) 3201 (63) 3238 (64) 3262 (65) 3308 (66) 3356 (67) 3396 (68) 3459 (69) 3522 (70) 3582 (71) 436 (8) 452 (9) 528 (10) 542 (11) 558 (12) 574 (13) 801 (14) 827 (15)
$$E	=00000001	3582 (71)	1048 (15) 1103 (16) 1117 (17) 1151 (18) 1171 (19) 1293 (21) 1331 (22) 1368 (23) 1405 (24) 1496 (25) 1566 (26) 1629 (27) 1694 (28) 1758 (29) 179 (2) 1822 (30) 1885 (31) 1962 (32) 1990 (33) 2039 (34) 204 (3) 2073 (35) 2119 (36) 2150 (37) 2181 (38) 2213 (39) 2244 (40) 2273 (41) 2297 (42) 231 (4) 2322 (43) 2344 (44) 2417 (45) 2454 (46) 2497 (47) 2549 (48) 2585 (49) 2627 (50) 2670 (51) 2717 (52) 275 (5) 2764 (53) 2809 (54) 2856 (55) 2907 (56) 2929 (57) 2956 (58) 3009 (59) 3050 (60) 3121 (61) 3178 (62) 3236 (63) 3260 (64) 327 (6) 3306 (65) 3335 (66) 3394 (67) 3457 (68) 3520 (69) 3580 (70) 3620 (71) 447 (8) 494 (9) 540 (10) 556 (11) 572 (12) 777 (13) 820 (14)
$$M	=00000004	3618 (71)	1001 (15) 1010 (15) 1019 (15) 1028 (15) 1037 (15) 1046 (15) 1053 (16) 1098 (16) 1114 (17) 1136 (18) 1148 (18) 1165 (19) 1176 (20) 1208 (20) 1222 (20) 1242 (20) 1249 (21) 1281 (21) 1291 (21) 1329 (22) 1366 (23) 1403 (24) 141 (2) 1410 (25) 1471 (25) 1489 (25) 1544 (26) 1559 (26) 1604 (27) 1622 (27) 1669 (28) 168 (2) 1687 (28) 1733 (29) 1751 (29) 1797 (30) 1815 (30) 1860 (31) 1878 (31) 1890 (32) 1936 (32) 1949 (32) 1960 (32) 1978 (33) 198 (3) 1988 (33) 2018 (34) 2027 (34) 2061 (35) 2071 (35) 2107 (36) 2117 (36) 2138 (37) 2148 (37) 2169 (38) 2179 (38) 2201 (39) 2211 (39) 2232 (40) 224 (4) 2242 (40) 2271 (41) 2295 (42) 2320 (43)

```
                                                  2342   (44)   2349   (45)   2408   (45)   2447   (46)
                                                  2459   (47)   246    (5)    2492   (47)   2537   (48)
                                                  2547   (48)   2573   (49)   2583   (49)   2615   (50)
                                                  2625   (50)   2658   (51)   266    (5)    2668   (51)
                                                  2705   (52)   2715   (52)   2752   (53)   2762   (53)
                                                  2797   (54)   2807   (54)   2854   (55)   2902   (56)
                                                  2927   (57)   2954   (58)   2997   (59)   3007   (59)
                                                  3048   (60)   3055   (61)   3119   (61)   3150   (62)
                                                  3157   (62)   3166   (62)   3183   (63)   319    (6)
                                                  3215   (63)   3255   (64)   3299   (65)   332    (7)
                                                  3329   (66)   3340   (67)   3392   (67)   3424   (68)
                                                  3430   (68)   3442   (68)   3485   (69)   3491   (69)
                                                  3504   (69)   3545   (70)   3551   (70)   3564   (70)
                                                  3603   (71)   3618   (71)   365    (7)    373    (7)
                                                  380    (7)    389    (7)    398    (7)    409    (7)
                                                  416    (8)    445    (8)    462    (9)    472    (9)
                                                  482    (9)    492    (9)    499    (10)   538    (10)
                                                  552    (11)   570    (12)   586    (13)   600    (13)
                                                  614    (13)   627    (13)   640    (13)   653    (13)
                                                  666    (13)   679    (13)   692    (13)   706    (13)
                                                  719    (13)   741    (13)   756    (13)   769    (13)
                                                  775    (13)   782    (14)   817    (14)   839    (15)
                                                  848    (15)   857    (15)   866    (15)   875    (15)
                                                  884    (15)   893    (15)   902    (15)   911    (15)
                                                  920    (15)   929    (15)   938    (15)   947    (15)
                                                  956    (15)   965    (15)   974    (15)   983    (15)
                                                  992    (15)
$$N                    =00000000     3618   (71)   1001   (15)   1010   (15)   1019   (15)   1028   (15)
                                                  1037   (15)   1046   (15)   1098   (16)   1100   (16)
                                                  1114   (17)   1136   (18)   1148   (18)   1165   (19)
                                                  1167   (19)   1208   (20)   1222   (20)   1242   (20)
                                                  1281   (21)   1291   (21)   1329   (22)   1366   (23)
                                                  1403   (24)   1471   (25)   1489   (25)   1494   (25)
                                                  1544   (26)   1559   (26)   1564   (26)   1604   (27)
                                                  1622   (27)   1627   (27)   1669   (28)   168    (2)
                                                  1687   (28)   1692   (28)   1733   (29)   1751   (29)
                                                  1756   (29)   1797   (30)   1815   (30)   1820   (30)
                                                  1860   (31)   1878   (31)   1883   (31)   1936   (32)
                                                  1949   (32)   1960   (32)   1978   (33)   198    (3)
                                                  1988   (33)   2011   (34)   2018   (34)   2027   (34)
                                                 #-2037  (34)   2061   (35)   2071   (35)   2107   (36)
                                                  2117   (36)   2138   (37)   2148   (37)   2169   (38)
                                                  2179   (38)   2201   (39)   2211   (39)   2232   (40)
                                                  224    (4)    2242   (40)   2271   (41)   2295   (42)
                                                  2320   (43)   2342   (44)   2408   (45)   2447   (46)
                                                  246    (5)    2492   (47)   2494   (47)   252    (5)
                                                  2537   (48)   2547   (48)   2573   (49)   2583   (49)
                                                  2615   (50)   2625   (50)   2658   (51)   266    (5)
                                                  2668   (51)   2705   (52)   2715   (52)   272    (5)
                                                  2752   (53)   2762   (53)   2797   (54)   2807   (54)
                                                  2854   (55)   2902   (56)   2927   (57)   2954   (58)
                                                  2997   (59)   3007   (59)   3048   (60)   3119   (61)
                                                  3150   (62)   3157   (62)   3159   (62)   3166   (62)
                                                  3168   (62)   319    (6)    321    (6)    3215   (63)
                                                  3255   (64)   3299   (65)   3329   (66)   3392   (67)
                                                  3424   (68)   3430   (68)   3432   (68)   3442   (68)
                                                  3447   (68)   3485   (69)   3491   (69)   3493   (69)
```

				3504	(69)	3509	(69)	3545	(70)	3551	(70)				
				3553	(70)	3564	(70)	3569	(70)	3603	(71)				
				3618	(71)	365	(7)	373	(7)	380	(7)				
				389	(7)	398	(7)	409	(7)	445	(8)				
				462	(9)	472	(9)	482	(9)	492	(9)				
				538	(10)	552	(11)	570	(12)	586	(13)				
				600	(13)	614	(13)	627	(13)	640	(13)				
				653	(13)	666	(13)	679	(13)	692	(13)				
				706	(13)	719	(13)	741	(13)	756	(13)				
				769	(13)	775	(13)	817	(14)	83	(1)				
				839	(15)	848	(15)	857	(15)	866	(15)				
				875	(15)	884	(15)	893	(15)	902	(15)				
				911	(15)	920	(15)	929	(15)	938	(15)				
				947	(15)	956	(15)	965	(15)	974	(15)				
				983	(15)	992	(15)								
$$S		=FFFFFFFF	3621	(71)	1052	(16)	1053	(16)	1078	(16)	1105	(17)			
					1119	(18)	1153	(19)	1175	(20)	1176	(20)			
					1248	(21)	1249	(21)	1258	(21)	1295	(22)			
					1333	(23)	1370	(24)	140	(2)	1409	(25)			
					141	(2)	1410	(25)	1424	(25)	1498	(26)			
					152	(2)	1568	(27)	1631	(28)	1696	(29)			
					1760	(30)	1824	(31)	185	(3)	1889	(32)			
					1890	(32)	1922	(32)	1967	(33)	1992	(34)			
					2044	(35)	2078	(36)	211	(4)	2124	(37)			
					2155	(38)	2187	(39)	2218	(40)	2249	(41)			
					2278	(42)	2302	(43)	2327	(44)	2348	(45)			
					2349	(45)	2378	(45)	241	(5)	2419	(46)			
					2458	(47)	2459	(47)	2474	(47)	2499	(48)			
					2551	(49)	2587	(50)	2629	(51)	2672	(52)			
					2725	(53)	277	(6)	2771	(54)	2811	(55)			
					2858	(56)	2909	(57)	2931	(58)	2958	(59)			
					3011	(60)	3054	(61)	3055	(61)	3086	(61)			
					3123	(62)	3182	(63)	3183	(63)	3201	(63)			
					3238	(64)	3262	(65)	3308	(66)	331	(7)			
					332	(7)	3339	(67)	3340	(67)	3356	(67)			
					3396	(68)	3459	(69)	3522	(70)	3582	(71)			
					415	(8)	416	(8)	436	(8)	452	(9)			
					498	(10)	499	(10)	528	(10)	542	(11)			
					558	(12)	574	(13)	781	(14)	782	(14)			
					801	(14)	827	(15)							
$ENV		=00000001	137	(1)											
$ER		=FFFFFFFF	3621	(71)	1078	(16)	1105	(17)	1119	(18)	1153	(19)			
					1258	(21)	1295	(22)	1333	(23)	1370	(24)			
					1424	(25)	#-1471	(25)	#-1489	(25)	1498	(26)			
					152	(2)	#-1544	(26)	#-1559	(26)	1568	(27)			
					#-1604	(27)	#-1622	(27)	1631	(28)	#-1669	(28)			
					#-1687	(28)	1696	(29)	#-1733	(29)	#-1751	(29)			
					1760	(30)	#-1797	(30)	#-1815	(30)	1824	(31)			
					185	(3)	#-1860	(31)	#-1878	(31)	1922	(32)			
					1967	(33)	1992	(34)	2044	(35)	2078	(36)			
					211	(4)	2124	(37)	2155	(38)	2187	(39)			
					2218	(40)	2249	(41)	2278	(42)	2302	(43)			
					2327	(44)	2378	(45)	241	(5)	2419	(46)			
					2474	(47)	2499	(48)	2551	(49)	2587	(50)			
					2629	(51)	2672	(52)	2725	(53)	277	(6)			
					2771	(54)	2811	(55)	2858	(56)	2909	(57)			
					2931	(58)	2958	(59)	3011	(60)	3086	(61)			

```
                                                3123    (62)    3201    (63)    3238    (64)    3262    (65)
                                                3308    (66)    3356    (67)    3396    (68)    3459    (69)
                                                3522    (70)    3582    (71)    436     (8)     452     (9)
                                                528     (10)    542     (11)    558     (12)    574     (13)
                                                801     (14)    827     (15)
$MO                     =00000001       3622    (71)    3622    (71)
$S$                     =00000324-R     3582    (71)    1048    (15)    1078    (16)    1103    (16)    1105    (17)
                                                1117    (17)    1119    (18)    1151    (18)    1153    (19)
                                                1171    (19)    1258    (21)    1293    (21)    1295    (22)
                                                1331    (22)    1333    (23)    1368    (23)    1370    (24)
                                                1405    (24)    1424    (25)    1496    (25)    1498    (26)
                                                152     (2)     1566    (26)    1568    (27)    1629    (27)
                                                1631    (28)    1694    (28)    1696    (29)    1758    (29)
                                                1760    (30)    179     (2)     1822    (30)    1824    (31)
                                                185     (3)     1885    (31)    1922    (32)    1962    (32)
                                                1967    (33)    1990    (33)    1992    (34)    2039    (34)
                                                204     (3)     2044    (35)    2073    (35)    2078    (36)
                                                211     (4)     2119    (36)    2124    (37)    2150    (37)
                                                2155    (38)    2181    (38)    2187    (39)    2213    (39)
                                                2218    (40)    2244    (40)    2249    (41)    2273    (41)
                                                2278    (42)    2297    (42)    2302    (43)    231     (4)
                                                2322    (43)    2327    (44)    2344    (44)    2378    (45)
                                                241     (5)     2417    (45)    2419    (46)    2454    (46)
                                                2474    (47)    2497    (47)    2499    (48)    2549    (48)
                                                2551    (49)    2585    (49)    2587    (50)    2627    (50)
                                                2629    (51)    2670    (51)    2672    (52)    2717    (52)
                                                2725    (53)    275     (5)     2764    (53)    277     (6)
                                                2771    (54)    2809    (54)    2811    (55)    2856    (55)
                                                2858    (56)    2907    (56)    2909    (57)    2929    (57)
                                                2931    (58)    2956    (58)    2958    (59)    3009    (59)
                                                3011    (60)    3050    (60)    3086    (61)    3121    (61)
                                                3123    (62)    3178    (62)    3201    (63)    3236    (63)
                                                3238    (64)    3260    (64)    3262    (65)    327     (6)
                                                3306    (65)    3308    (66)    3335    (66)    3356    (67)
                                                3394    (67)    3396    (68)    3457    (68)    3459    (69)
                                                3520    (69)    3522    (70)    3580    (70)    3582    (71)
                                                3620    (71)    436     (8)     447     (8)     452     (9)
                                                494     (9)     528     (10)    540     (10)    542     (11)
                                                556     (11)    558     (12)    572     (12)    574     (13)
                                                777     (13)    801     (14)    820     (14)    827     (15)
$ST                     =00000000       3621    (71)    1001    (15)    1002    (15)    1010    (15)    1011    (15)
                                                1019    (15)    1020    (15)    1028    (15)    1029    (15)
                                                1037    (15)    1038    (15)    1046    (15)    1047    (15)
                                                1048    (15)    1050    (16)    1053    (16)    1078    (16)
                                                1098    (16)    1103    (16)    1105    (17)    1114    (17)
                                                1115    (17)    1117    (17)    1119    (18)    1136    (18)
                                                1137    (18)    1148    (18)    1149    (18)    1151    (18)
                                                1153    (19)    1165    (19)    1168    (19)    1171    (19)
                                                1173    (20)    1176    (20)    1208    (20)    1209    (20)
                                                1222    (20)    1223    (20)    1242    (20)    1243    (20)
                                                1245    (20)    1249    (21)    1258    (21)    1281    (21)
                                                1282    (21)    1291    (21)    1292    (21)    1293    (21)
                                                1295    (22)    1329    (22)    1330    (22)    1331    (22)
                                                1333    (23)    1366    (23)    1367    (23)    1368    (23)
                                                1370    (24)    1403    (24)    1404    (24)    1405    (24)
                                                1407    (25)    141     (2)     1410    (25)    1424    (25)
                                                1471    (25)    1472    (25)    1489    (25)    1490    (25)
```

1496 (25) 1498 (26) 152 (2) 1544 (26)
1545 (26) 1559 (26) 1560 (26) 1566 (26)
1568 (27) 1604 (27) 1605 (27) 1622 (27)
1623 (27) 1629 (27) 1631 (28) 1669 (28)
1670 (28) 168 (2) 1687 (28) 1688 (28)
1694 (28) 1696 (29) 171 (2) 1733 (29)
1734 (29) 1751 (29) 1752 (29) 1758 (29)
1760 (30) 179 (2) 1797 (30) 1798 (30)
1815 (30) 1816 (30) 1822 (30) 1824 (31)
185 (3) 1860 (31) 1861 (31) 1878 (31)
1879 (31) 1885 (31) 1887 (32) 1890 (32)
1922 (32) 1936 (32) 1937 (32) 1949 (32)
1951 (32) 1960 (32) 1961 (32) 1962 (32)
1967 (33) 1978 (33) 1979 (33) 198 (3)
1988 (33) 1989 (33) 1990 (33) 1992 (34)
201 (3) 2018 (34) 2019 (34) 2027 (34)
2028 (34) 2039 (34) 204 (3) 2044 (35)
2061 (35) 2062 (35) 2071 (35) 2072 (35)
2073 (35) 2078 (36) 2107 (36) 2108 (36)
211 (4) 2117 (36) 2118 (36) 2119 (36)
2124 (37) 2138 (37) 2139 (37) 2148 (37)
2149 (37) 2150 (37) 2155 (38) 2169 (38)
2170 (38) 2179 (38) 2180 (38) 2181 (38)
2187 (39) 2201 (39) 2202 (39) 2211 (39)
2212 (39) 2213 (39) 2218 (40) 2232 (40)
2233 (40) 224 (4) 2242 (40) 2243 (40)
2244 (40) 2249 (41) 227 (4) 2271 (41)
2272 (41) 2273 (41) 2278 (42) 2295 (42)
2296 (42) 2297 (42) 2302 (43) 231 (4)
2320 (43) 2321 (43) 2322 (43) 2327 (44)
2342 (44) 2343 (44) 2344 (44) 2346 (45)
2349 (45) 2378 (45) 2408 (45) 2409 (45)
241 (5) 2417 (45) 2419 (46) 2447 (46)
2448 (46) 2454 (46) 2456 (47) 2459 (47)
246 (5) 2474 (47) 2492 (47) 2495 (47)
2497 (47) 2499 (48) 253 (5) 2537 (48)
2538 (48) 2547 (48) 2548 (48) 2549 (48)
2551 (49) 2573 (49) 2574 (49) 2583 (49)
2584 (49) 2585 (49) 2587 (50) 2615 (50)
2616 (50) 2625 (50) 2626 (50) 2627 (50)
2629 (51) 2658 (51) 2659 (51) 266 (5)
2668 (51) 2669 (51) 2670 (51) 2672 (52)
2705 (52) 2706 (52) 2715 (52) 2716 (52)
2717 (52) 2725 (53) 273 (5) 275 (5)
2752 (53) 2753 (53) 2762 (53) 2763 (53)
2764 (53) 277 (6) 2771 (54) 2797 (54)
2798 (54) 2807 (54) 2808 (54) 2809 (54)
2811 (55) 2854 (55) 2855 (55) 2856 (55)
2858 (56) 2902 (56) 2906 (56) 2907 (56)
2909 (57) 2927 (57) 2928 (57) 2929 (57)
2931 (58) 2954 (58) 2955 (58) 2956 (58)
2958 (59) 2997 (59) 2998 (59) 3007 (59)
3008 (59) 3009 (59) 3011 (60) 3048 (60)
3049 (60) 3050 (60) 3052 (61) 3055 (61)
3086 (61) 3119 (61) 3120 (61) 3121 (61)
3123 (62) 3150 (62) 3151 (62) 3157 (62)
3160 (62) 3166 (62) 3169 (62) 3178 (62)

$TN =0000000F 3622 (71)

3180 (63) 3183 (63) 319 (6) 3201 (63)
3215 (63) 3216 (63) 3236 (63) 3238 (64)
3255 (64) 3256 (64) 3260 (64) 3262 (65)
327 (6) 329 (7) 3299 (65) 3300 (65)
3306 (65) 3308 (66) 332 (7) 3329 (66)
3330 (66) 3335 (66) 3337 (67) 3340 (67)
3356 (67) 3392 (67) 3393 (67) 3394 (67)
3396 (68) 3424 (68) 3430 (68) 3436 (68)
3442 (68) 3448 (68) 3457 (68) 3459 (69)
3485 (69) 3491 (69) 3499 (69) 3504 (69)
3510 (69) 3520 (69) 3522 (70) 3545 (70)
3551 (70) 3559 (70) 3564 (70) 3570 (70)
3580 (70) 3582 (71) 3603 (71) 3604 (71)
3618 (71) 3619 (71) 3620 (71) 3621 (71)
365 (7) 366 (7) 373 (7) 374 (7)
380 (7) 381 (7) 389 (7) 390 (7)
398 (7) 399 (7) 409 (7) 410 (7)
412 (7) 416 (8) 436 (8) 445 (8)
446 (8) 447 (8) 452 (9) 462 (9)
463 (9) 472 (9) 473 (9) 482 (9)
483 (9) 492 (9) 493 (9) 494 (9)
496 (10) 499 (10) 528 (10) 538 (10)
539 (10) 540 (10) 542 (11) 552 (11)
554 (11) 555 (11) 556 (11) 558 (12)
570 (12) 571 (12) 572 (12) 574 (13)
586 (13) 587 (13) 600 (13) 601 (13)
614 (13) 615 (13) 627 (13) 628 (13)
640 (13) 641 (13) 653 (13) 654 (13)
666 (13) 667 (13) 679 (13) 680 (13)
692 (13) 693 (13) 706 (13) 707 (13)
719 (13) 720 (13) 741 (13) 742 (13)
756 (13) 757 (13) 769 (13) 770 (13)
775 (13) 776 (13) 777 (13) 779 (14)
782 (14) 801 (14) 817 (14) 818 (14)
820 (14) 827 (15) 839 (15) 840 (15)
848 (15) 849 (15) 857 (15) 858 (15)
866 (15) 867 (15) 875 (15) 876 (15)
884 (15) 885 (15) 893 (15) 894 (15)
902 (15) 903 (15) 911 (15) 912 (15)
920 (15) 921 (15) 929 (15) 930 (15)
938 (15) 939 (15) 947 (15) 948 (15)
956 (15) 957 (15) 965 (15) 966 (15)
974 (15) 975 (15) 983 (15) 984 (15)
992 (15) 993 (15)
1001 (15) 1010 (15) 1019 (15) 1028 (15)
1037 (15) 1046 (15) 1050 (16) 1052 (16)
1053 (16) 1098 (16) 1114 (17) 1136 (18)
1148 (18) 1165 (19) 1173 (20) 1175 (20)
1176 (20) 1208 (20) 1222 (20) 1242 (20)
1245 (20) 1248 (21) 1249 (21) 1281 (21)
1291 (21) 1329 (22) 1366 (23) 140 (2)
1403 (24) 1407 (25) 1409 (25) 141 (2)
1410 (25) 1471 (25) 1489 (25) 1544 (26)
1559 (26) 1604 (27) 1622 (27) 1669 (28)
168 (2) 1687 (28) 1733 (29) 1751 (29)
1797 (30) 1815 (30) 1860 (31) 1878 (31)
1887 (32) 1889 (32) 1890 (32) 1936 (32)

ABORT_IF_FAIL	00000000-XR			
ALL	=00000001	83	(1)	
ARRAY_MSG1	00000000-XR			
ARRAY_MSG10	00000000-XR			
ARRAY_MSG11	00000000-XR			
ARRAY_MSG12	00000000-XR			
ARRAY_MSG13	00000000-XR			
ARRAY_MSG14	00000000-XR			
ARRAY_MSG15	00000000-XR			
ARRAY_MSG16	00000000-XR			
ARRAY_MSG17	00000000-XR			
ARRAY_MSG18	00000000-XR			
ARRAY_MSG19	00000000-XR			

1949 (32) 1960 (32) 1978 (33) 198 (3)
1988 (33) 2018 (34) 2027 (34) 2061 (35)
2071 (35) 2107 (36) 2117 (36) 2138 (37)
2148 (37) 2169 (38) 2179 (38) 2201 (39)
2211 (39) 2232 (40) 224 (4) 2242 (40)
2271 (41) 2295 (42) 2320 (43) 2342 (44)
2346 (45) 2348 (45) 2349 (45) 2408 (45)
2447 (46) 2456 (47) 2458 (47) 2459 (47)
246 (5) 2492 (47) 2537 (48) 2547 (48)
2573 (49) 2583 (49) 2615 (50) 2625 (50)
2658 (51) 266 (5) 2668 (51) 2705 (52)
2715 (52) 2752 (53) 2762 (53) 2797 (54)
2807 (54) 2854 (55) 2902 (56) 2927 (57)
2954 (58) 2997 (59) 3007 (59) 3048 (60)
3052 (61) 3054 (61) 3055 (61) 3119 (61)
3150 (62) 3157 (62) 3166 (62) 3180 (63)
3182 (63) 3183 (63) 319 (6) 3215 (63)
3255 (64) 329 (7) 3299 (65) 331 (7)
332 (7) 3329 (66) 3337 (67) 3339 (67)
3340 (67) 3392 (67) 3424 (68) 3430 (68)
3442 (68) 3485 (69) 3491 (69) 3504 (69)
3545 (70) 3551 (70) 3564 (70) 3603 (71)
3618 (71) 3621 (71) 3622 (71) 365 (7)
373 (7) 380 (7) 389 (7) 398 (7)
409 (7) 412 (7) 415 (8) 416 (8)
445 (8) 462 (9) 472 (9) 482 (9)
492 (9) 496 (10) 498 (10) 499 (10)
538 (10) 552 (11) 570 (12) 586 (13)
600 (13) 614 (13) 627 (13) 640 (13)
653 (13) 666 (13) 679 (13) 692 (13)
706 (13) 719 (13) 741 (13) 756 (13)
769 (13) 775 (13) 779 (14) 781 (14)
782 (14) 817 (14) 839 (15) 848 (15)
857 (15) 866 (15) 875 (15) 884 (15)
893 (15) 902 (15) 911 (15) 920 (15)
929 (15) 938 (15) 947 (15) 956 (15)
965 (15) 974 (15) 983 (15) 992 (15)
553 (11)
1053 (16) 1176 (20) 1249 (21) 141 (2)
1410 (25) 1890 (32) 2349 (45) 2459 (47)
3055 (61) 3183 (63) 332 (7) 3340 (67)
416 (8) 499 (10) 782 (14)
1489 (25) 266 (5) 3157 (62) 3166 (62)
3442 (68) 3504 (69) 3564 (70) 365 (7)
380 (7) 538 (10)
653 (13) 902 (15) 920 (15)
938 (15) 956 (15)
1028 (15) 1148 (18) 974 (15) 992 (15)
2408 (45) 2705 (52) 817 (14)
1114 (17) 1208 (20) 1222 (20) 2018 (34)
224 (4) 246 (5) 2954 (58) 3215 (63)
398 (7) 409 (7)
2537 (48) 3007 (59) 3048 (60)
1019 (15) 1136 (18)
1001 (15)
983 (15)
1037 (15) 911 (15) 929 (15) 947 (15)

```
                                                 965    (15)
ARRAY_MSG2                       00000000-XR     1366   (23)     1403   (24)     570    (12)
ARRAY_MSG20                      00000000-XR     741    (13)
ARRAY_MSG21                      00000000-XR     586    (13)
ARRAY_MSG22                      00000000-XR     168    (2)
ARRAY_MSG23                      00000000-XR     389    (7)
ARRAY_MSG24                      00000000-XR     1559   (26)     1622   (27)     1687   (28)     1751   (29)
                                                 1815   (30)     1878   (31)     3255   (64)     3299   (65)
                                                 3329   (66)
ARRAY_MSG26                      00000000-XR     839    (15)     857    (15)     875    (15)
ARRAY_MSG27                      00000000-XR     893    (15)
ARRAY_MSG29                      00000000-XR     1329   (22)
ARRAY_MSG3                       00000000-XR     1988   (33)     2807   (54)
ARRAY_MSG31                      00000000-XR     482    (9)
ARRAY_MSG32                      00000000-XR     1242   (20)     2573   (49)
ARRAY_MSG33                      00000000-XR     706    (13)     719    (13)
ARRAY_MSG34                      00000000-XR     2997   (59)
ARRAY_MSG35                      00000000-XR     462    (9)      472    (9)
ARRAY_MSG36                      00000000-XR     2138   (37)
ARRAY_MSG37                      00000000-XR     2615   (50)     2658   (51)     492    (9)
ARRAY_MSG38                      00000000-XR     848    (15)
ARRAY_MSG39                      00000000-XR     692    (13)
ARRAY_MSG4                       00000000-XR     1960   (32)     2071   (35)     2447   (46)     2583   (49)
                                                 2625   (50)     2668   (51)     2715   (52)     2752   (53)
                                                 2762   (53)     3430   (68)     3491   (69)     3551   (70)
ARRAY_MSG40                      00000000-XR     756    (13)
ARRAY_MSG41                      00000000-XR     2061   (35)     866    (15)
ARRAY_MSG42                      00000000-XR     2854   (55)     640    (13)
ARRAY_MSG43                      00000000-XR     627    (13)
ARRAY_MSG44                      00000000-XR     614    (13)
ARRAY_MSG45                      00000000-XR     600    (13)
ARRAY_MSG46                      00000000-XR     1978   (33)
ARRAY_MSG47                      00000000-XR     2492   (47)
ARRAY_MSG48                      00000000-XR     2797   (54)
ARRAY_MSG49                      00000000-XR     2902   (56)
ARRAY_MSG5                       00000000-XR     2027   (34)     2169   (38)     2179   (38)     2201   (39)
                                                 2211   (39)     2242   (40)     2295   (42)
ARRAY_MSG50                      00000000-XR     2107   (36)
ARRAY_MSG51                      00000000-XR     1949   (32)
ARRAY_MSG52                      00000000-XR     1936   (32)
ARRAY_MSG53                      00000000-XR     1281   (21)
ARRAY_MSG56                      00000000-XR     373    (7)
ARRAY_MSG57                      00000000-XR     1291   (21)     445    (8)
ARRAY_MSG6                       00000000-XR     2117   (36)     2148   (37)     2271   (41)     2547   (48)
ARRAY_MSG60                      00000000-XR     198    (3)      2232   (40)     2927   (57)
ARRAY_MSG61                      00000000-XR     3618   (71)
ARRAY_MSG62                      00000000-XR     1471   (25)     1544   (26)     1604   (27)     1669   (28)
                                                 1733   (29)     1797   (30)     1860   (31)     3119   (61)
                                                 3150   (62)     3392   (67)     3424   (68)     3485   (69)
                                                 3545   (70)     3603   (71)
ARRAY_MSG7                       00000000-XR     1010   (15)     1046   (15)     775    (13)
ARRAY_MSG8                       00000000-XR     552    (11)     769    (13)
ARRAY_MSG9                       00000000-XR     666    (13)     679    (13)     884    (15)
ASR_OFFSET                       00000000-XR   #-2922   (57)
ATTENTION                        00000000-XR   #-1970   (33)
BASE_001                         00000000-R      140    (2)      141    (2)
BASE_002                         00000000-R      331    (7)      332    (7)
```

BASE_003	00000000-R	415	(8)	416	(8)						
BASE_004	00000000-R	498	(10)	499	(10)						
BASE_005	00000000-R	781	(14)	782	(14)						
BASE_006	00000000-R	1052	(16)	1053	(16)						
BASE_007	00000000-R	1175	(20)	1176	(20)						
BASE_008	00000000-R	1248	(21)	1249	(21)						
BASE_009	00000000-R	1409	(25)	1410	(25)						
BASE_010	00000000-R	1889	(32)	1890	(32)						
BASE_011	00000000-R	2348	(45)	2349	(45)						
BASE_012	00000000-R	2458	(47)	2459	(47)						
BASE_013	00000000-R	3054	(61)	3055	(61)						
BASE_014	00000000-R	3182	(63)	3183	(63)						
BASE_015	00000000-R	3339	(67)	3340	(67)						
BCR	00000000-XR			#-1310	(22)	#-1347	(23)	#-1384	(24)	#-1446	(25)
				#-1519	(26)	#-1580	(27)	#-1645	(28)	#-1709	(29)
				#-1773	(30)	#-1836	(31)	#-2086	(36)	#-2161	(38)
				#-2252	(41)	#-2394	(45)	#-2435	(46)	#-2516	(48)
				#-2524	(48)	#-2602	(50)	#-2643	(51)	#-2683	(52)
				#-2691	(52)	#-2730	(53)	#-2736	(53)	#-2781	(54)
				#-2830	(55)	#-2877	(56)	#-2976	(59)	#-2999	(59)
				#-3031	(60)	#-3100	(61)	#-3136	(62)	#-3209	(63)
				#-3210	(63)	#-3243	(64)	#-3248	(64)	#-3278	(65)
				#-3293	(65)	#-3314	(66)	#-3321	(66)	#-3375	(67)
				#-3408	(68)	#-3470	(69)	#-3530	(70)	#-3590	(71)
				#-3605	(71)	#-392	(7)	#-393	(7)	#-394	(7)
BIT...	=00000002	137	(1)	137	(1)						
BIT0	=00000001	133	(1)	#-2397	(45)						
BIT1	=00000002	133	(1)	#-2397	(45)						
BIT10	=00000400	133	(1)								
BIT11	=00000800	133	(1)								
BIT12	=00001000	133	(1)	#-2484	(47)						
BIT13	=00002000	133	(1)								
BIT14	=00004000	133	(1)								
BIT15	=00008000	133	(1)								
BIT16	=00010000	133	(1)	#-3207	(63)						
BIT17	=00020000	133	(1)								
BIT18	=00040000	133	(1)								
BIT19	=00080000	133	(1)								
BIT2	=00000004	133	(1)								
BIT20	=00100000	133	(1)								
BIT21	=00200000	133	(1)								
BIT22	=00400000	133	(1)								
BIT23	=00800000	133	(1)								
BIT24	=01000000	133	(1)								
BIT25	=02000000	133	(1)								
BIT26	=04000000	133	(1)								
BIT27	=08000000	133	(1)								
BIT28	=10000000	133	(1)								
BIT29	=20000000	133	(1)								
BIT3	=00000008	133	(1)	#-1323	(22)	#-1360	(23)	#-1397	(24)		
BIT30	=40000000	133	(1)								
BIT31	=80000000	133	(1)	#-2380	(45)						
BIT4	=00000010	133	(1)								
BIT5	=00000020	133	(1)								
BIT6	=00000040	133	(1)								
BIT7	=00000080	133	(1)								
BIT8	=00000100	133	(1)	#-3411	(68)						

```
BIT9                            =00000200      133    (1)
BLKSND_COMD                      00000000-XR
                                                      #-1315   (22)   #-1319   (22)   #-1352   (23)   #-1356   (23)
                                                      #-1389   (24)   #-1393   (24)   #-2094   (36)   #-2253   (41)
                                                      #-2437   (46)   #-2981   (59)   #-2988   (59)   #-631    (13)
BYTE0                            00000000-XR
                                                      #-1235   (20)   #-1275   (21)   #-1286   (21)   #-2949   (58)
                                                      #-376    (7)    #-531    (10)   #-546    (11)   #-564    (12)
                                                      #-580    (13)   #-593    (13)   #-607    (13)   #-621    (13)
                                                      #-634    (13)   #-647    (13)   #-660    (13)   #-673    (13)
                                                      #-686    (13)   #-700    (13)   #-714    (13)   #-736    (13)
                                                      #-750    (13)   #-763    (13)
BYTE1                            00000000-XR
                                                      #-1235   (20)   #-1286   (21)   #-1323   (22)   #-1360   (23)
                                                      #-1397   (24)   #-2949   (58)   #-376    (7)    #-531    (10)
                                                      #-546    (11)   #-564    (12)   #-580    (13)   #-593    (13)
                                                      #-607    (13)   #-621    (13)   #-634    (13)   #-647    (13)
                                                      #-660    (13)   #-673    (13)   #-686    (13)   #-700    (13)
                                                      #-714    (13)   #-750    (13)   #-763    (13)
BYTE2                            00000000-XR
                                                      #-1323   (22)   #-1360   (23)   #-1397   (24)   #-3000   (59)
                                                      #-3249   (64)
BYTE3                            00000000-XR
                                                      #-1323   (22)   #-1360   (23)   #-1397   (24)   #-2397   (45)
                                                      #-3000   (59)   #-3162   (62)   #-3249   (64)
CAR                              00000000-XR
                                                      #-1318   (22)   #-1355   (23)   #-1392   (24)   #-2396   (45)
CEP_FUNCTIONAL                  =00000000      137    (1)
CEP_REPAIR                      =00000001      137    (1)     137    (1)
COMP_MSG1                        00000000-XR            2320   (43)
COMP_MSG2                        00000000-XR            2342   (44)
CONT_PC                          00000000-XR
                                                      #-160    (2)    #-189    (3)    #-215    (4)    #-2487   (47)
CR                               00000000-XR
                                                      #-1076   (16)   #-1197   (20)   #-1198   (20)   #-1231   (20)
                                                      #-1232   (20)   #-1270   (21)   #-1271   (21)   #-1283   (21)
                                                      #-1303   (22)   #-1311   (22)   #-1340   (23)   #-1348   (23)
                                                      #-1377   (24)   #-1385   (24)   #-1434   (25)   #-1448   (25)
                                                      #-1505   (26)   #-1521   (26)   #-1577   (27)   #-1578   (27)
                                                      #-1641   (28)   #-1642   (28)   #-1706   (29)   #-1707   (29)
                                                      #-1769   (30)   #-1770   (30)   #-1833   (31)   #-1834   (31)
                                                      #-187    (3)    #-1924   (32)   #-1925   (32)   #-1938   (32)
                                                      #-1968   (33)   #-1969   (33)   #-2003   (34)   #-2047   (35)
                                                      #-2048   (35)   #-2079   (36)   #-2080   (36)   #-2125   (37)
                                                      #-2126   (37)   #-213    (4)    #-2156   (38)   #-2157   (38)
                                                      #-2188   (39)   #-2189   (39)   #-2219   (40)   #-2220   (40)
                                                      #-2250   (41)   #-2251   (41)   #-2279   (42)   #-2280   (42)
                                                      #-2285   (42)   #-2286   (42)   #-2289   (42)   #-2303   (43)
                                                      #-2304   (43)   #-2328   (44)   #-2329   (44)   #-2377   (45)
                                                      #-2391   (45)   #-242    (5)    #-2432   (46)   #-2436   (46)
                                                      #-2506   (48)   #-2507   (48)   #-254    (5)    #-2558   (49)
                                                      #-2559   (49)   #-257    (5)    #-2593   (50)   #-2594   (50)
                                                      #-2635   (51)   #-2636   (51)   #-2679   (52)   #-2680   (52)
                                                      #-2726   (53)   #-2727   (53)   #-274    (5)    #-2772   (54)
                                                      #-2773   (54)   #-2822   (55)   #-2824   (55)   #-2869   (56)
                                                      #-2871   (56)   #-2919   (57)   #-2920   (57)   #-2941   (58)
                                                      #-2942   (58)   #-2967   (59)   #-2968   (59)   #-3020   (60)
                                                      #-3021   (60)   #-3089   (61)   #-3101   (61)   #-3107   (61)
                                                      #-3127   (62)   #-3205   (63)   #-3240   (64)   #-3241   (64)
                                                      #-3263   (65)   #-3275   (65)   #-3276   (65)   #-3310   (66)
                                                      #-3311   (66)   #-3362   (67)   #-3363   (67)   #-3381   (67)
                                                      #-3400   (68)   #-357    (7)    #-369    (7)    #-437    (8)
                                                      #-438    (8)    #-439    (8)    #-440    (8)    #-453    (9)
                                                      #-455    (9)    #-456    (9)    #-457    (9)    #-464    (9)
                                                      #-466    (9)    #-467    (9)    #-474    (9)    #-476    (9)
```

```
                                                #-477    (9)    #-484    (9)    #-486    (9)    #-487    (9)
                                                #-529    (10)   #-543    (11)   #-544    (11)   #-559    (12)
                                                #-560    (12)   #-575    (13)   #-576    (13)   #-588    (13)
                                                #-589    (13)   #-602    (13)   #-603    (13)   #-616    (13)
                                                #-617    (13)   #-629    (13)   #-630    (13)   #-642    (13)
                                                #-643    (13)   #-655    (13)   #-656    (13)   #-668    (13)
                                                #-669    (13)   #-681    (13)   #-682    (13)   #-694    (13)
                                                #-695    (13)   #-708    (13)   #-709    (13)   #-721    (13)
                                                #-725    (13)   #-726    (13)   #-745    (13)   #-746    (13)
                                                #-758    (13)   #-759    (13)   #-803    (14)   #-830    (15)
CRTED_READ_DATA                 00000000-XR     #-2832   (55)
CSR                             00000000-XR     #-243    (5)
DATA_001                        00000030-R  141    (2)     141    (2)
DATA_002                        00000018-R  332    (7)     332    (7)
DATA_003                        0000001C-R  416    (8)     416    (8)
DATA_004                        0000001C-R  499    (10)    499    (10)
DATA_005                        00000030-R  782    (14)    782    (14)
DATA_006                        00000018-R  1053   (16)    1053   (16)
DATA_007                        00000024-R  1176   (20)    1176   (20)
DATA_008                        00000018-R  1249   (21)    1249   (21)
DATA_009                        00000024-R  1410   (25)    1410   (25)
DATA_010                        00000024-R  1890   (32)    1890   (32)
DATA_011                        00000018-R  2349   (45)    2349   (45)
DATA_012                        00000024-R  2459   (47)    2459   (47)
DATA_013                        00000018-R  3055   (61)    3055   (61)
DATA_014                        00000014-R  3183   (63)    3183   (63)
DATA_015                        00000020-R  3340   (67)    3340   (67)
DATA_IBUFFER                    00000000-XR     #-1437   (25)   #-1456   (25)    1477    (25)   #-1479   (25)
                                                #-1509   (26)   #-1529   (26)    1550    (26)   #-1552   (26)
                                                #-1574   (27)   #-1589   (27)    1610    (27)   #-1612   (27)
                                                #-1638   (28)   #-1654   (28)    1675    (28)   #-1677   (28)
                                                #-1703   (29)   #-1718   (29)    1739    (29)   #-1741   (29)
                                                #-1767   (30)   #-1782   (30)    1803    (30)   #-1805   (30)
                                                #-1831   (31)   #-1845   (31)    1866    (31)   #-1868   (31)
                                                 3221    (63)
DATA_OBUFFER                    00000000-XR     #-1436   (25)    1439    (25)    1476    (25)   #-1478   (25)
                                                #-1508   (26)    1512    (26)    1549    (26)   #-1551   (26)
                                                #-1573   (27)    1609    (27)   #-1611   (27)   #-1637   (28)
                                                 1674    (28)   #-1676   (28)   #-1702   (29)    1738    (29)
                                                #-1740   (29)   #-1766   (30)    1802    (30)   #-1804   (30)
                                                #-1830   (31)    1865    (31)   #-1867   (31)    3230    (63)
DATA_XFER_ABRT                  00000000-XR     #-1941   (32)   #-2052   (35)   #-2089   (36)   #-2264   (41)
                                                #-2308   (43)   #-2529   (48)   #-2607   (50)   #-2649   (51)
                                                #-2697   (52)   #-2744   (53)   #-2787   (54)   #-2879   (56)
                                                #-2979   (59)   #-3040   (60)   #-3112   (61)   #-3386   (67)
DATA_XFER_DONE                  00000000-XR     #-1464   (25)   #-1537   (26)   #-1597   (27)   #-1662   (28)
                                                #-1726   (29)   #-1790   (30)   #-1853   (31)   #-1942   (32)
                                                #-2051   (35)   #-2090   (36)   #-2264   (41)   #-2307   (43)
                                                #-2530   (48)   #-2608   (50)   #-2651   (51)   #-2698   (52)
                                                #-2745   (53)   #-2788   (54)   #-2833   (55)   #-2880   (56)
                                                #-2979   (59)   #-3041   (60)   #-3111   (61)   #-3144   (62)
                                                #-3147   (62)   #-3385   (67)   #-3417   (68)   #-3478   (69)
                                                #-3538   (70)   #-3597   (71)   #-3611   (71)
DATA_XFER_LATE                  00000000-XR     #-2088   (36)   #-2263   (41)
DCM                             00000000-XR     #-2840   (55)   #-2888   (56)
DEFAULT                        =00000000    83     (1)     1053    (16)    1176    (20)    1249    (21)     141    (2)
                                                 1410    (25)    1890    (32)    2349    (45)    2459    (47)
```

		3055	(61)	3183	(63)	332	(7)	3340	(67)
		416	(8)	499	(10)	782	(14)		
DR	00000000-XR	#-1275	(21)	#-1276	(21)	#-1286	(21)	#-1314	(22)
		#-1315	(22)	#-1316	(22)	#-1317	(22)	#-1319	(22)
		#-1320	(22)	#-1321	(22)	#-1351	(23)	#-1352	(23)
		#-1353	(23)	#-1354	(23)	#-1356	(23)	#-1357	(23)
		#-1358	(23)	#-1388	(24)	#-1389	(24)	#-1390	(24)
		#-1391	(24)	#-1393	(24)	#-1394	(24)	#-1395	(24)
		#-1450	(25)	#-1453	(25)	#-1454	(25)	#-1455	(25)
		#-1456	(25)	#-1460	(25)	#-1461	(25)	#-1523	(26)
		#-1526	(26)	#-1527	(26)	#-1528	(26)	#-1529	(26)
		#-1533	(26)	#-1534	(26)	#-1583	(27)	#-1586	(27)
		#-1587	(27)	#-1588	(27)	#-1589	(27)	#-1593	(27)
		#-1594	(27)	#-1648	(28)	#-1651	(28)	#-1652	(28)
		#-1653	(28)	#-1654	(28)	#-1658	(28)	#-1659	(28)
		#-1712	(29)	#-1715	(29)	#-1716	(29)	#-1717	(29)
		#-1718	(29)	#-1722	(29)	#-1723	(29)	#-1776	(30)
		#-1779	(30)	#-1780	(30)	#-1781	(30)	#-1782	(30)
		#-1786	(30)	#-1787	(30)	#-1839	(31)	#-1842	(31)
		#-1843	(31)	#-1844	(31)	#-1845	(31)	#-1849	(31)
		#-1850	(31)	#-1928	(32)	#-1939	(32)	#-1940	(32)
		#-1971	(33)	#-1973	(33)	#-2092	(36)	#-2094	(36)
		#-2096	(36)	#-2097	(36)	#-2099	(36)	#-2100	(36)
		#-2129	(37)	#-2160	(38)	#-2192	(39)	#-2223	(40)
		#-2253	(41)	#-2257	(41)	#-2258	(41)	#-2259	(41)
		#-2261	(41)	#-2262	(41)	#-2283	(42)	#-2312	(43)
		#-2334	(44)	#-2437	(46)	#-2439	(46)	#-2508	(48)
		#-2522	(48)	#-2560	(49)	#-2566	(49)	#-258	(5)
		#-259	(5)	#-2595	(50)	#-2842	(55)	#-2843	(55)
		#-2844	(55)	#-2845	(55)	#-2846	(55)	#-2890	(56)
		#-2891	(56)	#-2892	(56)	#-2893	(56)	#-2894	(56)
		#-2921	(57)	#-2924	(57)	#-2943	(58)	#-2944	(58)
		#-2981	(59)	#-2983	(59)	#-2984	(59)	#-2985	(59)
		#-2987	(59)	#-2988	(59)	#-2989	(59)	#-2990	(59)
		#-3022	(60)	#-3034	(60)	#-3035	(60)	#-3036	(60)
		#-3037	(60)	#-3038	(60)	#-3039	(60)	#-3104	(61)
		#-3105	(61)	#-3106	(61)	#-3108	(61)	#-3109	(61)
		#-3246	(64)	#-3280	(65)	#-3283	(65)	#-3284	(65)
		#-3304	(65)	#-3305	(65)	#-3316	(66)	#-3318	(66)
		#-3319	(66)	#-3333	(66)	#-3334	(66)	#-3378	(67)
		#-3379	(67)	#-3380	(67)	#-3382	(67)	#-3383	(67)
		#-376	(7)	#-530	(10)	#-532	(10)	#-547	(11)
		#-561	(12)	#-563	(12)	#-565	(12)	#-578	(13)
		#-581	(13)	#-591	(13)	#-594	(13)	#-605	(13)
		#-608	(13)	#-619	(13)	#-622	(13)	#-632	(13)
		#-635	(13)	#-645	(13)	#-648	(13)	#-658	(13)
		#-661	(13)	#-671	(13)	#-674	(13)	#-684	(13)
		#-687	(13)	#-696	(13)	#-700	(13)	#-710	(13)
		#-714	(13)	#-736	(13)	#-748	(13)	#-751	(13)
		#-761	(13)	#-764	(13)	#-771	(13)		
DRIVE0	00000000-XR	#-1195	(20)	#-1272	(21)	#-1312	(22)	#-1349	(23)
		#-1386	(24)	#-1447	(25)	#-1520	(26)	#-1581	(27)
		#-1646	(28)	#-1710	(29)	#-1774	(30)	#-1837	(31)
		#-1926	(32)	#-2001	(34)	#-2049	(35)	#-217	(4)
		#-2305	(43)	#-2438	(46)	#-2561	(49)	#-2831	(55)
		#-2878	(56)	#-2946	(58)	#-2977	(59)	#-3032	(60)
		#-3102	(61)	#-3244	(64)	#-3274	(65)	#-3376	(67)

```
                                #-697    (13)   #-711    (13)
DRIVE_ADR_TBL                   00000000-XR     #-723    (13)
DRIVE_OFFSET                    00000000-XR     #-2029   (34)
DS$BGNSUB                       00000000-XR      1078    (16)     1105    (17)     1119    (18)     1153    (19)
                                                 1258    (21)     1295    (22)     1333    (23)     1370    (24)
                                                 1424    (25)     1498    (26)     152     (2)      1568    (27)
                                                 1631    (28)     1696    (29)     1760    (30)     1824    (31)
                                                 185     (3)      1922    (32)     1967    (33)     1992    (34)
                                                 2044    (35)     2078    (36)     211     (4)      2124    (37)
                                                 2155    (38)     2187    (39)     2218    (40)     2249    (41)
                                                 2278    (42)     2302    (43)     2327    (44)     2378    (45)
                                                 241     (5)      2419    (46)     2474    (47)     2499    (48)
                                                 2551    (49)     2587    (50)     2629    (51)     2672    (52)
                                                 2725    (53)     277     (6)      2771    (54)     2811    (55)
                                                 2858    (56)     2909    (57)     2931    (58)     2958    (59)
                                                 3011    (60)     3086    (61)     3123    (62)     3201    (63)
                                                 3238    (64)     3262    (65)     3308    (66)     3356    (67)
                                                 3396    (68)     3459    (69)     3522    (70)     3582    (71)
                                                 436     (8)      452     (9)      528     (10)     542     (11)
                                                 558     (12)     574     (13)     801     (14)     827     (15)
DS$BREAK                        00000000-XR      1050    (16)     1173    (20)     1245    (20)     1407    (25)
                                                 1887    (32)     2346    (45)     2456    (47)     3052    (61)
                                                 3180    (63)     329     (7)      3337    (67)     3621    (71)
                                                 412     (7)      496     (10)     779     (14)
DS$CKLOOP                       00000000-XR      1002    (15)     1011    (15)     1020    (15)     1029    (15)
                                                 1038    (15)     1047    (15)     1115    (17)     1137    (18)
                                                 1149    (18)     1168    (19)     1209    (20)     1223    (20)
                                                 1243    (20)     1282    (21)     1292    (21)     1330    (22)
                                                 1367    (23)     1404    (24)     1472    (25)     1490    (25)
                                                 1545    (26)     1560    (26)     1605    (27)     1623    (27)
                                                 1670    (28)     1688    (28)     171     (2)      1734    (29)
                                                 1752    (29)     1798    (30)     1816    (30)     1861    (31)
                                                 1879    (31)     1937    (32)     1951    (32)     1961    (32)
                                                 1979    (33)     1989    (33)     201     (3)      2019    (34)
                                                 2028    (34)     2062    (35)     2072    (35)     2108    (36)
                                                 2118    (36)     2139    (37)     2149    (37)     2170    (38)
                                                 2180    (38)     2202    (39)     2212    (39)     2233    (40)
                                                 2243    (40)     227     (4)      2272    (41)     2296    (42)
                                                 2321    (43)     2343    (44)     2409    (45)     2448    (46)
                                                 2495    (47)     253     (5)      2538    (48)     2548    (48)
                                                 2574    (49)     2584    (49)     2616    (50)     2626    (50)
                                                 2659    (51)     2669    (51)     2706    (52)     2716    (52)
                                                 273     (5)      2753    (53)     2763    (53)     2798    (54)
                                                 2808    (54)     2855    (55)     2906    (56)     2928    (57)
                                                 2955    (58)     2998    (59)     3008    (59)     3049    (60)
                                                 3120    (61)     3151    (62)     3160    (62)     3169    (62)
                                                 3216    (63)     3256    (64)     3300    (65)     3330    (66)
                                                 3393    (67)     3436    (68)     3448    (68)     3499    (69)
                                                 3510    (69)     3559    (70)     3570    (70)     3604    (71)
                                                 3619    (71)     366     (7)      374     (7)      381     (7)
                                                 390     (7)      399     (7)      410     (7)      446     (8)
                                                 463     (9)      473     (9)      483     (9)      493     (9)
                                                 539     (10)     555     (11)     571     (12)     587     (13)
                                                 601     (13)     615     (13)     628     (13)     641     (13)
                                                 654     (13)     667     (13)     680     (13)     693     (13)
                                                 707     (13)     720     (13)     742     (13)     757     (13)
                                                 770     (13)     776     (13)     818     (14)     840     (15)
```

		849	(15)	858	(15)	867	(15)	876	(15)						
		885	(15)	894	(15)	903	(15)	912	(15)						
		921	(15)	930	(15)	939	(15)	948	(15)						
		957	(15)	966	(15)	975	(15)	984	(15)						
		993	(15)												
DS$CLRVEC	00000000-XR	178	(2)	203	(3)	230	(4)	2496	(47)						
		291	(6)	325	(6)	326	(6)								
DS$CVTREG	00000000-XR	250	(5)	270	(5)										
DS$ENDSUB	00000000-XR	1048	(15)	1103	(16)	1117	(17)	1151	(18)						
		1171	(19)	1293	(21)	1331	(22)	1368	(23)						
		1405	(24)	1496	(25)	1566	(26)	1629	(27)						
		1694	(28)	1758	(29)	179	(2)	1822	(30)						
		1885	(31)	1962	(32)	1990	(33)	2039	(34)						
		204	(3)	2073	(35)	2119	(36)	2150	(37)						
		2181	(38)	2213	(39)	2244	(40)	2273	(41)						
		2297	(42)	231	(4)	2322	(43)	2344	(44)						
		2417	(45)	2454	(46)	2497	(47)	2549	(48)						
		2585	(49)	2627	(50)	2670	(51)	2717	(52)						
		275	(5)	2764	(53)	2809	(54)	2856	(55)						
		2907	(56)	2929	(57)	2956	(58)	3009	(59)						
		3050	(60)	3121	(61)	3178	(62)	3236	(63)						
		3260	(64)	327	(6)	3306	(65)	3335	(66)						
		3394	(67)	3457	(68)	3520	(69)	3580	(70)						
		3620	(71)	447	(8)	494	(9)	540	(10)						
		556	(11)	572	(12)	777	(13)	820	(14)						
DS$ERRHARD	00000000-XR	1001	(15)	1010	(15)	1019	(15)	1028	(15)						
		1037	(15)	1046	(15)	1098	(16)	1114	(17)						
		1136	(18)	1148	(18)	1165	(19)	1208	(20)						
		1222	(20)	1242	(20)	1281	(21)	1291	(21)						
		1329	(22)	1366	(23)	1403	(24)	1471	(25)						
		1489	(25)	1544	(26)	1559	(26)	1604	(27)						
		1622	(27)	1669	(28)	168	(2)	1687	(28)						
		1733	(29)	1751	(29)	1797	(30)	1815	(30)						
		1860	(31)	1878	(31)	1936	(32)	1949	(32)						
		1960	(32)	1978	(33)	198	(3)	1988	(33)						
		2018	(34)	2027	(34)	2061	(35)	2071	(35)						
		2107	(36)	2117	(36)	2138	(37)	2148	(37)						
		2169	(38)	2179	(38)	2201	(39)	2211	(39)						
		2232	(40)	224	(4)	2242	(40)	2271	(41)						
		2295	(42)	2320	(43)	2342	(44)	2408	(45)						
		2447	(46)	246	(5)	2492	(47)	2537	(48)						
		2547	(48)	2573	(49)	2583	(49)	2615	(50)						
		2625	(50)	2658	(51)	266	(5)	2668	(51)						
		2705	(52)	2715	(52)	2752	(53)	2762	(53)						
		2797	(54)	2807	(54)	2854	(55)	2902	(56)						
		2927	(57)	2954	(58)	2997	(59)	3007	(59)						
		3048	(60)	3119	(61)	3150	(62)	3157	(62)						
		3166	(62)	319	(6)	3215	(63)	3255	(64)						
		3299	(65)	3329	(66)	3392	(67)	3424	(68)						
		3430	(68)	3442	(68)	3485	(69)	3491	(69)						
		3504	(69)	3545	(70)	3551	(70)	3564	(70)						
		3603	(71)	3618	(71)	365	(7)	373	(7)						
		380	(7)	389	(7)	398	(7)	409	(7)						
		445	(8)	462	(9)	472	(9)	482	(9)						
		492	(9)	538	(10)	552	(11)	570	(12)						
		586	(13)	600	(13)	614	(13)	627	(13)						
		640	(13)	653	(13)	666	(13)	679	(13)						

```
                                                692    (13)     706    (13)     719    (13)     741    (13)
                                                756    (13)     769    (13)     775    (13)     817    (14)
                                                839    (15)     848    (15)     857    (15)     866    (15)
                                                875    (15)     884    (15)     893    (15)     902    (15)
                                                911    (15)     920    (15)     929    (15)     938    (15)
                                                947    (15)     956    (15)     965    (15)     974    (15)
                                                983    (15)     992    (15)
DS$ESCAPE                       00000000-XR     554    (11)
DS$PRINTB                       00000000-XR    1494    (25)    1564    (26)    1627    (27)    1692    (28)
                                               1756    (29)    1820    (30)    1883    (31)    2494    (47)
                                               3159    (62)    3168    (62)
DS$PRINTX                       00000000-XR    1100    (16)    1167    (19)    2011    (34)    2037    (34)
                                                252    (5)      272    (5)      321    (6)     3432    (68)
                                               3447    (68)    3493    (69)    3509    (69)    3553    (70)
                                               3569    (70)
DS$SETIPL                       00000000-XR    2309    (43)    2313    (43)    2331    (44)    2335    (44)
DS$SETVEC                       00000000-XR     161    (2)      191    (3)      216    (4)     2485    (47)
                                                279    (6)      280    (6)      281    (6)
DS$WAITUS                       00000000-XR    1451    (25)    1524    (26)    1584    (27)    1649    (28)
                                               1713    (29)    1777    (30)    1840    (31)    2054    (35)
                                               2093    (36)    2311    (43)    2333    (44)    3247    (64)
                                               3281    (65)    3317    (66)
DSA$GL_FLAGS                    0000FE00       1483    (25)    1484    (25)    1556    (26)    1616    (27)
                                               1617    (27)    1681    (28)    1682    (28)    1745    (29)
                                               1746    (29)    1809    (30)    1810    (30)    1872    (31)
                                               1873    (31)   #-2009   (34)   #-2035   (34)    3173    (62)
DSA$GL_PASSNO                    0000FE54     #-2007   (34)   #-2033   (34)
DSA$M_TRACE                     =00000400     #-2009   (34)   #-2035   (34)
DSA$V_OPER                      =0000000C     #-1483   (25)   #-1616   (27)   #-1681   (28)   #-1745   (29)
                                              #-1809   (30)   #-1872   (31)
DSA$V_QUICK                     =00000008     #-1484   (25)   #-1556   (26)   #-1617   (27)   #-1682   (28)
                                              #-1746   (29)   #-1810   (30)   #-1873   (31)   #-3173   (62)
DT_ABORT                         00000000-XR  #-1283   (21)   #-1938   (32)   #-3107   (61)   #-3127   (62)
                                              #-3381   (67)   #-3400   (68)   #-475    (9)
DT_BUSY                          00000000-XR  #-1929   (32)   #-2132   (37)   #-2142   (37)   #-2163   (38)
                                              #-2173   (38)   #-2195   (39)   #-2205   (39)   #-2226   (40)
                                              #-2236   (40)   #-2284   (42)   #-2330   (44)   #-2441   (46)
EDM                              00000000-XR  #-2835   (55)   #-2838   (55)   #-2883   (56)   #-2886   (56)
ENV$M_DOMAIN                    =00000002     137    (1)
ENV$M_LEVEL                     =00000001     137    (1)
ENV$M_SUPER                     =000003FC     137    (1)
ENV$S_DOMAIN                    =00000001     137    (1)
ENV$S_LEVEL                     =00000001     137    (1)
ENV$S_SUPER                     =00000008     137    (1)
ENV$V_DOMAIN                    =00000001     137    (1)      137    (1)
ENV$V_LEVEL                     =00000000     137    (1)      137    (1)
ENV$V_SUPER                     =00000002     137    (1)
ENV$_CPU                        =00000000     137    (1)      137    (1)
ENV$_FUNCTIONAL                 =00000000     137    (1)      137    (1)
ENV$_REPAIR                     =00000001     137    (1)      137    (1)
ENV$_SUPER                      =00000001     137    (1)
ENV$_SYSTEM                     =00000001     137    (1)      137    (1)
ERCE                             00000000-XR             #-2840   (55)   #-2888   (56)
ERR_STAT                         00000000-XR             #-2881   (56)
EXT_REG_MSG                      00000000-XR               1194   (20)     1211   (20)
FMT_ANY_REG                      00000000-XR               1194   (20)     1211   (20)     250    (5)      270    (5)
                                                           3204   (63)     3271   (65)     382    (7)      391    (7)
```

```
                                                  804    (14)    831    (15)
FMT_BADNEXUS                  00000000-XR         2494   (47)
FMT_CONTRL_REG                00000000-XR         368    (7)     432    (8)
FMT_DIAG_REG                  00000000-XR         1269   (21)    1433   (25)    1504   (26)    1576   (27)
                                                  375    (7)     526    (10)
FMT_DRIVENORES                00000000-XR         2037   (34)
FMT_DRIVERES                  00000000-XR         2011   (34)
FMT_DTERR                     00000000-XR         3159   (62)    3168   (62)
FMT_DUADR_ERR                 00000000-XR         1167   (19)
FMT_DUALRS_ERR                00000000-XR         319    (6)
FMT_DUALRS_MSG                00000000-XR         321    (6)
FMT_DUALWT_ERR                00000000-XR         1098   (16)
FMT_DUALWT_MSG                00000000-XR         1100   (16)
FMT_ERRSUM                    00000000-XR         1494   (25)    1564   (26)    1627   (27)    1692   (28)
                                                  1756   (29)    1820   (30)    1883   (31)
FMT_IBUS0                     00000000-XR         272    (5)
FMT_IBUS1                     00000000-XR         252    (5)
FMT_MAP_REG                   00000000-XR         1074   (16)    401    (7)
FMT_STATUS_REG                00000000-XR         1228   (20)    1463   (25)    1536   (26)    1596   (27)
                                                  1661   (28)    1725   (29)    1789   (30)    1852   (31)
                                                  1912   (32)    2046   (35)    2483   (47)    3110   (61)
                                                  3143   (62)    3361   (67)    3585   (71)    359    (7)
FMT_VARER                     00000000-XR         3432   (68)    3493   (69)    3553   (70)
FMT_XFERERR                   00000000-XR         3447   (68)    3509   (69)    3569   (70)
IBC_MSG                       00000000-XR        #-3003  (59)
INH_BYTE_CNT                  00000000-XR        #-485   (9)
INTERRUPT_EN                  00000000-XR        #-465   (9)
INVRT_MAP_PAR                 00000000-XR        #-2595  (50)   #-618   (13)
INVRT_MB_CPAR                 00000000-XR        #-2560  (49)   #-2921  (57)   #-2944  (58)   #-604   (13)
INVRT_MB_DPAR                 00000000-XR        #-2508  (48)   #-2522  (48)   #-590   (13)
INVRT_SILO_PAR                00000000-XR        #-3022  (60)   #-3039  (60)   #-747   (13)
ISSUE_MMREAD                  00000000-XR         2395   (45)
LUN                           00000000-XR        #-1001  (15)   #-1010  (15)   #-1019  (15)   #-1028  (15)
                                                 #-1037  (15)   #-1046  (15)   #-1098  (16)   #-1114  (17)
                                                 #-1136  (18)   #-1148  (18)   #-1165  (19)   #-1208  (20)
                                                 #-1222  (20)   #-1242  (20)   #-1281  (21)   #-1291  (21)
                                                 #-1329  (22)   #-1366  (23)   #-1403  (24)   #-1471  (25)
                                                 #-1489  (25)   #-1544  (26)   #-1559  (26)   #-1604  (27)
                                                 #-1622  (27)   #-1669  (28)   #-168   (2)    #-1687  (28)
                                                 #-1733  (29)   #-1751  (29)   #-1797  (30)   #-1815  (30)
                                                 #-1860  (31)   #-1878  (31)   #-1936  (32)   #-1949  (32)
                                                 #-1960  (32)   #-1978  (33)   #-198   (3)    #-1988  (33)
                                                 #-2018  (34)   #-2027  (34)   #-2061  (35)   #-2071  (35)
                                                 #-2107  (36)   #-2117  (36)   #-2138  (37)   #-2148  (37)
                                                 #-2169  (38)   #-2179  (38)   #-2201  (39)   #-2211  (39)
                                                 #-2232  (40)   #-224   (4)    #-2242  (40)   #-2271  (41)
                                                 #-2295  (42)   #-2320  (43)   #-2342  (44)   #-2408  (45)
                                                 #-2447  (46)   #-246   (5)    #-2492  (47)   #-2537  (48)
                                                 #-2547  (48)   #-2573  (49)   #-2583  (49)   #-2615  (50)
                                                 #-2625  (50)   #-2658  (51)   #-266   (5)    #-2668  (51)
                                                 #-2705  (52)   #-2715  (52)   #-2752  (53)   #-2762  (53)
                                                 #-2797  (54)   #-2807  (54)   #-2854  (55)   #-2902  (56)
                                                 #-2927  (57)   #-2954  (58)   #-2997  (59)   #-3007  (59)
                                                 #-3048  (60)   #-3119  (61)   #-3150  (62)   #-3157  (62)
                                                 #-3166  (62)   #-319   (6)    #-3215  (63)   #-3255  (64)
                                                 #-3299  (65)   #-3329  (66)   #-3392  (67)   #-3424  (68)
                                                 #-3430  (68)   #-3442  (68)   #-3485  (69)   #-3491  (69)
```

```
                                        #-3504  (69)    #-3545  (70)    #-3551  (70)    #-3564  (70)
                                        #-3603  (71)    #-3618  (71)    #-365   (7)     #-373   (7)
                                        #-380   (7)     #-389   (7)     #-398   (7)     #-409   (7)
                                        #-445   (8)     #-462   (9)     #-472   (9)     #-482   (9)
                                        #-492   (9)     #-538   (10)    #-552   (11)    #-570   (12)
                                        #-586   (13)    #-600   (13)    #-614   (13)    #-627   (13)
                                        #-640   (13)    #-653   (13)    #-666   (13)    #-679   (13)
                                        #-692   (13)    #-706   (13)    #-719   (13)    #-741   (13)
                                        #-756   (13)    #-769   (13)    #-775   (13)    #-817   (14)
                                        #-839   (15)    #-848   (15)    #-857   (15)    #-866   (15)
                                        #-875   (15)    #-884   (15)    #-893   (15)    #-902   (15)
                                        #-911   (15)    #-920   (15)    #-929   (15)    #-938   (15)
                                        #-947   (15)    #-956   (15)    #-965   (15)    #-974   (15)
                                        #-983   (15)    #-992   (15)
MACH_CHK_SRV            00000000-XR       161   (2)       191   (3)       216   (4)      2485   (47)
MAINT_MODE              00000000-XR     #-1198  (20)    #-1232  (20)    #-1271  (21)    #-1311  (22)
                                        #-1348  (23)    #-1385  (24)    #-1448  (25)    #-1521  (26)
                                        #-1578  (27)    #-1642  (28)    #-1707  (29)    #-1770  (30)
                                        #-1834  (31)    #-1925  (32)    #-1969  (33)    #-2048  (35)
                                        #-2080  (36)    #-2126  (37)    #-2157  (38)    #-2189  (39)
                                        #-2220  (40)    #-2251  (41)    #-2280  (42)    #-2285  (42)
                                        #-2286  (42)    #-2304  (43)    #-2329  (44)    #-2436  (46)
                                        #-2507  (48)    #-2559  (49)    #-257   (5)     #-2594  (50)
                                        #-2636  (51)    #-2680  (52)    #-2727  (53)    #-2773  (54)
                                        #-2824  (55)    #-2871  (56)    #-2920  (57)    #-2942  (58)
                                        #-2968  (59)    #-3021  (60)    #-3101  (61)    #-3241  (64)
                                        #-3276  (65)    #-3311  (66)    #-3363  (67)    #-454   (9)
                                        #-455   (9)     #-457   (9)     #-544   (11)    #-560   (12)
                                        #-576   (13)    #-589   (13)    #-603   (13)    #-617   (13)
                                        #-630   (13)    #-643   (13)    #-656   (13)    #-669   (13)
                                        #-682   (13)    #-695   (13)    #-709   (13)    #-726   (13)
                                        #-746   (13)    #-759   (13)
MAP_1_PATTRN            00000000-XR     #-1107  (17)    #-1160  (19)
MAP_INVALID             00000000-XR     #-2650  (51)
MAP_OFFSET              00000000-XR     #-1077  (16)    #-1307  (22)    #-1344  (23)    #-1381  (24)
                                        #-1432  (25)    #-1503  (26)    #-190   (3)     #-1917  (32)
                                        #-2081  (36)    #-2381  (45)    #-2421  (46)    #-2512  (48)
                                        #-2827  (55)    #-2874  (56)    #-2972  (59)    #-3027  (60)
                                        #-3094  (61)    #-3219  (63)    #-3369  (67)    #-403   (7)
MAP_PE                  00000000-XR     #-2607  (50)
MAP_PTR_MSK             00000000-XR     #-1443  (25)    #-1516  (26)    #-2514  (48)    #-2600  (50)
                                        #-2639  (51)    #-2777  (54)    #-2829  (55)    #-2876  (56)
                                        #-2974  (59)    #-3029  (60)
MASS_CNTRL_PE           00000000-XR     #-2567  (49)
MASS_CTOD               00000000-XR     #-1273  (21)    #-1285  (21)    #-376   (7)     #-531   (10)
                                        #-546   (11)    #-580   (13)    #-593   (13)    #-607   (13)
                                        #-621   (13)    #-634   (13)    #-647   (13)    #-660   (13)
                                        #-673   (13)    #-686   (13)    #-713   (13)    #-730   (13)
                                        #-750   (13)
MASS_DATA_PE            00000000-XR     #-2529  (48)
MASS_ECP                00000000-XR     #-2980  (59)
MASS_FAIL               00000000-XR     #-1273  (21)    #-1284  (21)    #-545   (11)    #-562   (12)
                                        #-579   (13)    #-592   (13)    #-606   (13)    #-620   (13)
                                        #-633   (13)    #-646   (13)    #-659   (13)    #-672   (13)
                                        #-685   (13)    #-701   (13)    #-713   (13)    #-730   (13)
                                        #-749   (13)    #-762   (13)
MASS_RUN                00000000-XR     #-1273  (21)
```

```
MASS_WCLK                       00000000-XR                    #-579    (13)
MBDIB_SEL                       00000000-XR                    #-1455   (25)   #-1528   (26)   #-1588   (27)   #-1653   (28)
                                                               #-1717   (29)   #-1781   (30)   #-1844   (31)   #-683    (13)
MBE                             =00000002      83      (1)
MEM_CSR0                        00000000-XR                    #-2903   (56)   #-2905   (56)
MEM_CSR1                        00000000-XR                    #-2834   (55)   #-2835   (55)   #-2838   (55)   #-2840   (55)
                                                               #-2847   (55)   #-2882   (56)   #-2883   (56)   #-2886   (56)
                                                               #-2888   (56)   #-2895   (56)
MIOBUFFER                       00000000-XR                      1304   (22)     1341   (23)     1378   (24)     1914   (32)
                                                                 2509   (48)     2596   (50)     2637   (51)   #-2681   (52)
                                                               #-2692   (52)   #-2728   (53)   #-2737   (53)     2823   (55)
                                                                 2870   (56)     2969   (59)     3023   (60)   #-3024   (60)
                                                                 3090   (61)   #-3098   (61)   #-3099   (61)     3129   (62)
                                                                 3226   (63)     3264   (65)     3364   (67)   #-3406   (68)
                                                               #-3439   (68)     3443   (68)   #-3468   (69)   #-3501   (69)
                                                                 3505   (69)   #-3561   (70)     3565   (70)   #-3588   (71)
MISSED_XFER                     00000000-XR                    #-2050   (35)   #-2306   (43)
MM_XFER                         00000000-XR                      2520   (48)     2528   (48)     2606   (50)     2648   (51)
                                                                 2689   (52)     2696   (52)     2734   (53)     2742   (53)
                                                                 2785   (54)     3140   (62)     3416   (68)     3477   (69)
                                                                 3537   (70)     3595   (71)     3610   (71)
MSG_001                         00000002-R     140     (2)       141   (2)
MSG_002                         00000002-R     331     (7)       332   (7)
MSG_003                         00000002-R     415     (8)       416   (8)
MSG_004                         00000002-R     498     (10)      499   (10)
MSG_005                         00000002-R     781     (14)      782   (14)
MSG_006                         00000002-R     1052    (16)      1053  (16)
MSG_007                         00000002-R     1175    (20)      1176  (20)
MSG_008                         00000002-R     1248    (21)      1249  (21)
MSG_009                         00000002-R     1409    (25)      1410  (25)
MSG_010                         00000002-R     1889    (32)      1890  (32)
MSG_011                         00000002-R     2348    (45)      2349  (45)
MSG_012                         00000002-R     2458    (47)      2459  (47)
MSG_013                         00000002-R     3054    (61)      3055  (61)
MSG_014                         00000002-R     3182    (63)      3183  (63)
MSG_015                         00000002-R     3339    (67)      3340  (67)
NIBBLE1                         00000000-XR                    #-1323   (22)   #-1360   (23)   #-1397   (24)
NIBBLE4                         00000000-XR                    #-564    (12)   #-763    (13)
NODRIVE                         00000000-XR                    #-2012   (34)   #-2038   (34)
NOOP                            00000000-XR                    #-698    (13)   #-712    (13)
NO_RESPONSE                     00000000-XR                    #-2789   (54)
NXM_ADR                         00000000-XR                    #-163    (2)    #-193    (3)    #-219    (4)
NXM_RD_FLAG                     00000000-XR                    #-162    (2)      165    (2)    #-192    (3)      195    (3)
                                                               #-218    (4)      221    (4)    #-2486   (47)     2490   (47)
PF_FIELD                        00000000-XR                    #-1305   (22)   #-1342   (23)   #-1379   (24)   #-2510   (48)
                                                               #-2775   (54)   #-2825   (55)   #-2872   (56)   #-2970   (59)
                                                               #-3025   (60)   #-3091   (61)   #-3133   (62)   #-3222   (63)
                                                               #-3227   (63)   #-3231   (63)   #-3366   (67)   #-3426   (68)
                                                               #-3487   (69)   #-3547   (70)
PGM_INIT                        00000000-XR                    #-1076   (16)   #-1197   (20)   #-1231   (20)   #-1270   (21)
                                                               #-1303   (22)   #-1340   (23)   #-1377   (24)   #-1434   (25)
                                                               #-1505   (26)   #-1577   (27)   #-1641   (28)   #-1706   (29)
                                                               #-1769   (30)   #-1833   (31)   #-187    (3)    #-1924   (32)
                                                               #-1968   (33)   #-2003   (34)   #-2047   (35)   #-2079   (36)
                                                               #-2125   (37)   #-213    (4)    #-2156   (38)   #-2188   (39)
                                                               #-2219   (40)   #-2250   (41)   #-2279   (42)   #-2289   (42)
                                                               #-2303   (43)   #-2328   (44)   #-2377   (45)   #-2391   (45)
```

		#-242 (5) #-2432 (46) #-2506 (48) #-254 (5)
		#-2558 (49) #-2593 (50) #-2635 (51) #-2679 (52)
		#-2726 (53) #-274 (5) #-2772 (54) #-2822 (55)
		#-2869 (56) #-2919 (57) #-2941 (58) #-2967 (59)
		#-3020 (60) #-3089 (61) #-3205 (63) #-3240 (64)
		#-3263 (65) #-3275 (65) #-3310 (66) #-3362 (67)
		#-357 (7) #-437 (8) #-453 (9) #-464 (9)
		#-474 (9) #-484 (9) #-529 (10) #-543 (11)
		#-559 (12) #-575 (13) #-588 (13) #-602 (13)
		#-616 (13) #-629 (13) #-642 (13) #-655 (13)
		#-668 (13) #-681 (13) #-694 (13) #-708 (13)
		#-721 (13) #-725 (13) #-745 (13) #-758 (13)
		#-803 (14) #-830 (15)
PM	00000000-XR	#-2835 (55) #-2840 (55) #-2883 (56) #-2888 (56)
PRINT_CMDER	00000000-XR	1329 (22) 1366 (23) 1403 (24)
PRINT_DTERR	00000000-XR	1489 (25) 1559 (26) 1622 (27) 1687 (28)
		1751 (29) 1815 (30) 1878 (31)
PRINT_IBCERR	00000000-XR	3007 (59)
PRINT_MAPERR	00000000-XR	2408 (45)
PRINT_NXADR	00000000-XR	168 (2) 198 (3) 224 (4)
PRINT_SBE	00000000-XR	1001 (15) 1010 (15) 1019 (15) 1028 (15)
		1037 (15) 1046 (15) 1114 (17) 1136 (18)
		1148 (18) 1165 (19) 1208 (20) 1222 (20)
		1242 (20) 1281 (21) 1291 (21) 1471 (25)
		1544 (26) 1604 (27) 1669 (28) 1733 (29)
		1797 (30) 1860 (31) 1936 (32) 1949 (32)
		1960 (32) 1978 (33) 1988 (33) 2027 (34)
		2061 (35) 2071 (35) 2107 (36) 2117 (36)
		2138 (37) 2148 (37) 2169 (38) 2179 (38)
		2201 (39) 2211 (39) 2232 (40) 2242 (40)
		2271 (41) 2295 (42) 2320 (43) 2342 (44)
		2447 (46) 2537 (48) 2547 (48) 2573 (49)
		2583 (49) 2615 (50) 2625 (50) 2658 (51)
		2668 (51) 2705 (52) 2715 (52) 2752 (53)
		2762 (53) 2797 (54) 2807 (54) 2854 (55)
		2902 (56) 2954 (58) 2997 (59) 3048 (60)
		3119 (61) 3150 (62) 3215 (63) 3255 (64)
		3299 (65) 3329 (66) 3392 (67) 3424 (68)
		3485 (69) 3545 (70) 3603 (71) 3618 (71)
		365 (7) 373 (7) 380 (7) 389 (7)
		398 (7) 409 (7) 445 (8) 462 (9)
		472 (9) 482 (9) 492 (9) 538 (10)
		552 (11) 570 (12) 586 (13) 600 (13)
		614 (13) 627 (13) 640 (13) 653 (13)
		666 (13) 679 (13) 692 (13) 706 (13)
		719 (13) 741 (13) 756 (13) 769 (13)
		775 (13) 817 (14) 839 (15) 848 (15)
		857 (15) 866 (15) 875 (15) 884 (15)
		893 (15) 902 (15) 911 (15) 920 (15)
		929 (15) 938 (15) 947 (15) 956 (15)
		965 (15) 974 (15) 983 (15) 992 (15)
PROGRAM_ERROR	00000000-XR	#-2132 (37) #-2163 (38) #-2195 (39) #-2226 (40)
		#-2284 (42)
READ	00000000-XR	#-1274 (21) #-1387 (24) #-1927 (32) #-2053 (35)
		#-2128 (37) #-2130 (37) #-2159 (38) #-2191 (39)
		#-2222 (40) #-2256 (41) #-2282 (42) #-2310 (43)
		#-2332 (44) #-2438 (46) #-2525 (48) #-2982 (59)

```
                                        #-3137  (62)    #-3413  (68)
READ_REV                00000000 XR     #-3474  (69)
REG_NAME                00000000-XR     #-1074  (16)    #-1194  (20)    #-1211  (20)    #-1228  (20)
                                        #-1269  (21)    #-1433  (25)    #-1463  (25)    #-1504  (26)
                                        #-1536  (26)    #-1576  (27)    #-1596  (27)    #-1661  (28)
                                        #-1725  (29)    #-1789  (30)    #-1852  (31)    #-1912  (32)
                                        #-2046  (35)    #-2483  (47)    #-3110  (61)    #-3143  (62)
                                        #-3204  (63)    #-3271  (65)    #-3361  (67)    #-3585  (71)
                                        #-359   (7)     #-368   (7)     #-375   (7)     #-382   (7)
                                        #-391   (7)     #-401   (7)     #-432   (8)     #-526   (10)
                                        #-804   (14)    #-831   (15)
REG_NO                  00000000-XR     #-1074  (16)    #-1194  (20)    #-1211  (20)    #-1228  (20)
                                        #-1269  (21)    #-1433  (25)    #-1463  (25)    #-1504  (26)
                                        #-1536  (26)    #-1576  (27)    #-1596  (27)    #-1661  (28)
                                        #-1725  (29)    #-1789  (30)    #-1852  (31)    #-1912  (32)
                                        #-2046  (35)    #-2483  (47)    #-3110  (61)    #-3143  (62)
                                        #-3204  (63)    #-3271  (65)    #-3361  (67)    #-3585  (71)
                                        #-359   (7)     #-368   (7)     #-375   (7)     #-382   (7)
                                        #-391   (7)     #-401   (7)     #-432   (8)     #-526   (10)
                                        #-804   (14)    #-831   (15)
REG_STRING              00000000-XR     #-1074  (16)    #-1194  (20)    #-1211  (20)    #-1228  (20)
                                        #-1269  (21)    #-1433  (25)    #-1463  (25)    #-1504  (26)
                                        #-1536  (26)    #-1576  (27)    #-1596  (27)    #-1661  (28)
                                        #-1725  (29)    #-1789  (30)    #-1852  (31)    #-1912  (32)
                                        #-2046  (35)    #-2483  (47)    #-3110  (61)    #-3143  (62)
                                        #-3204  (63)    #-3271  (65)    #-3361  (67)    #-3585  (71)
                                        #-359   (7)     #-368   (7)     #-375   (7)     #-382   (7)
                                        #-391   (7)     #-401   (7)     #-432   (8)     #-526   (10)
                                        #-804   (14)    #-831   (15)
REPORT_BUFFER           00000000-XR       250   (5)     #-252   (5)       270   (5)     #-272   (5)
RH11TB_MSK              00000000-XR     #-1236  (20)    #-1465  (25)    #-1538  (26)    #-1598  (27)
                                        #-1663  (28)    #-1727  (29)    #-1791  (30)    #-1854  (31)
                                        #-1930  (32)    #-1943  (32)    #-1954  (32)    #-1982  (33)
                                        #-2055  (35)    #-2066  (35)    #-2101  (36)    #-2112  (36)
                                        #-2133  (37)    #-2143  (37)    #-2164  (38)    #-2174  (38)
                                        #-2196  (39)    #-2206  (39)    #-2227  (40)    #-2237  (40)
                                        #-2265  (41)    #-2290  (42)    #-2314  (43)    #-2336  (44)
                                        #-2442  (46)    #-2531  (48)    #-2542  (48)    #-2568  (49)
                                        #-2578  (49)    #-2609  (50)    #-2620  (50)    #-2652  (51)
                                        #-2663  (51)    #-2699  (52)    #-2710  (52)    #-2746  (53)
                                        #-2757  (53)    #-2791  (54)    #-2802  (54)    #-2848  (55)
                                        #-2896  (56)    #-2991  (59)    #-3042  (60)    #-3113  (61)
                                        #-3387  (67)    #-3418  (68)    #-3479  (69)    #-3539  (70)
                                        #-3598  (71)    #-3612  (71)
RH11TB_PRES             00000000-XR     #-1191  (20)    #-1266  (21)    #-1430  (25)    #-1923  (32)
                                        #-2375  (45)    #-2480  (47)    #-3087  (61)    #-3202  (63)
                                        #-3357  (67)
RHBCR_MSG               00000000-XR       3204  (63)      3271  (65)      391   (7)
RHCR_MSG                00000000-XR       368   (7)       432   (8)
RHDR_MSG                00000000-XR       1269  (21)      1433  (25)      1504  (26)      1576  (27)
                                          375   (7)       526   (10)
RHMAPR_MSG              00000000-XR       1074  (16)      401   (7)
RHSR_MSG                00000000-XR       1228  (20)      1463  (25)      1536  (26)      1596  (27)
                                          1661  (28)      1725  (29)      1789  (30)      1852  (31)
                                          1912  (32)      2046  (35)      2483  (47)      3110  (61)
                                          3143  (62)      3361  (67)      3585  (71)      359   (7)
RHVAR_MSG               00000000-XR       382   (7)       804   (14)      831   (15)
```

```
RH_CUR_ADR                  00000000-XR                 #-1075   (16)   #-1192   (20)   #-1267   (21)   #-1302   (22)
                                                        #-1339   (23)   #-1376   (24)   #-1431   (25)   #-1502   (26)
                                                        #-158    (2)    #-173    (2)    #-186    (3)    #-1913   (32)
                                                        #-212    (4)    #-2376   (45)   #-2388   (45)   #-2428   (46)
                                                        #-2481   (47)   #-282    (6)    #-2821   (55)   #-2868   (56)
                                                        #-2918   (57)   #-2966   (59)   #-3019   (60)   #-3088   (61)
                                                        #-3203   (63)   #-3358   (67)   #-355    (7)    #-430    (8)
                                                        #-527    (10)   #-802    (14)   #-829    (15)
SEP_FUNCTIONAL              =00000002     137    (1)
SEP_REPAIR                  =00000003     137    (1)
SILO_PE                     00000000-XR                 #-3041   (60)
SIM_ATTN                    00000000-XR                 #-1971   (33)   #-1973   (33)   #-2943   (58)   #-670    (13)
SIM_EBL                     00000000-XR                 #-1320   (22)   #-1321   (22)   #-1357   (23)   #-1358   (23)
                                                        #-1394   (24)   #-1395   (24)   #-1460   (25)   #-1461   (25)
                                                        #-1533   (26)   #-1534   (26)   #-1593   (27)   #-1594   (27)
                                                        #-1658   (28)   #-1659   (28)   #-1722   (29)   #-1723   (29)
                                                        #-1786   (30)   #-1787   (30)   #-1849   (31)   #-1850   (31)
                                                        #-1939   (32)   #-1940   (32)   #-2099   (36)   #-2100   (36)
                                                        #-2261   (41)   #-2262   (41)   #-2845   (55)   #-2846   (55)
                                                        #-2893   (56)   #-2894   (56)   #-2989   (59)   #-2990   (59)
                                                        #-3037   (60)   #-3038   (60)   #-3108   (61)   #-3109   (61)
                                                        #-3304   (65)   #-3305   (65)   #-3333   (66)   #-3334   (66)
                                                        #-3382   (67)   #-3383   (67)   #-644    (13)   #-771    (13)
SIM_EXCEPT                  00000000-XR                 #-2987   (59)   #-760    (13)
SIM_OCC                     00000000-XR                 #-1276   (21)   #-1284   (21)   #-1314   (22)   #-1351   (23)
                                                        #-1388   (24)   #-1450   (25)   #-1523   (26)   #-1583   (27)
                                                        #-1648   (28)   #-1712   (29)   #-1776   (30)   #-1839   (31)
                                                        #-1928   (32)   #-2092   (36)   #-2129   (37)   #-2160   (38)
                                                        #-2192   (39)   #-2223   (40)   #-2257   (41)   #-2283   (42)
                                                        #-2312   (43)   #-2334   (44)   #-2439   (46)   #-2842   (55)
                                                        #-2890   (56)   #-2983   (59)   #-3034   (60)   #-3104   (61)
                                                        #-3246   (64)   #-3280   (65)   #-3316   (66)   #-3378   (67)
                                                        #-657    (13)
SIM_SCLK                    00000000-XR                 #-1316   (22)   #-1317   (22)   #-1353   (23)   #-1354   (23)
                                                        #-1390   (24)   #-1391   (24)   #-1453   (25)   #-1454   (25)
                                                        #-1526   (26)   #-1527   (26)   #-1586   (27)   #-1587   (27)
                                                        #-1651   (28)   #-1652   (28)   #-1715   (29)   #-1716   (29)
                                                        #-1779   (30)   #-1780   (30)   #-1842   (31)   #-1843   (31)
                                                        #-2096   (36)   #-2097   (36)   #-2258   (41)   #-2259   (41)
                                                        #-2843   (55)   #-2844   (55)   #-2891   (56)   #-2892   (56)
                                                        #-2984   (59)   #-2985   (59)   #-3035   (60)   #-3036   (60)
                                                        #-3105   (61)   #-3106   (61)   #-3283   (65)   #-3284   (65)
                                                        #-3318   (66)   #-3319   (66)   #-3379   (67)   #-3380   (67)
                                                        #-577    (13)
SIZ...                      =00000008     137    (1)     137    (1)
SR                          00000000-XR                 #-1239   (20)   #-1466   (25)   #-1539   (26)   #-1599   (27)
                                                        #-1664   (28)   #-1728   (29)   #-1792   (30)   #-1855   (31)
                                                        #-1931   (32)   #-1944   (32)   #-1952   (32)   #-1955   (32)
                                                        #-1972   (33)   #-1980   (33)   #-1983   (33)   #-2005   (34)
                                                        #-2020   (34)   #-2022   (34)   #-2056   (35)   #-2063   (35)
                                                        #-2064   (35)   #-2102   (36)   #-2109   (36)   #-2110   (36)
                                                        #-2131   (37)   #-2140   (37)   #-2141   (37)   #-2162   (38)
                                                        #-2171   (38)   #-2172   (38)   #-2194   (39)   #-2203   (39)
                                                        #-2204   (39)   #-2225   (40)   #-2234   (40)   #-2235   (40)
                                                        #-2266   (41)   #-2288   (42)   #-2315   (43)   #-2337   (44)
                                                        #-2440   (46)   #-2532   (48)   #-2539   (48)   #-2540   (48)
                                                        #-2564   (49)   #-2575   (49)   #-2576   (49)   #-2610   (50)
```

```
                                                #-2617   (50)   #-2618   (50)   #-2653   (51)   #-2660   (51)
                                                #-2661   (51)   #-2700   (52)   #-2707   (52)   #-2708   (52)
                                                #-2747   (53)   #-2754   (53)   #-2755   (53)   #-2792   (54)
                                                #-2799   (54)   #-2800   (54)   #-2849   (55)   #-2897   (56)
                                                #-2923   (57)   #-2992   (59)   #-3043   (60)   #-3114   (61)
                                                #-3145   (62)   #-3384   (67)   #-3404   (68)   #-3419   (68)
                                                #-3466   (69)   #-3480   (69)   #-3529   (70)   #-3540   (70)
                                                #-3586   (71)   #-3596   (71)   #-360    (7)    #-361    (7)
                                                #-3613   (71)     772   (13)

T10_S1                          0000006C-R   1922   (32)
T10_S10                         00000804-R   2249   (41)
T10_S10_X                       000008B0-R   2273   (41)
T10_S11                         000008BB-R   2278   (42)
T10_S11_X                       00000942-R   2297   (42)
T10_S12                         0000094D-R   2302   (43)
T10_S12_X                       000009E5-R   2322   (43)
T10_S13                         000009F0-R   2327   (44)
T10_S13_X                       00000A75-R   2344   (44)
T10_S1_X                        00000170-R   1962   (32)
T10_S2                          0000017B-R   1967   (33)
T10_S2_X                        00000219-R   1990   (33)
T10_S3                          00000224-R   1992   (34)
T10_S3_X                        00000316-R   2039   (34)
T10_S4                          00000321-R   2044   (35)
T10_S4_X                        000003F6-R   2073   (35)
T10_S5                          00000401-R   2078   (36)
T10_S5_X                        00000506-R   2119   (36)
T10_S6                          00000511-R   2124   (37)
T10_S6_X                        000005C5-R   2150   (37)
T10_S7                          000005D0-R   2155   (38)
T10_S7_X                        00000681-R   2181   (38)
T10_S8                          0000068C-R   2187   (39)
T10_S8_X                        0000073D-R   2213   (39)
T10_S9                          00000748-R   2218   (40)
T10_S9_X                        000007F9-R   2244   (40)
T11_S1                          00000031-R   2378   (45)
T11_S1_X                        000000EE-R   2417   (45)
T11_S2                          000000F9-R   2419   (46)
T11_S2_X                        000001C2-R   2454   (46)
T12_S1                          0000002A-R   2474   (47)
T12_S10                         00000838-R   2858   (56)
T12_S10_X                       0000094B-R   2907   (56)
T12_S11                         00000956-R   2909   (57)
T12_S11_X                       000009B4-R   2929   (57)
T12_S12                         000009BF-R   2931   (58)
T12_S12_X                       00000A31-R   2956   (58)
T12_S13                         00000A3C-R   2958   (59)
T12_S13_X                       00000B6F-R   3009   (59)
T12_S14                         00000B7A-R   3011   (60)
T12_S14_X                       00000C68-R   3050   (60)
T12_S1_X                        000000C9-R   2497   (47)
T12_S2                          000000D4-R   2499   (48)
T12_S2_X                        000001DD-R   2549   (48)
T12_S3                          000001E8-R   2551   (49)
T12_S3_X                        00000296-R   2585   (49)
T12_S4                          000002A1-R   2587   (50)
T12_S4_X                        00000379-R   2627   (50)
```

```
T12_S5                          00000384-R   2629   (51)
T12_S5_X                        0000045F-R   2670   (51)
T12_S6                          0000046A-R   2672   (52)
T12_S6_X                        00000555-R   2717   (52)
T12_S7                          00000560-R   2725   (53)
T12_S7_X                        0000063F-R   2764   (53)
T12_S8                          0000064A-R   2771   (54)
T12_S8_X                        00000720-R   2809   (54)
T12_S9                          0000072B-R   2811   (55)
T12_S9_X                        0000082D-R   2856   (55)
T13_S1                          0000001E-R   3086   (61)
T13_S1_X                        00000122-R   3121   (61)
T13_S2                          0000012D-R   3123   (62)
T13_S2_X                        00000275-R   3178   (62)
T14_S1                          0000001A-R   3201   (63)
T14_S1_X                        000000EE-R   3236   (63)
T14_S2                          000000F9-R   3238   (64)
T14_S2_X                        0000017B-R   3260   (64)
T14_S3                          00000186-R   3262   (65)
T14_S3_X                        0000028C-R   3306   (65)
T14_S4                          00000297-R   3308   (66)
T14_S4_X                        0000033F-R   3335   (66)
T15_S1                          00000026-R   3356   (67)
T15_S1_X                        00000125-R   3394   (67)
T15_S2                          00000130-R   3396   (68)
T15_S2_X                        00000280-R   3457   (68)
T15_S3                          0000028B-R   3459   (69)
T15_S3_X                        000003E5-R   3520   (69)
T15_S4                          000003F0-R   3522   (70)
T15_S4_X                        00000539-R   3580   (70)
T15_S5                          00000544-R   3582   (71)
T15_S5_X                        00000642-R   3620   (71)
T1_S1                           00000036-R   152    (2)
T1_S1_X                         000000C1-R   179    (2)
T1_S2                           000000CC-R   185    (3)
T1_S2_X                         0000015A-R   204    (3)
T1_S3                           00000165-R   211    (4)
T1_S3_X                         000001F4-R   231    (4)
T1_S4                           000001FF-R   241    (5)
T1_S4_X                         00000308-R   275    (5)
T1_S5                           00000313-R   277    (6)
T1_S5_X                         00000423-R   327    (6)
T3_S1                           00000046-R   436    (8)
T3_S1_X                         0000008F-R   447    (8)
T3_S2                           0000009A-R   452    (9)
T3_S2_X                         000001BA-R   494    (9)
T4_S1                           00000046-R   528    (10)
T4_S1_X                         00000090-R   540    (10)
T4_S2                           0000009B-R   542    (11)
T4_S2_X                         00000104-R   556    (11)      554    (11)
T4_S3                           0000010F-R   558    (12)
T4_S3_X                         0000016E-R   572    (12)
T4_S4                           00000179-R   574    (13)
T4_S4_X                         000006A8-R   777    (13)
T5_S1                           00000036-R   801    (14)
T5_S1_X                         000000B4-R   820    (14)
T5_S2                           000000BF-R   827    (15)
```

```
T5_S2_X                         00000668-R   1048   (15)
T6_S1                           00000052-R   1078   (16)
T6_S1_X                         000000D1-R   1103   (16)
T6_S2                           000000DC-R   1105   (17)
T6_S2_X                         00000129-R   1117   (17)
T6_S3                           00000134-R   1119   (18)
T6_S3_X                         000001D9-R   1151   (18)
T6_S4                           000001E4-R   1153   (19)
T6_S4_X                         00000256-R   1171   (19)
T8_S1                           0000001E-R   1258   (21)
T8_S1_X                         000000F5-R   1293   (21)
T8_S2                           00000100-R   1295   (22)
T8_S2_X                         000001D6-R   1331   (22)
T8_S3                           000001E1-R   1333   (23)
T8_S3_X                         000002B7-R   1368   (23)
T8_S4                           000002C2-R   1370   (24)
T8_S4_X                         00000398-R   1405   (24)
T9_S1                           0000002A-R   1424   (25)
T9_S1_X                         000001F2-R   1496   (25)
T9_S2                           000001FD-R   1498   (26)
T9_S2_X                         000003AC-R   1566   (26)
T9_S3                           000003B7-R   1568   (27)
T9_S3_X                         00000550-R   1629   (27)
T9_S4                           0000055B-R   1631   (28)
T9_S4_X                         000006DE-R   1694   (28)
T9_S5                           000006E9-R   1696   (29)
T9_S5_X                         0000086C-R   1758   (29)
T9_S6                           00000877-R   1760   (30)
T9_S6_X                         000009FA-R   1822   (30)
T9_S7                           00000A05-R   1824   (31)
T9_S7_X                         00000B88-R   1885   (31)
TEST_001                        00000034-R   141    (2)    141    (2)
TEST_001_X                      00000431-R   329    (7)
TEST_002                        0000001C-R   332    (7)    332    (7)
TEST_002_X                      00000210-R   412    (7)
TEST_003                        00000020-R   416    (8)    416    (8)
TEST_003_X                      000001C8-R   496    (10)
TEST_004                        00000020-R   499    (10)   499    (10)
TEST_004_X                      000006B6-R   779    (14)
TEST_005                        00000034-R   782    (14)   782    (14)
TEST_005_X                      00000676-R   1050   (16)
TEST_006                        0000001C-R   1053   (16)   1053   (16)
TEST_006_X                      00000264-R   1173   (20)
TEST_007                        00000028-R   1176   (20)   1176   (20)
TEST_007_X                      00000181-R   1245   (20)
TEST_008                        0000001C-R   1249   (21)   1249   (21)
TEST_008_X                      000003A6-R   1407   (25)
TEST_009                        00000028-R   1410   (25)   1410   (25)
TEST_009_X                      00000B96-R   1887   (32)
TEST_010                        00000028-R   1890   (32)   1890   (32)
TEST_010_X                      00000A83-R   2346   (45)
TEST_011                        0000001C-R   2349   (45)   2349   (45)
TEST_011_X                      000001D0-R   2456   (47)
TEST_012                        00000028-R   2459   (47)   2459   (47)
TEST_012_X                      00000C76-R   3052   (61)
TEST_013                        0000001C-R   3055   (61)   3055   (61)
TEST_013_X                      00000283-R   3180   (63)
```

TEST_014	00000018-R	3183	(63)	3183	(63)								
TEST_014_X	0000034D-R	3337	(67)										
TEST_015	00000024-R	3340	(67)	3340	(67)								
TEST_015_X	00000650-R	3621	(71)										
VALID_BIT	00000000-XR			#-1140	(18)	#-1306	(22)	#-1343	(23)	#-1380	(24)		
				#-1441	(25)	#-1514	(26)	#-1916	(32)	#-2082	(36)		
				#-2423	(46)	#-2430	(46)	#-2511	(48)	#-2598	(50)		
				#-2640	(51)	#-2684	(52)	#-2776	(54)	#-2826	(55)		
				#-2873	(56)	#-2971	(59)	#-3026	(60)	#-3092	(61)		
				#-3223	(63)	#-3228	(63)	#-3233	(63)	#-3266	(65)		
				#-3367	(67)								
VAR	00000000-XR			#-1004	(15)	#-1005	(15)	#-1013	(15)	#-1014	(15)		
				#-1022	(15)	#-1023	(15)	#-1031	(15)	#-1032	(15)		
				#-1040	(15)	#-1041	(15)	#-1309	(22)	#-1346	(23)		
				#-1383	(24)	#-1444	(25)	#-1517	(26)	#-1579	(27)		
				#-1643	(28)	#-1708	(29)	#-1771	(30)	#-1835	(31)		
				#-2087	(36)	#-2127	(37)	#-2158	(38)	#-2190	(39)		
				#-2193	(39)	#-2221	(40)	#-2255	(41)	#-2281	(42)		
				#-2393	(45)	#-2434	(46)	#-2515	(48)	#-2523	(48)		
				#-255	(5)	#-256	(5)	#-2601	(50)	#-2642	(51)		
				#-2644	(51)	#-2682	(52)	#-2690	(52)	#-2729	(53)		
				#-2735	(53)	#-2780	(54)	#-2829	(55)	#-2876	(56)		
				#-2975	(59)	#-3030	(60)	#-3097	(61)	#-3135	(62)		
				#-3242	(64)	#-3277	(65)	#-3313	(66)	#-3372	(67)		
				#-3401	(68)	#-3409	(68)	#-3412	(68)	#-3425	(68)		
				#-3435	(68)	#-3451	(68)	#-3463	(69)	#-3471	(69)		
				#-3473	(69)	#-3486	(69)	#-3498	(69)	#-3514	(69)		
				#-3526	(70)	#-3531	(70)	#-3533	(70)	#-3546	(70)		
				#-3558	(70)	#-3574	(70)	#-3591	(71)	#-3606	(71)		
				#-383	(7)	#-384	(7)	#-385	(7)	#-810	(14)		
				#-811	(14)	#-833	(15)	#-834	(15)	#-842	(15)		
				#-843	(15)	#-851	(15)	#-852	(15)	#-860	(15)		
				#-861	(15)	#-869	(15)	#-870	(15)	#-878	(15)		
				#-879	(15)	#-887	(15)	#-888	(15)	#-896	(15)		
				#-897	(15)	#-905	(15)	#-906	(15)	#-914	(15)		
				#-915	(15)	#-923	(15)	#-924	(15)	#-932	(15)		
				#-933	(15)	#-941	(15)	#-942	(15)	#-950	(15)		
				#-951	(15)	#-959	(15)	#-960	(15)	#-968	(15)		
				#-969	(15)	#-977	(15)	#-978	(15)	#-986	(15)		
				#-987	(15)	#-995	(15)	#-996	(15)				
VAR_PAT1	00000000-XR			#-832	(15)								
VAR_PAT10	00000000-XR			#-913	(15)								
VAR_PAT11	00000000-XR			#-922	(15)								
VAR_PAT12	00000000-XR			#-931	(15)								
VAR_PAT13	00000000-XR			#-940	(15)								
VAR_PAT14	00000000-XR			#-949	(15)								
VAR_PAT15	00000000-XR			#-958	(15)								
VAR_PAT16	00000000-XR			#-967	(15)								
VAR_PAT17	00000000-XR			#-976	(15)								
VAR_PAT18	00000000-XR			#-985	(15)								
VAR_PAT19	00000000-XR			#-994	(15)								
VAR_PAT2	00000000-XR			#-841	(15)								
VAR_PAT20	00000000-XR			#-1003	(15)								
VAR_PAT21	00000000-XR			#-1012	(15)								
VAR_PAT22	00000000-XR			#-1021	(15)								
VAR_PAT23	00000000-XR			#-1030	(15)								
VAR_PAT24	00000000-XR			#-1039	(15)								

```
VAR_PAT3                        00000000-XR     #-850   (15)
VAR_PAT4                        00000000-XR     #-859   (15)
VAR_PAT5                        00000000-XR     #-868   (15)
VAR_PAT6                        00000000-XR     #-877   (15)
VAR_PAT7                        00000000-XR     #-886   (15)
VAR_PAT8                        00000000-XR     #-895   (15)
VAR_PAT9                        00000000-XR     #-904   (15)
WRITE                           00000000-XR     #-1313  (22)  #-1449  (25)  #-1522  (26)  #-1582  (27)
                                                #-1647  (28)  #-1711  (29)  #-1775  (30)  #-1838  (31)
                                                #-2091  (36)  #-2517  (48)  #-2603  (50)  #-2645  (51)
                                                #-2686  (52)  #-2731  (53)  #-2782  (54)  #-2841  (55)
                                                #-2889  (56)  #-3033  (60)  #-3103  (61)  #-3245  (64)
                                                #-3279  (65)  #-3315  (66)  #-3377  (67)
WRITE_CHECK                     00000000-XR     #-1350  (23)  #-2693  (52)  #-2739  (53)  #-3592  (71)
WRITE_CHECKREV                  00000000-XR     #-3534  (70)  #-3607  (71)
WRITE_CHK_HIGH                  00000000-XR     #-2744  (53)
WRITE_CHK_LOW                   00000000-XR     #-2697  (52)
```

Macros Cross Reference

MACRO	SIZE	DEFINITION	REFERENCES...
$D1_BSUBT	1	152 (2)	1078 (16) 1105 (17) 1119 (18) 1153 (19) 1258 (21)
			1295 (22) 1333 (23) 1370 (24) 1424 (25) 1498 (26)
			152 (2) 1568 (27) 1631 (28) 1696 (29) 1760 (30)
			1824 (31) 185 (3) 1922 (32) 1967 (33) 1992 (34)
			2044 (35) 2078 (36) 211 (4) 2124 (37) 2155 (38)
			2187 (39) 2218 (40) 2249 (41) 2278 (42) 2302 (43)
			2327 (44) 2378 (45) 241 (5) 2419 (46) 2474 (47)
			2499 (48) 2551 (49) 2587 (50) 2629 (51) 2672 (52)
			2725 (53) 277 (6) 2771 (54) 2811 (55) 2858 (56)
			2909 (57) 2931 (58) 2958 (59) 3011 (60) 3086 (61)
			3123 (62) 3201 (63) 3238 (64) 3262 (65) 3308 (66)
			3356 (67) 3396 (68) 3459 (69) 3522 (70) 3582 (71)
			436 (8) 452 (9) 528 (10) 542 (11) 558 (12)
			574 (13) 801 (14) 827 (15)
$D1_BTEST	1	141 (2)	1053 (16) 1176 (20) 1249 (21) 141 (2) 1410 (25)
			1890 (32) 2349 (45) 2459 (47) 3055 (61) 3183 (63)
			332 (7) 3340 (67) 416 (8) 499 (10) 782 (14)
$D1_ERR	1	1471 (25)	1471 (25) 1489 (25) 1544 (26) 1559 (26) 1604 (27)
			1622 (27) 1669 (28) 1687 (28) 1733 (29) 1751 (29)
			1797 (30) 1815 (30) 1860 (31) 1878 (31)
$D1_ERR_S	1	168 (2)	1001 (15) 1010 (15) 1019 (15) 1028 (15) 1037 (15)
			1046 (15) 1098 (16) 1114 (17) 1136 (18) 1148 (18)
			1165 (19) 1208 (20) 1222 (20) 1242 (20) 1281 (21)
			1291 (21) 1329 (22) 1366 (23) 1403 (24) 1471 (25)
			1489 (25) 1544 (26) 1559 (26) 1604 (27) 1622 (27)
			1669 (28) 168 (2) 1687 (28) 1733 (29) 1751 (29)
			1797 (30) 1815 (30) 1860 (31) 1878 (31) 1936 (32)
			1949 (32) 1960 (32) 1978 (33) 198 (3) 1988 (33)
			2018 (34) 2027 (34) 2061 (35) 2071 (35) 2107 (36)
			2117 (36) 2138 (37) 2148 (37) 2169 (38) 2179 (38)
			2201 (39) 2211 (39) 2232 (40) 224 (4) 2242 (40)
			2271 (41) 2295 (42) 2320 (43) 2342 (44) 2408 (45)
			2447 (46) 246 (5) 2492 (47) 2537 (48) 2547 (48)
			2573 (49) 2583 (49) 2615 (50) 2625 (50) 2658 (51)
			266 (5) 2668 (51) 2705 (52) 2715 (52) 2752 (53)
			2762 (53) 2797 (54) 2807 (54) 2854 (55) 2902 (56)
			2927 (57) 2954 (58) 2997 (59) 3007 (59) 3048 (60)
			3119 (61) 3150 (62) 3157 (62) 3166 (62) 319 (6)
			3215 (63) 3255 (64) 3299 (65) 3329 (66) 3392 (67)
			3424 (68) 3430 (68) 3442 (68) 3485 (69) 3491 (69)
			3504 (69) 3545 (70) 3551 (70) 3564 (70) 3603 (71)
			3618 (71) 365 (7) 373 (7) 380 (7) 389 (7)
			398 (7) 409 (7) 445 (8) 462 (9) 472 (9)
			482 (9) 492 (9) 538 (10) 552 (11) 570 (12)
			586 (13) 600 (13) 614 (13) 627 (13) 640 (13)
			653 (13) 666 (13) 679 (13) 692 (13) 706 (13)
			713 (13) 741 (13) 756 (13) 769 (13) 775 (13)
			817 (14) 839 (15) 848 (15) 857 (15) 866 (15)
			875 (15) 884 (15) 893 (15) 902 (15) 911 (15)
			920 (15) 929 (15) 938 (15) 947 (15) 956 (15)

```
                                        965    (15)    974    (15)    983    (15)    992    (15)
$D1_ESCSUB      1       554    (11)     554    (11)
$D1_ESUBT       1       179    (2)      1048   (15)    1103   (16)    1117   (17)    1151   (18)    1171   (19)
                                        1293   (21)    1331   (22)    1368   (23)    1405   (24)    1496   (25)
                                        1566   (26)    1629   (27)    1694   (28)    1758   (29)    179    (2)
                                        1822   (30)    1885   (31)    1962   (32)    1990   (33)    2039   (34)
                                        204    (3)     2073   (35)    2119   (36)    2150   (37)    2181   (38)
                                        2213   (39)    2244   (40)    2273   (41)    2297   (42)    231    (4)
                                        2322   (43)    2344   (44)    2417   (45)    2454   (46)    2497   (47)
                                        2549   (48)    2585   (49)    2627   (50)    2670   (51)    2717   (52)
                                        275    (5)     2764   (53)    2809   (54)    2856   (55)    2907   (56)
                                        2929   (57)    2956   (58)    3009   (59)    3050   (60)    3121   (61)
                                        3178   (62)    3236   (63)    3260   (64)    327    (6)     3306   (65)
                                        3335   (66)    3394   (67)    3457   (68)    3520   (69)    3580   (70)
                                        3620   (71)    447    (8)     494    (9)     540    (10)    556    (11)
                                        572    (12)    777    (13)    820    (14)
$D1_ETEST       1       329    (7)      1050   (16)    1173   (20)    1245   (20)    1407   (25)    1887   (32)
                                        2346   (45)    2456   (47)    3052   (61)    3180   (63)    329    (7)
                                        3337   (67)    3621   (71)    412    (7)     496    (10)    779    (14)
$D1_PRINT_S     1       252    (5)      1100   (16)    1167   (19)    1494   (25)    1564   (26)    1627   (27)
                                        1692   (28)    1756   (29)    1820   (30)    1883   (31)    2011   (34)
                                        2037   (34)    2494   (47)    252    (5)     272    (5)     3159   (62)
                                        3168   (62)    321    (6)     3432   (68)    3447   (68)    3493   (69)
                                        3509   (69)    3553   (70)    3569   (70)
$D1_SBTTL       2       140    (2)      1052   (16)    1175   (20)    1248   (21)    140    (2)     1409   (25)
                                        1889   (32)    2348   (45)    2458   (47)    3054   (61)    3182   (63)
                                        331    (7)     3339   (67)    415    (8)     498    (10)    781    (14)
$D2_BDATA       1       141    (2)      1053   (16)    1176   (20)    1249   (21)    141    (2)     1410   (25)
                                        1890   (32)    2349   (45)    2459   (47)    3055   (61)    3183   (63)
                                        332    (7)     3340   (67)    416    (8)     499    (10)    782    (14)
$D2_BTEST       1       141    (2)      1053   (16)    1176   (20)    1249   (21)    141    (2)     1410   (25)
                                        1890   (32)    2349   (45)    2459   (47)    3055   (61)    3183   (63)
                                        332    (7)     3340   (67)    416    (8)     499    (10)    782    (14)
$D2_SBTTL       1       140    (2)      1052   (16)    1175   (20)    1248   (21)    140    (2)     1409   (25)
                                        1889   (32)    2348   (45)    2458   (47)    3054   (61)    3182   (63)
                                        331    (7)     3339   (67)    415    (8)     498    (10)    781    (14)
$DEF            1       137    (1)
$DEFINI         1       134    (1)      134    (1)     135    (1)     136    (1)
$DS_BGNMOD      1       137    (1)      137    (1)
$DS_BGNSUB      1       152    (2)      1078   (16)    1105   (17)    1119   (18)    1153   (19)    1258   (21)
                                        1295   (22)    1333   (23)    1370   (24)    1424   (25)    1498   (26)
                                        152    (2)     1568   (27)    1631   (28)    1696   (29)    1760   (30)
                                        1824   (31)    185    (3)     1922   (32)    1967   (33)    1992   (34)
                                        2044   (35)    2078   (36)    211    (4)     2124   (37)    2155   (38)
                                        2187   (39)    2218   (40)    2249   (41)    2278   (42)    2302   (43)
                                        2327   (44)    2378   (45)    241    (5)     2419   (46)    2474   (47)
                                        2499   (48)    2551   (49)    2587   (50)    2629   (51)    2672   (52)
                                        2725   (53)    277    (6)     2771   (54)    2811   (55)    2858   (56)
                                        2909   (57)    2931   (58)    2958   (59)    3011   (60)    3086   (61)
                                        3123   (62)    3201   (63)    3238   (64)    3262   (65)    3308   (66)
                                        3356   (67)    3396   (68)    3459   (69)    3522   (70)    3582   (71)
                                        436    (8)     452    (9)     528    (10)    542    (11)    558    (12)
                                        574    (13)    801    (14)    827    (15)
$DS_BGNTEST     1       141    (2)      1053   (16)    1176   (20)    1249   (21)    141    (2)     1410   (25)
                                        1890   (32)    2349   (45)    2459   (47)    3055   (61)    3183   (63)
                                        332    (7)     3340   (67)    416    (8)     499    (10)    782    (14)
$DS_BITDEF      2       133    (1)      133    (1)
```

$DS_BNOPER	1	1483	(25)	1483	(25)	1616	(27)	1681	(28)	1745	(29)	1809	(30)		
				1872	(31)										
$DS_BNQUICK	1	1484	(25)	1484	(25)	1556	(26)	1617	(27)	1682	(28)	1746	(29)		
				1810	(30)	1873	(31)								
$DS_BQUICK	1	3173	(62)	3173	(62)										
$DS_BREAK	1	329	(7)	1050	(16)	1173	(20)	1245	(20)	1407	(25)	1887	(32)		
				2346	(45)	2456	(47)	3052	(61)	3180	(63)	329	(7)		
				3337	(67)	3621	(71)	412	(7)	496	(10)	779	(14)		
$DS_CKLOOP	1	171	(2)	1002	(15)	1011	(15)	1020	(15)	1029	(15)	1038	(15)		
				1047	(15)	1115	(17)	1137	(18)	1149	(18)	1168	(19)		
				1209	(20)	1223	(20)	1243	(20)	1282	(21)	1292	(21)		
				1330	(22)	1367	(23)	1404	(24)	1472	(25)	1490	(25)		
				1545	(26)	1560	(26)	1605	(27)	1623	(27)	1670	(28)		
				1688	(28)	171	(2)	1734	(29)	1752	(29)	1798	(30)		
				1816	(30)	1861	(31)	1879	(31)	1937	(32)	1951	(32)		
				1961	(32)	1979	(33)	1989	(33)	201	(3)	2019	(34)		
				2028	(34)	2062	(35)	2072	(35)	2108	(36)	2118	(36)		
				2139	(37)	2149	(37)	2170	(38)	2180	(38)	2202	(39)		
				2212	(39)	2233	(40)	2243	(40)	227	(4)	2272	(41)		
				2296	(42)	2321	(43)	2343	(44)	2409	(45)	2448	(46)		
				2495	(47)	253	(5)	2538	(48)	2548	(48)	2574	(49)		
				2584	(49)	2616	(50)	2626	(50)	2659	(51)	2669	(51)		
				2706	(52)	2716	(52)	273	(5)	2753	(53)	2763	(53)		
				2798	(54)	2808	(54)	2855	(55)	2906	(56)	2928	(57)		
				2955	(58)	2998	(59)	3008	(59)	3049	(60)	3120	(61)		
				3151	(62)	3160	(62)	3169	(62)	3216	(63)	3256	(64)		
				3300	(65)	3330	(66)	3393	(67)	3436	(68)	3448	(68)		
				3499	(69)	3510	(69)	3559	(70)	3570	(70)	3604	(71)		
				3619	(71)	366	(7)	374	(7)	381	(7)	390	(7)		
				399	(7)	410	(7)	446	(8)	463	(9)	473	(9)		
				483	(9)	493	(9)	539	(10)	555	(11)	571	(12)		
				587	(13)	601	(13)	615	(13)	628	(13)	641	(13)		
				654	(13)	667	(13)	680	(13)	693	(13)	707	(13)		
				720	(13)	742	(13)	757	(13)	770	(13)	776	(13)		
				818	(14)	840	(15)	849	(15)	858	(15)	867	(15)		
				876	(15)	885	(15)	894	(15)	903	(15)	912	(15)		
				921	(15)	930	(15)	939	(15)	948	(15)	957	(15)		
				966	(15)	975	(15)	984	(15)	993	(15)				
$DS_CLRVEC_S	1	178	(2)	178	(2)	203	(3)	230	(4)	2496	(47)	291	(6)		
				325	(6)	326	(6)								
$DS_CVTREG_S	1	247	(5)	247	(5)	267	(5)								
$DS_DSADEF	5	134	(1)	134	(1)										
$DS_ENDDATA	1	141	(2)	1053	(16)	1176	(20)	1249	(21)	141	(2)	1410	(25)		
				1890	(32)	2349	(45)	2459	(47)	3055	(61)	3183	(63)		
				332	(7)	3340	(67)	416	(8)	499	(10)	782	(14)		
$DS_ENDMOD	1	3622	(71)	3622	(71)										
$DS_ENDSUB	1	179	(2)	1048	(15)	1103	(16)	1117	(17)	1151	(18)	1171	(19)		
				1293	(21)	1331	(22)	1368	(23)	1405	(24)	1496	(25)		
				1566	(26)	1629	(27)	1694	(28)	1758	(29)	179	(2)		
				1822	(30)	1885	(31)	1962	(32)	1990	(33)	2039	(34)		
				204	(3)	2073	(35)	2119	(36)	2150	(37)	2181	(38)		
				2213	(39)	2244	(40)	2273	(41)	2297	(42)	231	(4)		
				2322	(43)	2344	(44)	2417	(45)	2454	(46)	2497	(47)		
				2549	(48)	2585	(49)	2627	(50)	2670	(51)	2717	(52)		
				275	(5)	2764	(53)	2809	(54)	2856	(55)	2907	(56)		
				2929	(57)	2956	(58)	3009	(59)	3050	(60)	3121	(61)		
				3178	(62)	3236	(63)	3260	(64)	327	(6)	3306	(65)		

				3335	(66)	3394	(67)	3457	(68)	3520	(69)	3580	(70)
				3620	(71)	447	(8)	494	(9)	540	(10)	556	(11)
				572	(12)	777	(13)	820	(14)				
$DS_ENDTEST	1	329	(7)	1050	(16)	1173	(20)	1245	(20)	1407	(25)	1887	(32)
				2346	(45)	2456	(47)	3052	(61)	3180	(63)	329	(7)
				3337	(67)	3621	(71)	412	(7)	496	(10)	779	(14)
$DS_ENVDEF	2	137	(1)	137	(1)								
$DS_ERRHARD_S	1	166	(2)	1008	(15)	1017	(15)	1026	(15)	1035	(15)	1044	(15)
				1097	(16)	1112	(17)	1134	(18)	1146	(18)	1164	(19)
				1206	(20)	1220	(20)	1240	(20)	1279	(21)	1289	(21)
				1326	(22)	1363	(23)	1400	(24)	1469	(25)	1487	(25)
				1542	(26)	1557	(26)	1602	(27)	1620	(27)	166	(2)
				1667	(28)	1685	(28)	1731	(29)	1749	(29)	1795	(30)
				1813	(30)	1858	(31)	1876	(31)	1934	(32)	1947	(32)
				1958	(32)	196	(3)	1976	(33)	1986	(33)	2017	(34)
				2025	(34)	2059	(35)	2069	(35)	2105	(36)	2115	(36)
				2136	(37)	2146	(37)	2167	(38)	2177	(38)	2199	(39)
				2209	(39)	222	(4)	2230	(40)	2240	(40)	2269	(41)
				2293	(42)	2318	(43)	2340	(44)	2406	(45)	2445	(46)
				245	(5)	2491	(47)	2535	(48)	2545	(48)	2571	(49)
				2581	(49)	2613	(50)	2623	(50)	265	(5)	2656	(51)
				2666	(51)	2703	(52)	2713	(52)	2750	(53)	2760	(53)
				2795	(54)	2805	(54)	2852	(55)	2900	(56)	2926	(57)
				2952	(58)	2995	(59)	3004	(59)	3046	(60)	3117	(61)
				3148	(62)	3156	(62)	3165	(62)	318	(6)	3213	(63)
				3253	(64)	3297	(65)	3327	(66)	3390	(67)	3422	(68)
				3429	(68)	3441	(68)	3483	(69)	3490	(69)	3503	(69)
				3543	(70)	3550	(70)	3563	(70)	3601	(71)	3616	(71)
				363	(7)	371	(7)	378	(7)	387	(7)	396	(7)
				407	(7)	443	(8)	460	(9)	470	(9)	480	(9)
				490	(9)	536	(10)	550	(11)	568	(12)	584	(13)
				598	(13)	612	(13)	625	(13)	638	(13)	651	(13)
				664	(13)	677	(13)	690	(13)	704	(13)	717	(13)
				739	(13)	754	(13)	767	(13)	773	(13)	815	(14)
				837	(15)	846	(15)	855	(15)	864	(15)	873	(15)
				882	(15)	891	(15)	900	(15)	909	(15)	918	(15)
				927	(15)	936	(15)	945	(15)	954	(15)	963	(15)
				972	(15)	981	(15)	990	(15)	999	(15)		
$DS_ESCAPE	1	554	(11)	554	(11)								
$DS_HPODEF	2	135	(1)	135	(1)								
$DS_PAGE	1	138	(1)	1051	(16)	1174	(20)	1246	(20)	138	(1)	1408	(25)
				1888	(32)	2347	(45)	2457	(47)	3053	(61)	3181	(63)
				330	(7)	3338	(67)	413	(7)	497	(10)	780	(14)
$DS_PRINTB_S	1	1494	(25)	1494	(25)	1564	(26)	1627	(27)	1692	(28)	1756	(29)
				1820	(30)	1883	(31)	2494	(47)	3158	(62)	3167	(62)
$DS_PRINTX_S	1	251	(5)	1099	(16)	1166	(19)	2011	(34)	2037	(34)	251	(5)
				271	(5)	320	(6)	3431	(68)	3445	(68)	3492	(69)
				3507	(69)	3552	(70)	3567	(70)				
$DS_RH750_DEF	1	136	(1)	136	(1)								
$DS_SBTTL	1	140	(2)	1052	(16)	1175	(20)	1248	(21)	140	(2)	1409	(25)
				1889	(32)	2348	(45)	2458	(47)	3054	(61)	3182	(63)
				331	(7)	3339	(67)	415	(8)	498	(10)	781	(14)
$DS_SECDEF	1	83	(1)	83	(1)								
$DS_SETIPL_S	1	2309	(43)	2309	(43)	2313	(43)	2331	(44)	2335	(44)		
$DS_SETVEC_S	1	161	(2)	161	(2)	191	(3)	216	(4)	2485	(47)	279	(6)
				280	(6)	281	(6)						
$DS_WAITUS_S	1	1451	(25)	1451	(25)	1524	(26)	1584	(27)	1649	(28)	1713	(29)

				1777	(30)	1840	(31)	2054	(35)	2093	(36)	2311	(43)		
				2333	(44)	3247	(64)	3281	(65)	3317	(66)				
$EQU	1	137	(1)	133	(1)	137	(1)								
$EQULS1	1	137	(1)	137	(1)										
$EQULST	1	137	(1)	137	(1)										
$GBLINI	2	133	(1)	133	(1)	137	(1)								
$PUSHADR	1	168	(2)	1001	(15)	1010	(15)	1019	(15)	1028	(15)	1037	(15)		
				1046	(15)	1098	(16)	1114	(17)	1136	(18)	1148	(18)		
				1165	(19)	1208	(20)	1222	(20)	1242	(20)	1281	(21)		
				1291	(21)	1329	(22)	1366	(23)	1403	(24)	1451	(25)		
				1471	(25)	1489	(25)	1524	(26)	1544	(26)	1559	(26)		
				1584	(27)	1604	(27)	1622	(27)	1649	(28)	1669	(28)		
				168	(2)	1687	(28)	1713	(29)	1733	(29)	1751	(29)		
				1777	(30)	1797	(30)	1815	(30)	1840	(31)	1860	(31)		
				1878	(31)	1936	(32)	1949	(32)	1960	(32)	1978	(33)		
				198	(3)	1988	(33)	2018	(34)	2027	(34)	2054	(35)		
				2061	(35)	2071	(35)	2093	(36)	2107	(36)	2117	(36)		
				2138	(37)	2148	(37)	2169	(38)	2179	(38)	2201	(39)		
				2211	(39)	2232	(40)	224	(4)	2242	(40)	2271	(41)		
				2295	(42)	2311	(43)	2320	(43)	2333	(44)	2342	(44)		
				2408	(45)	2447	(46)	246	(5)	2492	(47)	250	(5)		
				2537	(48)	2547	(48)	2573	(49)	2583	(49)	2615	(50)		
				2625	(50)	2658	(51)	266	(5)	2668	(51)	270	(5)		
				2705	(52)	2715	(52)	2752	(53)	2762	(53)	2797	(54)		
				2807	(54)	2854	(55)	2902	(56)	2927	(57)	2954	(58)		
				2997	(59)	3007	(59)	3048	(60)	3119	(61)	3150	(62)		
				3157	(62)	3166	(62)	319	(6)	3215	(63)	3247	(64)		
				3255	(64)	3281	(65)	3299	(65)	3317	(66)	3329	(66)		
				3392	(67)	3424	(68)	3430	(68)	3442	(68)	3485	(69)		
				3491	(69)	3504	(69)	3545	(70)	3551	(70)	3564	(70)		
				3603	(71)	3618	(71)	365	(7)	373	(7)	380	(7)		
				389	(7)	398	(7)	409	(7)	445	(8)	462	(9)		
				472	(9)	482	(9)	492	(9)	538	(10)	552	(11)		
				570	(12)	586	(13)	600	(13)	614	(13)	627	(13)		
				640	(13)	653	(13)	666	(13)	679	(13)	692	(13)		
				706	(13)	719	(13)	741	(13)	756	(13)	769	(13)		
				775	(13)	817	(14)	839	(15)	848	(15)	857	(15)		
				866	(15)	875	(15)	884	(15)	893	(15)	902	(15)		
				911	(15)	920	(15)	929	(15)	938	(15)	947	(15)		
				956	(15)	965	(15)	974	(15)	983	(15)	992	(15)		
$VIELD	1			137	(1)										
$VIELD1	1	137	(1)	137	(1)										
ERRPREP	2	87	(1)	1073	(16)	1193	(20)	1210	(20)	1227	(20)	1268	(21)		
				1433	(25)	1463	(25)	1504	(26)	1536	(26)	1576	(27)		
				1596	(27)	1661	(28)	1725	(29)	1789	(30)	1852	(31)		
				1911	(32)	2045	(35)	2482	(47)	3110	(61)	3141	(62)		
				3204	(63)	3271	(65)	3359	(67)	358	(7)	3583	(71)		
				367	(7)	375	(7)	382	(7)	391	(7)	400	(7)		
				431	(8)	525	(10)	804	(14)	831	(15)				
EXPECT_INTER	1	118	(1)												
NO_INTER	1	125	(1)												

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	28	00:00:00.08	00:00:00.22
Command processing	87	00:00:00.22	00:00:00.63
Pass 1	2385	00:00:43.27	00:00:56.45
Symbol table sort	3	00:00:00.80	00:00:00.82
Pass 2	2102	00:00:11.78	00:00:15.01
Symbol table output	4	00:00:00.27	00:00:00.42
Psect synopsis output	3	00:00:00.05	00:00:00.05
Cross-reference output	599	00:00:07.80	00:00:08.80
Assembler run totals	5212	00:01:04.28	00:01:22.41

The working set limit was 2048 pages.
764206 bytes (1493 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 586 non-local and 355 local symbols.
3623 source lines were read in Pass 1, producing 148 object records in Pass 2.
84 pages of virtual memory were used to define 52 macros.

Macro library statistics

Macro library name	Macros defined
SYS$COMMON: SYSLIB DIAG.MLB;953	41
SYS$COMMON: SYSLIB STARLET.MLB;2	6
TOTALS (all libraries)	47

556 GETS were required to define 47 macros.

There were no errors, warnings or information messages.

```
MACRO/CRO/LIS=ECCAA2.SEQ ECCAA2
```

Table of contents

(1)	81	DECLARATIONS
(2)	149	TEST 16: SILO TEST
(4)	331	TEST 17: INTERRUPT TESTS
(19)	1057	TEST 18: ATTENTION SUMMARY REGISTER TESTS
(21)	1125	TEST 19: IGNORE BYTE COUNTER MODE TEST
(25)	1323	TEST 20: RH11-TB TESTS

```
0000     1 .TITLE  ECCAA_3 RH750 REPAIR LEVEL MODULE 3
0000     2 .IDENT  /1.8/
0000     3
0000     4 ;
0000     5 ; COPYRIGHT (C) 1980,1983
0000     6 ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
0000     7 ;
0000     8 ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON  A
0000     9 ; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLU-
0000    10 ; SION OF THE ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE,  OR  ANY
0000    11 ; OTHER  COPIES  THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE
0000    12 ; AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON  SUCH  SYSTEM
0000    13 ; AND  TO  ONE WHO AGREES TO THESE LICENSE TERMS.  TITLE TO AND
0000    14 ; OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.
0000    15 ;
0000    16 ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
0000    17 ; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
0000    18 ; EQUIPMENT CORPORATION.
0000    19 ;
0000    20 ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR  RELIABILITY  OF
0000    21 ; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
0000    22 ;
0000    23
0000    24 ;++
0000    25 ; FACILITY: STAR DIAGNOSTIC LIBRARY
0000    26 ;
0000    27 ; ABSTRACT:
0000    28 ;
0000    29 ; THIS PROGRAM VERIFIES THE MASS BUS ADAPTER FUNCTIONALITY
0000    30 ;
0000    31 ; ENVIRONMENT: DIAGNOSTIC SUPERVISOR
0000    32 ;
0000    33 ; REVISION HISTORY:
0000    34 ;
0000    35 ;       1.0     Initial Release
0000    36 ;       1.1     Don Monroe      January 1981
0000    37 ;               Test 20
0000    38 ;               Sub 2 - Fixed to skip if other drives are present.
0000    39 ;               Sub 6 - Changed 500 usec wait to 600 usec.
0000    40 ;               Sub 8 thru 14 - Modified data buffer initial data pattern
0000    41 ;                 to account for the fact that the MBE only has a 16 bit
0000    42 ;                 data buffer register.
0000    43 ;               Sub 19,20 - Added these subtests to perform WRITECHECK
0000    44 ;                 FORWARD and REVERSE with shifting data pattern.
0000    45 ;       1.2     Dom Andella     May 1981
0000    46 ;               Added two subtests to test 10 to test the DT Busy
0000    47 ;               One-Shot and timing components.
0000    48 ;       1.3     Dan Milleville  June 1981
0000    49 ;               Changed ASCIC messages for MDP numbers from 0 - 7 to 1 - 8,
0000    50 ;               and changed message printed when MBE selected but not present
0000    51 ;               to a hard error.
0000    52 ;       1.4     Dan Milleville  June 1981
0000    53 ;               Changed the timing in Test 10 Subtest 12 to 1 MS., and Test
0000    54 ;               10 Subtest 13 to 300 US.
0000    55 ;       1.5     Dan Milleville  Jan 1982
0000    56 ;               Added Subtest 5 to Test 1 - checks that the Unit Under Test
0000    57 ;               (UUT) responds to only 1 address between 40F28800(X) and
```

```
                0000     58 ;                40F2F800(X), in steps of 1000(X)
                0000     59 ;                Added Subtest 1 to Test 6 - checks that there is no "bit cross-
                0000     60 ;                talk" between the 100(X) Map Registers, i.e. writing to 1
                0000     61 ;                register does not result in any other bits in any other regis-
                0000     62 ;                ters becoming set.
                0000     63 ;        1.6     Dan Milleville  Nov 1982
                0000     64 ;                Because of an ECO, a TSTL had to be added to T1S5 in the routine
                0000     65 ;                that tried to clear all registers.
                0000     66 ;
                0000     67 ;        1.7     Ricardo De Andrade May 1983
                0000     68 ;                Fixed Test 1, used to fail with two MBA's because of a bad
                0000     69 ;                call to the P_TABLE when the UUT was RH1 or RH2 .
                0000     70 ;
                0000     71 ;
                0000     72 ;        1.8     Susan E. Hutcheson                      January 1985
                0000     73 ;                Test 12 subtest 10 forces a double bit error but does not clean
                0000     74 ;                up error status flags in CRS0. These bits must be cleared man-
                0000     75 ;                ually. Search on  1.8 .
                0000     76 ;--
```

```
0000     77
0000     78 ;--
0000     79
0000     80
0000     81 .SBTTL  DECLARATIONS
0000     82
0000     83 ;
0000     84 ; LIBRARY FILES:
0000     85 ;
0000     86 .LIBRARY        /SYS$LIBRARY:DIAG/
0000     87
0000     88 $DS_SECDEF      <ALL,MBE>
0000     89 ;++
0000     90 ; MACROS UNIQUE TO THIS PROGRAM
0000     91 ;--
0000     92         .MACRO  ERRPREP, MREG_NAME,MREG_NO ,MREG_STRING ,SUFFIX = BLANK
0000     93 ;++
0000     94 ; THIS MACRO IS USED TO SET UP ERROR DISPLAY PARAMETERS
0000     95 ; FOR THE PRINT ERROR ROUTINE
0000     96 ;
0000     97 ;       MREG_NAME IS A POINTER TO THE STRING DESCRIBING THE REGISTER
0000     98 ;       WHICH PRECEEDS THE ERROR CONDITION
0000     99 ;
0000    100 ;       MREG_NO IS THE RH750 REGISTER NUMBER UNDER TEST.
0000    101 ;
0000    102 ;       MREG_STRING IS A POINTER TO THE STRING CONTAINING THE BIT
0000    103 ;       MNEMONICS OF THE REGISTER UNDER TEST.
0000    104 ;
0000    105 ;       SUFFIX IS A POINTER TO THE SUFFIX APPENDED TO THE ERROR REPORT.
0000    106 ;--
0000    107         .IF     NB MREG_NAME
0000    108                 MOVAL   MREG_NAME,REG_NAME
0000    109         .ENDC
0000    110         .IF     NB MREG_NO
0000    111                 MOVZBL  #MREG_NO,REG_NO
0000    112         .ENDC
0000    113         .IF     NB MREG_STRING
0000    114                 MOVAL   MREG_STRING,REG_STRING
0000    115         .ENDC
0000    116         .IF     DF SUFFIX
0000    117                 MOVAL   SUFFIX,SUFFIX_PTR
0000    118         .ENDC
0000    119         .ENDM
0000    120 ;++
0000    121 ;
0000    122 ; THE FOLLOWING MACROS ARE USED IN THE TESTING OF RH750 INTERRUPTS
0000    123 ;
0000    124 ;--
0000    125 .MACRO  EXPECT_INTER
0000    126                 MTPR    #0,#IPLR
0000    127                 MOVB    #1,EXP_INTERRUPT
0000    128                 CLRB    INT_OCCURRED
0000    129                 BISL    #INTERRUPT_EN,CR(R2)
0000    130 .ENDM
0000    131
0000    132 .MACRO  NO_INTER
0000    133         CLRB    EXP_INTERRUPT
```

```
0000   134          CLRB    INT_OCCURRED
0000   135 .ENDM
0000   136
0000   137 ;
0000   138 ; EQUATED SYMBOLS
0000   139 ;
0000   140 $DS_BITDEF
0000   141 $DS_DSADEF
0000   142
0000   143 $DS_BGNMOD      CEP_REPAIR,16
0000       ;;;     Macro-32 Diagnostic macro library Version 7.0 "DIAG.MLB (186)"
0000   144 $DS_PAGE
```

```
                    0000                .SAVE   LOCAL_BLOCK
                    0000                .PSECT  $ABS$,ABS
          00000000  FE70                .=0
                    0000                .IIF    NB,HP$Q_DEVICE, HP$Q_DEVICE:
          00000008  0000                .IIF    NB,.BLKQ,       .BLKQ   1
                    0008                .IIF    NB,HP$W_SIZE,   HP$W_SIZE:
          0000000A  0008                .IIF    NB,.BLKW,       .BLKW   1
                    000A                .IIF    NB,HP$B_FLAGS,  HP$B_FLAGS:
          0000000B  000A                .IIF    NB,.BLKB,       .BLKB   1
                    000B                .IIF    NB,HP$B_DRIVE,  HP$B_DRIVE:
          0000000C  000B                .IIF    NB,.BLKB,       .BLKB   1
                    000C                .IIF    NB,HP$T_DEVICE, HP$T_DEVICE:
          00000018  000C                .IIF    NB,.BLKB,       .BLKB   12
                    0018                .IIF    NB,HP$A_DEVICE, HP$A_DEVICE:
          0000001C  0018                .IIF    NB,.BLKL,       .BLKL   1
                    001C                .IIF    NB,HP$A_DVA,    HP$A_DVA:
          00000020  001C                .IIF    NB,.BLKL,       .BLKL   1
                    0020                .IIF    NB,HP$A_LINK,   HP$A_LINK:
          00000024  0020                .IIF    NB,.BLKL,       .BLKL   1
                    0024                .IIF    NB,HP$W_VECTOR, HP$W_VECTOR:
          00000026  0024                .IIF    NB,.BLKW,       .BLKW   1
                    0026                .IIF    NB,HP$T_TYPE,   HP$T_TYPE:
          00000032  0026                .IIF    NB,.BLKB,       .BLKB   12
                    0032                .IIF    NB,HP$A_DEPENDENT,      HP$A_DEPENDENT:
                    0000                .RESTORE
                    0000                .SAVE   LOCAL_BLOCK
                    0000                .PSECT  $ABS$,ABS
          00000032  FE70                .=HP$A_DEPENDENT
                    0032                .IIF    NB,HP$B_RH750_BR,       HP$B_RH750_BR:
                    0032                .IIF    NB,RH750$B_BR,  RH750$B_BR:
          00000033  0032                .IIF    NB,.BLKB,       .BLKB   1
                    0033                .IIF    NB,HP$K_RH750_LEN,      HP$K_RH750_LEN:
                    0033                .IIF    NB,RH750$K_LEN, RH750$K_LEN:
                    0000                .RESTORE
                    0000    148
                    0000                .SBTTL  TEST 16:  SILO TEST
                00000000                .PSECT  TEST_016, PAGE, NOWRT
                    000C
                    000C    150 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                    000C        DATA_016:
          00000000  000C                .LONG   0                       ; TEST ARGUMENT TABLE TERMINATOR
                    0010        TEST_016::
              0000  0010                .WORD    M<>                    ; ENTRY MASK
                    0012    151 ;**
                    0012    152 ; TEST DESCRIPTION:
                    0012    153 ;
                    0012    154 ;       THIS TEST IS COMPRISED OF TWO SUBTESTS.
                    0012    155 ;
                    0012    156 ;       SUBTEST ONE VERIFIES THAT DATA IN THE SILO IS NOT TRANSFERRED
                    0012    157 ;       TO MEMORY WHEN AN ERROR OCCURS DURING A TRANSFER. THIS IS
                    0012    158 ;       ACCOMPLISHED BY FIRST PERFORMING A WRITE-TO-DRIVE FUNCTION
                    0012    159 ;       WITH THE MB DATA PARITY BIT INVERTED. NEXT, A READ-FROM-DRIVE
                    0012    160 ;       IS ATTEMPTED REFERENCING THE BAD DATA. AS A RESULT, NO DATA
                    0012    161 ;       SHOULD BE TRANSFERRED TO MEMORY.
                    0012    162 ;
                    0012    163 ;       SUBTEST TWO VERIFIES THAT DATA IN THE SILO IS NOT TRANSFERRED
                    0012    164 ;       TO MEMORY AS A RESULT OF AN INVALID MAP ERROR. THIS IS
```

```
                                0012   165 ;       ACCOMPLISHED BY SETTING UP A TRANSFER TO OCCUR ACROSS TWO
                                0012   166 ;       PAGES IN MEMORY, WITH THE SECOND PAGE BEING INVALID. THE SILO
                                0012   167 ;       DATA FROM THE FIRST PAGE SHOULD BE SUCCESSFULLY TRANSFERRED TO MEMORY
                                0012   168 ;       WHILE THE SILO DATA FROM THE SECOND PAGE SHOULD NOT BE TRANSFERRED.
                                0012   169 ;
                                0012   170 ;--
                                0012   171 $DS_BGNSUB
                                0012        T16_S1::
00000000'9F   00000000'EF   FA  0012                    CALLG   $S$, a#DS$BGNSUB
                    FFE0'   30  001D   172          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
        52  00000000'EF     D0  0020   173          MOVL    RH_CUR_ADR,R2           ; MOVE RH750 ADDRESS TO R2
  0000'C2   00000000'8F     C8  0027   174          BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
        56  00000000'EF     DE  0030   175          MOVAL   MIOBUFFER,R6            ; MOVE ADDRESS OF BUFFER TO R6
              55   56       D0  0037   176          MOVL    R6,R5                   ; GET COPY OF BUFFER ADDRESS
         57   56    00'8F    78  003A   177          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
        57  00000000'8F     C8  003F   178          BISL    #VALID_BIT,R7           ; SET V BIT IN MAP ENTRY
                    50      D4  0046   179          CLRL    R0                      ; CLEAR MAP INDEX
  5A    52  00000000'8F     C1  0048   180          ADDL3   #MAP_OFFSET,R2,R10      ; R10 NOW POINTS TO MAP REG 0
            6A40    57      D0  0050   181 10$:     MOVL    R7,(R10) R0             ; WRITE MAP REGISTER R0
    F4  50  000000FF  8F    F3  0054   182          AOBLEQ  #255,R0,10$             ; DO FOR ALL MAP REGISTERS
              0000'C2       D4  005C   183 20$:     CLRL    VAR(R2)                 ; CLEAR VIRTUAL ADDRESS
              85    00      D2  0060   184          MCOML   #0,(R5)+                ; WRITE 1'S PATTERN INTO FIRST
              65    00      D2  0063   185          MCOML   #0,(R5)                 ; 8 BYTES OF BUFFER
            0000'C2   06    CE  0066   186          MNEGL   #6,BCR(R2)              ; SET BYTE COUNT = 6
  0000'C2   00000000'8F     C8  006B   187          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
  0000'C2   00000000'8F     C8  0074   188          BISL    #INVRT_MB_DPAR,DR(R2)   ; INVERT MB DATA PARITY
        50  00000000'EF     D0  007D   189          MOVL    DRIVE0,R0               ; GET ADDRESS OF PSEUDO DRIVE
        60  00000000'8F     D0  0084   190          MOVL    #WRITE,(R0)             ; ISSUE WRITE COMMAND TO DRIVE
  0000'C2   00000000'8F     C8  008B   191          BISL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
  0000'C2   00000000'8F     C8  0094   192          BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
  0000'C2   00000000'8F     CA  009D   193          BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
  0000'C2   00000000'8F     C8  00A6   194          BISL    #DT_ABORT,CR(R2)        ; FORCE ABORT
  0000'C2   00000000'8F     C8  00AF   195          BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL END OF TRANSFER
  0000'C2   00000000'8F     CA  00B8   196          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
                              00C1   197          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                              00C1   198                  FMT_STATUS_REG,-
                              00C1   199                  UNEXPECTED
00000000'EF   00000000'EF   DE  00C1                    MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A  00CC                    MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  00D3                    MOVAL   FMT_STATUS_REG,REG_STRING
              00000000'8F   D0  00DE   200          MOVL    #DATA_XFER_DONE!-       ; SET UP EXPECTED STATUS
                        54      00E4   201                  DATA_XFER_ABRT,R4
        54  00000000'EF     C8  00E5   202          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
            53    0000'C2   D0  00EC   203          MOVL    SR(R2),R3               ; READ STATUS REGISTER
                  54    53  D1  00F1   204          CMPL    R3,R4                   ; IS STATUS AS EXPECTED
                        1B  13  00F4   205          BEQL    30$                     ; BRANCH IF STATUS IS OK
                            00F6   206          $DS_ERRHARD_S   #1,LUN,-            ; PRINT ERROR
                            00F6   207                          ARRAY_MSG54,-
                            00F6   208                          PRINT_SBE
              00000000'EF   DF  00F6                        PUSHAL  PRINT_SBE
              00000000'EF   DF  00FC                        PUSHAL  ARRAY_MSG54
              00000000'EF   DD  0102                        PUSHL   LUN
                        01  DD  0108                       PUSHL   #1
        00000000'9F     04  FB  010A                       CALLS   #$$M, a#DS$ERRHARD
                            0111   209 30$:     $DS_CKLOOP      20$                 ; SCOPE LOOP
00000000'9F     FF47  CF    FA  0111                        CALLG   20$, a#DS$CKLOOP
        57  00000000'EF     D0  011A   210          MOVL    DRIVE0,R7               ; GET ADDRESS OF DRIVE0
```

```
                0000'C2   00000000'8F   CA   0121   211 40$:     BICL    #DT_ABORT,CR(R2)          ; CLEAR ABORT
                    0000'C2           D4   012A   212            CLRL    VAR(R2)                   ; CLEAR VAR POINTS TO BYTE ZERO
                00000000'EF           7C   012E   213            CLRQ    MIOBUFFER                 ; CLEAR FIRST 8 BYTES OF BUFFER
                00000008'EF           7C   0134   214            CLRQ    MIOBUFFER + 8             ; CLEAR SECOND QUADWORD OF BUFFER
                0000'C2   00000064 8F   CE   013A   215            MNEGL   #100,BCR(R2)              ; SET BCR TO 100
                     67   00000000'8F   D0   0143   216            MOVL    #READ,(R7)                ; ISSUE READ COMMAND
                0000'C2   00000000'8F   C8   014A   217            BISL    #SIM_OCC,DR(R2)           ; ASSERT OCCUPIED
                                   50   D4   0153   218            CLRL    R0                        ; CLEAR SCLK COUNTER
                0000'C2   00000000'8F   C8   0155   219 50$:      BISL    #SIM_SCLK,DR(R2)          ; ASSERT SCLK
                0000'C2   00000000'8F   CA   015E   220            BICL    #SIM_SCLK,DR(R2)          ; DEASSERT SCLK
                           EA 50   05   F3   0167   221            AOBLEQ  #5,R0,50$                 ; ASSERT SCLK 6 TIMES
                                          016B   222            $DS_WAITUS_S    #10               ; WAIT FOR SILO TO EMPTY
                                   00   DD   016B                   PUSHL   #0
                                   0A   DD   016D                   PUSHL   #10
                      00000000'9F   02   FB   016F                   CALLS   #2, @#DS$WAITUS
                      53   00000008'EF   D0   0176   223            MOVL    MIOBUFFER + 8,R3          ; READ FIRST HALF OF QUADWORD
                      54   0000000C'EF   D0   017D   224            MOVL    MIOBUFFER + 12,R4         ; READ SECOND HALF OF QUADWORD
                                   55   D4   0184   225            CLRL    R5                        ; SET UP EXPECTED VALUES
                                   56   D4   0186   226            CLRL    R6                        ; SET UP SECOND HALF OF QUADWORD
                               55   53   D1   0188   227            CMPL    R3,R5                     ; COMPARE FIRST HALVES OF QUADWORD
                                   05   12   018B   228            BNEQ    60$                       ; IF ERROR BRANCH TO REPORT
                               56   54   D1   018D   229            CMPL    R4,R6                     ; COMPARE SECOND HALVES OF QUADWORD
                                   35   13   0190   230            BEQL    70$                       ; SKIP IF CORRECT
                                          0192   231 60$:      $DS_ERRHARD_S   #2,LUN,-          ; REPORT ERROR
                                          0192   232                            ARRAY_MSG61
                                   00   DD   0192                   PUSHL   #0
                      00000000'EF       DF   0194                   PUSHAL  ARRAY_MSG61
                      00000000'EF       DD   019A                   PUSHL   LUN
                                   02   DD   01A0                   PUSHL   #2
                00000000'9F        04   FB   01A2                   CALLS   #$$M, @#DS$ERRHARD
                5B   00000008'EF        DE   01A9   233            MOVAL   MIOBUFFER +8,R11          ; GET ADDRESS OF SECOND QW OF BUFFER
                                          01B0   234            $DS_PRINTX_S    FMT_SILOERROR,-   ; PRINT PARAMETERS
                                          01B0   235                            R6,R5,R4,R3,R11
                                   5B   DD   01B0                   PUSHL   R11
                                   53   DD   01B2                   PUSHL   R3
                                   54   DD   01B4                   PUSHL   R4
                                   55   DD   01B6                   PUSHL   R5
                                   56   DD   01B8                   PUSHL   R6
                      00000000'EF       9F   01BA                   PUSHAB  FMT_SILOERROR
                00000000'9F        06   FB   01C0                   CALLS   #$$N, @#DS$PRINTX
           0000'C2   00000000'8F        C8   01C7   236 70$:      BISL    #SIM_EBL,DR(R2)           ; SET END OF BLOCK
           0000'C2   00000000'8F        CA   01D0   237            BICL    #SIM_EBL,DR(R2)           ; CLEAR EBL
                     53   0000'C2        D0   01D9   238            MOVL    DR(R2),R3                 ; READ DIAGNOSTIC REGISER
                       24 53   11        E1   01DE   239            BBC     #17,R3,80$                ; BRANCH IF EXCEPTION IS CLEAR
                                          01E2   240            $DS_ERRHARD_S   #3,LUN,-          ; REPORT ERROR
                                          01E2   241                            ARRAY_MSG4
                                   00   DD   01E2                   PUSHL   #0
                      00000000'EF       DF   01E4                   PUSHAL  ARRAY_MSG4
                      00000000'EF       DD   01EA                   PUSHL   LUN
                                   03   DD   01F0                   PUSHL   #3
                00000000'9F        04   FB   01F2                   CALLS   #$$M, @#DS$ERRHARD
                                          01F9   242            $DS_PRINTX_S    FMT_EXCP_ERR      ; REPORT EXCEPTION ERROR
                      00000000'EF       9F   01F9                   PUSHAB  FMT_EXCP_ERR
                00000000'9F        01   FB   01FF                   CALLS   #$$N, @#DS$PRINTX
                                          0206   243 80$:      $DS_CKLOOP      40$               ; SCOPE LOOP?
00000000'9F   FF17 CF        FA   0206                   CALLG   40$, @#DS$CKLOOP
                                          020F   244 $DS_ENDSUB
```

```
                        020F        T16_S1_X:
00000000'9F   00000000'EF   FA  020F                    CALLG   $S$, a#DS$ENDSUB
```

```
                                021A   246 $DS_BGNSUB
                                021A       T16_S2::
00000000'9F   0000000C'EF   FA  021A                 CALLG   $S$, a#DS$BGNSUB
                                0225   247
                                0225   248 ;
                                0225   249 ; Subtest 2: Cross page silo test
                                0225   250 ;
                                0225   251
      52   00000000'EF   D0     0225   252          MOVL    RH_CUR_ADR,R2          ; STORE RH750 ADDRESS
0000'C2    00000000'8F   C8     022C   253 10$:     BISL    #PGM_INIT,CR(R2)       ; INIT THE MUT
0000'C2    00000000'8F   C8     0235   254          BISL    #MAINT_MODE,CR(R2)     ; SET MAINT MODE
      56   00000000'EF   DE     023E   255          MOVAL   MIOBUFFER,R6           ; STORE BUFFER ADDRESS POINTER
           55   56       D0     0245   256          MOVL    R6,R5                  ; MAKE COPY OF BUFFER ADDRESS
      56   000001FF 8F   CA     0248   257          BICL    # X1FF,R6             ; R6 POINTS TO BYTE 0
           85   00       D2     024F   258          MCOML   #0,(R5)+               ; WRITE ALL ONES TO FIRST 8 BYTES
           65   00       D2     0252   259          MCOML   #0,(R5)                ; OF BUFFER TO SET UP FOR WRITE TO MB
       57  56   00'8F     78     0255   260          ASHL    #-PF_FIELD,R6,R7       ; EXTRACT PFN
      57   00000000'8F   C8     025A   261          BISL    #VALID_BIT,R7          ; SET V-BIT IN MAP ENTRY
5A    52   00000000'8F   C1     0261   262          ADDL3   #MAP_OFFSET,R2,R10     ; R10 POINTS TO MAP REG 0
           6A   57       D0     0269   263          MOVL    R7,(R10)               ; WRITE PFN INTO MAP REG 0
           0000'C2      D4     026C   264          CLRL    VAR(R2)                ; SELECT MAP REG 0
      0000'C2   06       CE     0270   265          MNEGL   #6,BCR(R2)             ; SET BCR FOR 6 BYTE TRANSFER
      50   00000000'EF   D0     0275   266          MOVL    DRIVE0,R0              ; GET ADDRESS OF DRIVE 0
      60   00000000'8F   D0     027C   267          MOVL    #WRITE,(R0)            ; WRITE DATA INTO MB MAINT MODE BUFF.
0000'C2    00000000'8F   C8     0283   268          BISL    #SIM_OCC,DR(R2)        ; SET OCCUPIED
0000'C2    00000000'8F   C8     028C   269          BISL    #SIM_SCLK,DR(R2)       ; ASSERT SCLK
0000'C2    00000000'8F   CA     0295   270          BICL    #SIM_SCLK,DR(R2)       ; DEASSERT SCLK
0000'C2    00000000'8F   C8     029E   271          BISL    #DT_ABORT,CR(R2)       ; FORCE ABORT
0000'C2    00000000'8F   C8     02A7   272          BISL    #SIM_EBL,DR(R2)        ; ASSERT EBL
0000'C2    00000000'8F   CA     02B0   273          BICL    #SIM_EBL,DR(R2)        ; DEASSERT EBL
      52   00000000'EF   D0     02B9   274          MOVL    RH_CUR_ADR,R2          ; STORE RH750 ADDRESS
0000'C2    00000000'8F   C8     02C0   275          BISL    #PGM_INIT,CR(R2)       ; INIT THE MUT
0000'C2    00000000'8F   C8     02C9   276          BISL    #MAINT_MODE,CR(R2)     ; SET MAINT MODE
      56   00000000'EF   DE     02D2   277          MOVAL   DATA_IBUFFER,R6        ; STORE ADDRESS OF BUFFER
       57  56   00'8F     78     02D9   278          ASHL    #-PF_FIELD,R6,R7       ; EXTRACT PFN
      57   00000000'8F   C8     02DE   279          BISL    #VALID_BIT,R7          ; SET VALID BIT FOR FIRST PAGE
5A    52   00000000'8F   C1     02E5   280          ADDL3   #MAP_OFFSET,R2,R10     ; R10 POINTS TO MAP REG 0
           6A   57       D0     02ED   281          MOVL    R7,(R10)               ; WRITE PFN INTO MAP REG 0
0000'C2    000001FC 8F   D0     02F6   282          MOVL    # X1FC,VAR(R2)        ; VAR POINTS TO LAST LONGWORD
                                02F9   283                                         ; ADDRESS IN FIRST PAGE
         58   57   01     C1     02F9   284          ADDL3   #1,R7,R8               ; GENERATE PFN FOR SECOND PAGE
           04 AA   58     D0     02FD   285          MOVL    R8,4(R10)              ; WRITE PFN INTO MAP REG 1
   04 AA   00000000'8F   CA     0301   286          BICL    #VALID_BIT,4(R10)      ; MAKE SECOND PAGE INVALID
           59   56       D0     0309   287          MOVL    R6,R9                  ; GET BUFFER ADDRESS AGAIN
      59   000001FC 8F   C8     030C   288          BISL    # X1FC,R9             ; POINT TO LAST LONG ADDR IN PAGE 0
                89       D4     0313   289          CLRL    (R9)+                  ; CLEAR LAST LONG WORD IN PAGE 0
                69       D4     0315   290          CLRL    (R9)                   ; CLEAR FIRST LONG WORD IN PAGE 1
           00000000'8F   D0     0317   291          MOVL    #MAP_INVALID!DATA_XFER_ABRT!- ; SET UP EXPECTED RESULTS
                54              031D   292                  DATA_XFER_DONE,R4
      0000'C2   08       CE     031E   293          MNEGL   #8,BCR(R2)             ; SET BYTE COUNT FOR 8 BYTES
      50   00000000'EF   D0     0323   294          MOVL    DRIVE0,R0              ; GET ADDRESS OF DRIVE 0
      60   00000000'8F   D0     032A   295          MOVL    #READ,(R0)             ; EXECUTE CMI WRITE
0000'C2    00000000'8F   D0     0331   296          MOVL    #SIM_OCC,DR(R2)        ; ASSERT OCCUPIED
                55       D4     033A   297          CLRL    R5                     ; CLEAR SCLK COUNTER
0000'C2    00000000'8F   C8     033C   298 20$:     BISL    #SIM_SCLK,DR(R2)       ; ASSERT SCLK
0000'C2    00000000'8F   CA     0345   299          BICL    #SIM_SCLK,DR(R2)       ; DEASSERT SCLK
        EA 55   04       F3     034E   300          AOBLEQ  #4,R5,20$              ; CLOCK 8 BYTES
```

```
0000'C2   00000000'8F   C8  0352   301          BISL    #SIM_EBL,DR(R2)          ; ASSERT EBL
0000'C2   00000000'8F   CA  035B   302          BICL    #SIM_EBL,DR(R2)          ; DEASSERT EBL
      54   00000000'EF   C8  0364   303          BISL    RH11TB_MSK,R4            ; LOAD EXPECTED MASK BIT
           53   0000'C2   D0  036B   304          MOVL    SR(R2),R3                ; READ SR
                54   53   D1  0370   305          CMPL    R3,R4                    ; CHECK FOR INVALID MAP
                     1B   13  0373   306          BEQL    30$                      ; BRANCH IF INV MAP IS SET
                                0375   307          $DS_ERRHARD_S   #1,LUN,-         ; REPORT ERROR
                                0375   308                          ARRAY_MSG4,-
                                0375   309                          PRINT_SBE
           00000000'EF   DF  0375                          PUSHAL  PRINT_SBE
           00000000'EF   DF  037B                          PUSHAL  ARRAY_MSG4
           00000000'EF   DD  0381                          PUSHL   LUN
                     01   DD  0387                          PUSHL   #1
      00000000'9F   04   FB  0389                          CALLS   #$$M, a#DS$ERRHARD
                                0390   310 30$:    $DS_CKLOOP      10$              ; SCOPE LOOP?
00000000'9F   FE98 CF   FA  0390                          CALLG   10$, a#DS$CKLOOP
                55   69   D0  0399   311          MOVL    (R9),R5                  ; READ FIRST LONGWORD IN PAGE 1
                55   00   D1  039C   312          CMPL    #0,R5                    ; CHECK FOR CONTENTS OF ZERO
                     1F   13  039F   313          BEQL    40$                      ; BRANCH IF DATA CORRECT
                                03A1   314          $DS_ERRHARD_S   #2,LUN,-         ; REPORT ERROR
                                03A1   315                          ARRAY_MSG4,-
                                03A1   316                          PRINT_SILOER,-
                                03A1   317                          #0,R5
                     55   DD  03A1                          PUSHL   R5
                     00   DD  03A3                          PUSHL   #0
           00000000'EF   DF  03A5                          PUSHAL  PRINT_SILOER
           00000000'EF   DF  03AB                          PUSHAL  ARRAY_MSG4
           00000000'EF   DD  03B1                          PUSHL   LUN
                     02   DD  03B7                          PUSHL   #2
      00000000'9F   06   FB  03B9                          CALLS   #$$M, a#DS$ERRHARD
                                03C0   318 40$:    $DS_CKLOOP      10$              ; SCOPE LOOP?
00000000'9F   FE68 CF   FA  03C0                          CALLG   10$, a#DS$CKLOOP
                5B   79   D0  03C9   319          MOVL    -(R9),R11                ; READ LAST LONGWORD PAGE 0
      5B   FFFFFFFF 8F   D1  03CC   320          CMPL    #-1,R11                  ; CHECK FOR CONTENTS OF ALL ONES
                     23   13  03D3   321          BEQL    50$                      ; BRANCH IF DATA CORRECT
                                03D5   322          $DS_ERRHARD_S   #3,LUN,-         ; REPORT ERROR
                                03D5   323                          ARRAY_MSG4,-
                                03D5   324                          PRINT_SILOER,-
                                03D5   325                          #-1,R11
                     5B   DD  03D5                          PUSHL   R11
           FFFFFFFF 8F   DD  03D7                          PUSHL   #-1
           00000000'EF   DF  03DD                          PUSHAL  PRINT_SILOER
           00000000'EF   DF  03E3                          PUSHAL  ARRAY_MSG4
           00000000'EF   DD  03E9                          PUSHL   LUN
                     03   DD  03EF                          PUSHL   #3
      00000000'9F   06   FB  03F1                          CALLS   #$$M, a#DS$ERRHARD
                                03F8   326 50$:    $DS_CKLOOP      10$              ; SCOPE LOOP?
00000000'9F   FE30 CF   FA  03F8                          CALLG   10$, a#DS$CKLOOP
                                0401   327 $DS_ENDSUB
                                0401       T16_S2_X:
00000000'9F   0000000C'EF   FA  0401                          CALLG   $S$, a#DS$ENDSUB
                                040C   328 $DS_ENDTEST
                50   01   D0  040C                          MOVL    #1, R0           ; NORMAL EXIT
                                040F       TEST_016_X::
      00000000'9F   6E   FA  040F                          CALLG   (SP), a#DS$BREAK
                           04  0416                          RET                      ; RETURN TO TEST SEQUENCER
                                0417   329 $DS_PAGE
```

```
                                  0417                .SBTTL  TEST 17:  INTERRUPT TESTS
                             00000000                         .PSECT  TEST_017, PAGE, NOWRT
                                  0014
                                  0014   332 $DS_BGNTEST      <DEFAULT,ALL>,,LONG
                                  0014       DATA_017:
                        00000000  0014                         .LONG   0                  ; TEST ARGUMENT TABLE TERMINATOR
                                  0018       TEST_017::
                            0000  0018                         .WORD    M<>               ; ENTRY MASK
                                  001A   333 ;++
                                  001A   334 ; TEST DESCRIPTION:
                                  001A   335 ;
                                  001A   336 ; THIS TEST CHECKS FOR THE CORRECT RH750 INTERRUPT VECTOR ADDRESS
                                  001A   337 ; AND VERIFIES THAT INTERRUPTS OCCUR FOR EVERY STATUS CONDITION
                                  001A   338 ; CAPABLE OF GENERATING AN INTERRUPT.
                                  001A   339 ;
                                  001A   340 ; TEST ALGORITHM:
                                  001A   341 ;
                                  001A   342 ; SET A FLAG INDICATING THAT INTERRUPTS ARE TO OCCUR,
                                  001A   343 ; CLEAR THE INT_OCCURRED FLAG, SET UP A CONDITION WHICH WILL
                                  001A   344 ; CAUSE THE INTERRUPT, CHECK THE INTERRUPT EVENT FLAG AND REPORT ERRORS
                                  001A   345 ; IF THE INTERRUPT FAILED TO OCCUR OR IF THE STATUS IS NOT AS EXPECTED.
                                  001A   346 ;--
                        FFE3'  30 001A   347           BSBW    RH11TB_PRES                ; CHECK FOR RH11TB
                                  001D   348           ERRPREP RHSR_MSG,2,FMT_STATUS_REG        ; PREPARE TO HANDLE ERROR
  00000000'EF   00000000'EF   DE  001D                         MOVAL   RHSR_MSG,REG_NAME
           00000000'EF   02   9A  0028                         MOVZBL  #2,REG_NO
  00000000'EF   00000000'EF   DE  002F                         MOVAL   FMT_STATUS_REG,REG_STRING
           52   00000000'EF   D0  003A   349           MOVL    RH_CUR_ADR,R2              ; MOVE RH750 ADDRESS INTO R2
           50   00000000'EF   DE  0041   350           MOVAL   MIOBUFFER,R0               ; GET BUFFER ADDRESS
            50   50   F7 8F   78  0048   351           ASHL    #-9,R0,R0                  ; NORMALIZE FOR MAP ENTRY
           50   00000000'8F   C8  004D   352           BISL    #VALID_BIT,R0              ; VALIDATE ENTRY
      51   52   00000000'8F   C1  0054   353           ADDL3   #MAP_OFFSET,R2,R1          ; R1 ---> MAP 0
                     61   50   D0  005C   354           MOVL    R0,(R1)                    ; WRITE MAP ZERO
                                  005F   355 ;++
                                  005F   356 ; CORRECT VECTOR AND DATA TRANSFER ABORTED INTERRUPT TEST
                                  005F   357 ;--
                                  005F   358           $DS_BGNSUB
                                  005F       T17_S1::
  00000000'9F   00000018'EF   FA  005F                         CALLG   $S$, a#DS$BGNSUB
                          55   D4  006A   359           CLRL    R5                         ; CLEAR VECTOR ADDRESS COUNTER
           58   00000000'EF   DE  006C   360           MOVAL   RH_VECS,R8                 ; GET VECTOR ADDRESS
                                  0073   361
           56   00000000'EF   DE  0073   362           MOVAL   ISR_TBL,R6                 ; GET ADDRESS OF INTR SERVICE ROUTINE TABLE
                                  007A   363           $DS_CLRVEC_S    RH_VECTOR          ; RELEASE THE CURRENT INTERRUPT VECTOR
                00000000'EF   DD  007A                         PUSHL   RH_VECTOR
           00000000'9F   01   FB  0080                         CALLS   #1, a#DS$CLRVEC
                                  0087   364 10$:      $DS_SETVEC_S    (R8),a(R6)         ; SET UP TO CAPTURE ALL RH VECTORS
                          00   DD  0087                         PUSHL   #0
                       00 B6   DF  0089                         PUSHAL  a(R6)
                          68   DD  008C                         PUSHL   (R8)
           00000000'9F   03   FB  008E                         CALLS   #3, a#DS$SETVEC
                     86   88   D1  0095   365           CMPL    (R8)+,(R6)+                ; POINT TO NEXT VECTOR AND SERVICE RTN
                  EB 55   0F   F3  0098   366           AOBLEQ  #15,R5,10$                 ; SET UP TO CAPTURE 16 VECTORS
                                  009C   367
                                  009C   368 ;
                                  009C   369 ; Start the Data Transfer Aborted test
                                  009C   370 ;
```

```
                                  009C   371
     0000'C2   00000000'8F   C8   009C   372 20$:   BISL    #PGM_INIT,CR(R2)          ; INIT RH750
     0000'C2   00000000'8F   C8   00A5   373         BISL    #MAINT_MODE,CR(R2)        ; PLACE RH750 IN MAINTENANCE MODE
          56   00000000'EF   D0   00AE   374         MOVL    DRIVE0,R6                 ; MOVE ADDRESS OF DRIVE 0 TO R6
          66   00000000'8F   D0   00B5   375         MOVL    #READ,(R6)                ; ISSUE READ COMMAND
     0000'C2   00000000'8F   C8   00BC   376         BISL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
          54   00000000'8F   D0   00C5   377         MOVL    #DT_BUSY,R4               ; SET R4' S BIT31
          54   00000000'EF   C8   00CC   378         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0   00D3   379         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                   54   53   D1   00D8   380         CMPL    R3,R4                     ; IS DT BUSY IN STATUS REGISTER?
                        1B   13   00DB   381         BEQL    30$                       ; BRANCH IF SO
                                  00DD   382         $DS_ERRHARD_S   #1,LUN,-          ; PRINT ERROR
                                  00DD   383                         ARRAY_MSG62,-
                                  00DD   384                         PRINT_SBE
              00000000'EF   DF   00DD                         PUSHAL  PRINT_SBE
              00000000'EF   DF   00E3                         PUSHAL  ARRAY_MSG62
              00000000'EF   DD   00E9                         PUSHL   LUN
                        01   DD   00EF                         PUSHL   #1
          00000000'9F   04   FB   00F1                         CALLS   #$$M, a#DS$ERRHARD
                                  00F8   385 30$:   $DS_CKLOOP      20$               ; SCOPE LOOP?
       00000000'9F   A1 AF   FA   00F8                         CALLG   20$, a#DS$CKLOOP
                                  0100   386         EXPECT_INTER                      ; PREPARE FOR INTERRUPT
          00000000'8F   00   DA   0100                         MTPR    #0,#IPLR
          00000000'EF   01   90   0107                         MOVB    #1,EXP_INTERRUPT
              00000000'EF   94   010E                         CLRB    INT_OCCURRED
     0000'C2   00000000'8F   C8   0114                         BISL    #INTERRUPT_EN,CR(R2)
     0000'C2   00000000'8F   C8   011D   387         BISL    #DT_ABORT,CR(R2)          ; SET UP TO CAUSE ABORT DATA XFER
     0000'C2   00000000'8F   C8   0126   388         BISL    #SIM_EBL,DR(R2)           ; ASSERT EBL
     0000'C2   00000000'8F   CA   012F   389         BICL    #SIM_EBL,DR(R2)           ; DEASSEERT EBL
          54   00000000'8F   D0   0138   390         MOVL    #DATA_XFER_ABRT!DATA_XFER_DONE,R4 ; EXPECTED VALUE
          54   00000000'EF   C8   013F   391         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0   0146   392         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
       1B 00000000'EF   00   E4   014B   393         BBSC    #0,INT_OCCURRED,35$       ; IF INTERRUPT OCCURRED SKIP REPORT
                                  0153   394         $DS_ERRHARD_S   #2,LUN,-
                                  0153   395                         ARRAY_MSG1,-
                                  0153   396                         PRINT_INTERR      ; PRINT ERROR
              00000000'EF   DF   0153                         PUSHAL  PRINT_INTERR
              00000000'EF   DF   0159                         PUSHAL  ARRAY_MSG1
              00000000'EF   DD   015F                         PUSHL   LUN
                        02   DD   0165                         PUSHL   #2
          00000000'9F   04   FB   0167                         CALLS   #$$M, a#DS$ERRHARD
                   54   53   D1   016E   397 35$:   CMPL    R3,R4                     ; IS STATUS CORRECT?
                        1B   13   0171   398         BEQL    40$                       ; BRANCH IF SO
                                  0173   399         $DS_ERRHARD_S   #3,LUN,-          ; PRINT ERROR
                                  0173   400                         ARRAY_MSG61,-
                                  0173   401                         PRINT_SBE
              00000000'EF   DF   0173                         PUSHAL  PRINT_SBE
              00000000'EF   DF   0179                         PUSHAL  ARRAY_MSG61
              00000000'EF   DD   017F                         PUSHL   LUN
                        03   DD   0185                         PUSHL   #3
          00000000'9F   04   FB   0187                         CALLS   #$$M, a#DS$ERRHARD
                                  018E   402 40$:   $DS_CKLOOP      20$               ; SCOPE LOOP?
       00000000'9F   FF0A CF   FA   018E                         CALLG   20$, a#DS$CKLOOP
   00000000'FF   00000000'8F   D0   0197   403 50$:   MOVL    #READ,aDRIVE0             ; ISSUE READ TO DRIVE
          54   00000000'8F   D0   01A2   404         MOVL    #DT_BUSY,R4               ; SET UP EXPECTED VALUE
          54   00000000'EF   C8   01A9   405         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0   01B0   406         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
```

```
                  54   53   D1  01B5   407          CMPL    R3,R4                    ; ARE STATUS BITS CLEARED
                       1B   13  01B8   408          BEQL    60$                      ; IF NO BITS ARE SET THEN PROCEED
                                01BA   409          $DS_ERRHARD_S   #4,LUN,-         ; PRINT ERROR
                                01BA   410                          ARRAY_MSG62,-
                                01BA   411                          PRINT_SBE
            00000000'EF    DF  01BA                        PUSHAL  PRINT_SBE
            00000000'EF    DF  01C0                        PUSHAL  ARRAY_MSG62
            00000000'EF    DD  01C6                        PUSHL   LUN
                       04   DD  01CC                        PUSHL   #4
        00000000'9F    04   FB  01CE                        CALLS   #$$M, a#DS$ERRHARD
                                01D5   412 60$:     $DS_CKLOOP      50$              ; SCOPE LOOP?
     00000000'9F   BF AF    FA  01D5                        CALLG   50$, a#DS$CKLOOP
00000000'EF   00000000'EF   D1  01DD   413          CMPL    TEST_VEC,RH_VECTOR       ; CHECK FOR THE CORRECT VECTOR
                       27   13  01E8   414          BEQL    70$                      ; BRANCH IF OK
                                01EA   415          $DS_ERRHARD_S   #5,LUN,-         ; REPORT ERROR
                                01EA   416                          ARRAY_MSG30,-
                                01EA   417                          PRINT_VECTERR,-
                                01EA   418                          TEST_VEC,RH_VECTOR
            00000000'EF    DD  01EA                        PUSHL   RH_VECTOR
            00000000'EF    DD  01F0                        PUSHL   TEST_VEC
            00000000'EF    DF  01F6                        PUSHAL  PRINT_VECTERR
            00000000'EF    DF  01FC                        PUSHAL  ARRAY_MSG30
            00000000'EF    DD  0202                        PUSHL   LUN
                       05   DD  0208                        PUSHL   #5
        00000000'9F    06   FB  020A                        CALLS   #$$M, a#DS$ERRHARD
                                0211   419 70$:     $DS_CKLOOP      10$              ; SCOPE LOOP?
    00000000'9F   FE72 CF   FA  0211                        CALLG   10$, a#DS$CKLOOP
                       55   D4  021A   420          CLRL    R5                       ; CLEAR VECTOR ADDRESS COUNTER
         58   00000000'EF   DE  021C   421          MOVAL   RH_VECS,R8               ; LOAD VECTOR ADDRESS
                                0223   422 80$:     $DS_CLRVEC_S    (R8)             ; RELEASE ALL RH VECTORS
                       68   DD  0223                        PUSHL   (R8)
        00000000'9F    01   FB  0225                        CALLS   #1, a#DS$CLRVEC
                       88   D5  022C   423          TSTL    (R8)+                    ; POINT TO NEXT VECTOR IN TABLE
              F1 55    0F   F3  022E   424          AOBLEQ  #15,R5,80$               ; RELEASE 16 VECTORS
                                0232   425
                                0232   426          $DS_ENDSUB
                                0232       T17_S1_X:
00000000'9F   00000018'EF   FA  0232                        CALLG   $S$, a#DS$ENDSUB
```

```
                                023D   428 ;**
                                023D   429 ; ATTENTION INTERRUPT TEST
                                023D   430 ;--
                                023D   431          $DS_BGNSUB
                                023D       T17_S2::
00000000'9F   00000024'EF   FA  023D                CALLG   $S$, a#DS$BGNSUB
                                0248   432          $DS_SETVEC_S    RH_VECTOR,RH_ISR; USE NORMAL INTERRUPT SERVICE ROUTINE
                      00    DD  0248                PUSHL   #0
              00000000'EF   DF  024A                PUSHAL  RH_ISR
              00000000'EF   DD  0250                PUSHL   RH_VECTOR
         00000000'9F   03   FB  0256                CALLS   #3, a#DS$SETVEC
    0000'C2   00000000'8F   C8  025D   433 60$:     BISL    #PGM_INIT,CR(R2)        ; INIT
    0000'C2   00000000'8F   C8  0266   434          BISL    #MAINT_MODE,CR(R2)      ; PUT IN MAINTENANCE MODE
                            026F   435          EXPECT_INTER                    ; ARM INTERRUPT FLAG
         00000000'8F   00   DA  026F                MTPR    #0,#IPLR
         00000000'EF   01   90  0276                MOVB    #1,EXP_INTERRUPT
              00000000'EF   94  027D                CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8  0283                BISL    #INTERRUPT_EN,CR(R2)
         54   00000000'8F   D0  028C   436          MOVL    #ATTENTION,R4           ; EXPECT ATTENTION
    0000'C2   00000000'8F   C8  0293   437          BISL    #SIM_ATTN,DR(R2)        ; CAUSE ATTN TO BE SET IN STATUS REG
              53   0000'C2   D0  029C   438          MOVL    SR(R2),R3               ; READ STATUS REGISTER
    0000'C2   00000000'8F   CA  02A1   439          BICL    #SIM_ATTN,DR(R2)        ; DEASSERT SIM_ATTN
      1B 00000000'EF   00   E4  02AA   440          BBSC    #0,INT_OCCURRED,65$     ; IF INTERRUPT OCCURRED PROCEED
                            02B2   441          $DS_ERRHARD_S   #1,LUN,-
                            02B2   442                          ARRAY_MSG3,-
                            02B2   443                          PRINT_INTERR    ; REPORT INTERRUPT FAILURE
              00000000'EF   DF  02B2                PUSHAL  PRINT_INTERR
              00000000'EF   DF  02B8                PUSHAL  ARRAY_MSG3
              00000000'EF   DD  02BE                PUSHL   LUN
                      01    DD  02C4                PUSHL   #1
         00000000'9F   04   FB  02C6                CALLS   #$$M, a#DS$ERRHARD
              54   53   D1  02CD   444 65$:     CMPL    R3,R4                   ; IS ATTN SET?
                   1B   13  02D0   445          BEQL    70$                     ; BRANCH IF SO
                            02D2   446          $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                            02D2   447                          ARRAY_MSG3,-
                            02D2   448                          PRINT_SBE
              00000000'EF   DF  02D2                PUSHAL  PRINT_SBE
              00000000'EF   DF  02D8                PUSHAL  ARRAY_MSG3
              00000000'EF   DD  02DE                PUSHL   LUN
                      02    DD  02E4                PUSHL   #2
         00000000'9F   04   FB  02E6                CALLS   #$$M, a#DS$ERRHARD
                            02ED   449 70$:     $DS_CKLOOP      60$             ; SCOPE LOOP?
00000000'9F   FF6C CF       FA  02ED                CALLG   60$, a#DS$CKLOOP
         0000'C2   53   D0  02F6   450 80$:     MOVL    R3,SR(R2)               ; CLEAR STATUS BITS
         53   0000'C2   D0  02FB   451          MOVL    SR(R2),R3               ; READ STATUS REGISTER
                   54   D4  0300   452          CLRL    R4                      ; CLEAR EXP RESULTS REG
    54   00000000'EF   C8  0302   453          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              54   53   D1  0309   454          CMPL    R3,R4                   ; CHECK RESULTS
                   1B   13  030C   455          BEQL    90$                     ; BRANCH IF NO ERRORS IN STATUS REG
                            030E   456          $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                            030E   457                          ARRAY_MSG3,-
                            030E   458                          PRINT_SBE
              00000000'EF   DF  030E                PUSHAL  PRINT_SBE
              00000000'EF   DF  0314                PUSHAL  ARRAY_MSG3
              00000000'EF   DD  031A                PUSHL   LUN
                      03    DD  0320                PUSHL   #3
         00000000'9F   04   FB  0322                CALLS   #$$M, a#DS$ERRHARD
```

```
                          0329   459 90$:    $DS_CKLOOP      80$              ; SCOPE LOOP?
        00000000'9F   CA AF   FA  0329                        CALLG   80$, a#DS$CKLOOP
                          0331   460          $DS_ENDSUB
                          0331        T17_S2_X:
    00000000'9F   00000024'EF   FA  0331                        CALLG   $S$, a#DS$ENDSUB
```

```
                                033C   462 ;**
                                033C   463 ; NON EXISTENT DRIVE INTERRUPT TEST
                                033C   464 ;--
                                033C   465         $DS_BGNSUB
                                033C       T17_S3::
00000000'9F   00000030'EF   FA  033C                CALLG   $S$, a#DS$BGNSUB
                                0347   466 ;**
                                0347   467 ; NOTE:
                                0347   468 ;
                                0347   469 ; IF 8 DRIVES ARE ATTACHED TO THE RH750 THEN THE INTERRUPT
                                0347   470 ; ASSOCIATED WITH "NED" CAN NOT BE TESTED.
                                0347   471 ;--
      57   00000000'EF      D0  0347   472         MOVL    DRIVE0,R7               ; PUT DRIVE0'S ADDRESS INTO R7
                     56     D4  034E   473         CLRL    R6                      ; CLEAR DRIVE COUNTER
 0000'C2   00000000'8F      C8  0350   474 110$:   BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                     50     D4  0359   475         CLRL    R0                      ; CLEAR DELAY COUNTER
                     67     D5  035B   476         TSTL    (R7)                    ; READ A DRIVE REGISTER
           53   0000'C2     D0  035D   477         MOVL    SR(R2),R3               ; READ STATUS
        02 53     12     E1  0362   478         BBC     #18,R3,120$             ; IF NED IS CLEAR THEN BRANCH HAVING
                                0366   479                                         ; DETECTED A NON EXISTENT DRIVE A TEST
                                0366   480                                         ; IS MADE FOR THE ASSOCIATED INTERRUPT
                     03     11  0366   481         BRB     125$                    ; SINCE NED IS SET, GO DO TEST
                   0051     31  0368   482 120$:   BRW     130$                    ; NED CLEAR, SO BRANCH
                                036B   483 125$:   EXPECT_INTER                   ; ENABLE INTERRUPTS
      00000000'8F     00    DA  036B                MTPR    #0,#IPLR
      00000000'EF     01    90  0372                MOVB    #1,EXP_INTERRUPT
           00000000'EF      94  0379                CLRB    INT_OCCURRED
 0000'C2   00000000'8F      C8  037F                BISL    #INTERRUPT_EN,CR(R2)
          FC 50     0A     F3  0388   484 126$:   AOBLEQ  #10,R0,126$             ; WAIT FOR INTERRUPT TO OCCUR
   42 00000000'EF     00    E4  038C   485         BBSC    #0,INT_OCCURRED,135$    ; IF INTERRUPT OCCURRED NO ERROR
             54     53     D0  0394   486         MOVL    R3,R4
                                0397   487         $DS_ERRHARD_S   #1,LUN,-
                                0397   488                         ARRAY_MSG58,-
                                0397   489                         PRINT_INTERR    ; REPORT ERROR
           00000000'EF      DF  0397                PUSHAL  PRINT_INTERR
           00000000'EF      DF  039D                PUSHAL  ARRAY_MSG58
           00000000'EF      DD  03A3                PUSHL   LUN
                     01     DD  03A9                PUSHL   #1
      00000000'9F    04     FB  03AB                CALLS   #$$M, a#DS$ERRHARD
                                03B2   490         $DS_CKLOOP      110$            ; SCOPE LOOP
00000000'9F   9B AF         FA  03B2                CALLG   110$, a#DS$CKLOOP
                     1A     11  03BA   491         BRB     135$                    ; EXIT IF NO LOOP ON ERROR
   57   00000000'8F         C0  03BC   492 130$:   ADDL    #DRIVE_OFFSET,R7        ; INCREMENT DRIVE ADDRESS POINTER
  FF87 56     01     07     9D  03C3   493         ACBB    #7,#1,R6,110$           ; LOOP IF ALL DRIVES NOT ACCESSED
                                03C9   494         $DS_PRINTX_S    FMT_NOINTEST    ; REPORT NED INTERRUPT NOT TESTED
           00000000'EF      9F  03C9                PUSHAB  FMT_NOINTEST
      00000000'9F    01     FB  03CF                CALLS   #$$N, a#DS$PRINTX
                                03D6   495 135$:
                                03D6   496         $DS_ENDSUB
                                03D6       T17_S3_X:
00000000'9F   00000030'EF   FA  03D6                CALLG   $S$, a#DS$ENDSUB
```

```
                                  03E1   498 ;++
                                  03E1   499 ; MISSED TRANSFER INTERRUPT TEST
                                  03E1   500 ;--
                                  03E1   501         $DS_BGNSUB
                                  03E1       T17_S4::
00000000'9F   0000003C'EF   FA    03E1                   CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8    03EC   502 140$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8    03F5   503         BISL    #MAINT_MODE,CR(R2)        ; PUT INTO MAINTENANCE MODE
         56   00000000'EF   D0    03FE   504         MOVL    DRIVE0,R6                 ; PUT ADDRESS OF DRIVE IN R6
         54   00000000'8F   D0    0405   505         MOVL    #DATA_XFER_DONE!DATA_XFER_ABRT!MISSED_XFER,R4   ; EXPECTED
                                  040C   506         EXPECT_INTER                      ; ARM INTERRUPTS
         00000000'8F   00   DA    040C                   MTPR    #0,#IPLR
         00000000'EF   01   90    0413                   MOVB    #1,EXP_INTERRUPT
              00000000'EF   94    041A                   CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8    0420                   BISL    #INTERRUPT_EN,CR(R2)
         66   00000000'8F   D0    0429   507         MOVL    #READ,(R6)                ; ISSUE READ COMMAND TO DRIVE
                                  0430   508         $DS_WAITUS_S    #200              ; WAIT 2 MILLISECONDS
                        00   DD    0430                   PUSHL   #0
              000000C8 8F   DD    0432                   PUSHL   #200
         00000000'9F   02   FB    0438                   CALLS   #2, a#DS$WAITUS
    0000'C2   00000000'8F   C8    043F   509         BISL    #SIM_EBL,DR(R2)           ; SET SIMULATE END OF BLOCK
    0000'C2   00000000'8F   CA    0448   510         BICL    #SIM_EBL,DR(R2)           ; CLEAR SIMULATE EBL
         54   00000000'EF   C8    0451   511         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0    0458   512         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
      1B 00000000'EF   00   E4    045D   513         BBSC    #0,INT_OCCURRED,145$      ; BRANCH IF INTERRUPT OCCURRED
                                  0465   514         $DS_ERRHARD_S   #1,LUN,-
                                  0465   515                         ARRAY_MSG57,-
                                  0465   516                         PRINT_INTERR      ; REPORT ERROR
              00000000'EF   DF    0465                   PUSHAL  PRINT_INTERR
              00000000'EF   DF    046B                   PUSHAL  ARRAY_MSG57
              00000000'EF   DD    0471                   PUSHL   LUN
                        01   DD    0477                   PUSHL   #1
         00000000'9F   04   FB    0479                   CALLS   #$$M, a#DS$ERRHARD
                   54   53   D1    0480   517 145$:   CMPL    R3,R4                     ; IS MISSED XFER BIT SET?
                        1B   13    0483   518         BEQL    150$                      ; BRANCH IF MISSED TRANSFER
                                  0485   519         $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                  0485   520                         ARRAY_MSG4,-
                                  0485   521                         PRINT_SBE
              00000000'EF   DF    0485                   PUSHAL  PRINT_SBE
              00000000'EF   DF    048B                   PUSHAL  ARRAY_MSG4
              00000000'EF   DD    0491                   PUSHL   LUN
                        02   DD    0497                   PUSHL   #2
         00000000'9F   04   FB    0499                   CALLS   #$$M, a#DS$ERRHARD
                                  04A0   522 150$:   $DS_CKLOOP      140$              ; SCOPE LOOP?
    00000000'9F   FF48 CF   FA    04A0                   CALLG   140$, a#DS$CKLOOP
00000000'FF   00000000'8F   D0    04A9   523 160$:   MOVL    #READ,aDRIVE0             ; ISSUE READ COMMAND
         54   00000000'8F   D0    04B4   524         MOVL    #DT_BUSY,R4               ; SET UP EXPECTED VALUE
         54   00000000'EF   C8    04BB   525         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0    04C2   526         MOVL    SR(R2),R3                 ; CHECK FOR SA1 BITS IN STATUS REG
                   54   53   D1    04C7   527         CMPL    R3,R4                     ; COMPARE ACTUAL AND EXPECTED STATUS
                        1B   13    04CA   528         BEQL    170$                      ; BRANCH IF STATE IS OK
                                  04CC   529         $DS_ERRHARD_S   #3,LUN,-          ; PRINT ERROR
                                  04CC   530                         ARRAY_MSG60,-
                                  04CC   531                         PRINT_SBE
              00000000'EF   DF    04CC                   PUSHAL  PRINT_SBE
              00000000'EF   DF    04D2                   PUSHAL  ARRAY_MSG60
              00000000'EF   DD    04D8                   PUSHL   LUN
```

```
                        03   DD  04DE                          PUSHL   #3
          00000000'9F   04   FB  04E0                          CALLS   #$$M, a#DS$ERRHARD
                                 04E7   532 170$:   $DS_CKLOOP      140$            ; SCOPE LOOP?
     00000000'9F   FF01 CF   FA  04E7                          CALLG   140$, a#DS$CKLOOP
                                 04F0   533         $DS_ENDSUB
                                 04F0       T17_S4_X:
  00000000'9F   0000003C'EF   FA  04F0                          CALLG   $S$, a#DS$ENDSUB
```

```
                                04FB    535 ;**
                                04FB    536 ; MASS BUS EXCEPTION INTERRUPT TEST
                                04FB    537 ;--
                                04FB    538         $DS_BGNSUB
                                04FB            T17_S5::
00000000'9F   00000048'EF   FA  04FB                    CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0506    539 180$:   BISL    #PGM_INIT,CR(R2)        ; INIT RH
    0000'C2   00000000'8F   C8  050F    540         BISL    #MAINT_MODE,CR(R2)      ; PUT IT INTO MAINTENANCE MODE
    50   52   00000000'8F   C1  0518    541         ADDL3   #MAP_OFFSET,R2,R0       ; R0 NOW POINTS TO MAP REG 0
         60   00000000'8F   D0  0520    542         MOVL    #VALID_BIT,(R0)         ; VALIDATE MAP REG 0
                                0527    543                                         ; THIS IS REQUIRED TO PREVENT STATUS
                                0527    544                                         ; INDICATING INVALID MAP WHEN THE
                                0527    545                                         ; PHYSICAL ADDRESS IS ASSEMBLED
    0000'C2   FFFFFFD7 8F   D0  0527    546         MOVL    #-41,BCR(R2)            ; WRITE BCR
                   0000'C2  D4  0530    547         CLRL    VAR(R2)                 ; CLEAR VIRTUAL ADDRESS REGISTER
                                0534    548         EXPECT_INTER                    ; ARM INTERRUPTS
         00000000'8F   00   DA  0534                    MTPR    #0,#IPLR
         00000000'EF   01   90  053B                    MOVB    #1,EXP_INTERRUPT
                00000000'EF   94  0542                  CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8  0548                    BISL    #INTERRUPT_EN,CR(R2)
         54   00000000'8F   D0  0551    549         MOVL    #DATA_XFER_LATE,R4      ; SET UP EXPECTED STATUS
         54   00000000'8F   C8  0558    550         BISL    #DATA_XFER_ABRT!DATA_XFER_DONE,R4 ; EXPECT THESE BITS
         66   00000000'8F   D0  055F    551         MOVL    #WRITE,(R6)             ; ISSUE WRITE COMMAND TO PSEUDO DRIVE
    0000'C2   00000000'8F   D0  0566    552         MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                056F    553         $DS_WAITUS_S    #10             ; WAIT FOR SILO TO FILL
                        00  DD  056F                    PUSHL   #0
                        0A  DD  0571                    PUSHL   #10
    00000000'9F         02  FB  0573                    CALLS   #2, a#DS$WAITUS
0000'C2   00000000'8F   C8  057A    554         BISL    #BLKSND_COMD,DR(R2)     ; SET BLOCK SENDING COMMAND
                                0583    555                                     ; INHIBITS SENDING READ CMD TO MEMORY
                        50  D4  0583    556         CLRL    R0                  ; CLEAR SCLK COUNTER
0000'C2   00000000'8F   C8  0585    557 2000$:  BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
0000'C2   00000000'8F   CA  058E    558         BICL    #SIM_SCLK,DR(R2)        ; NEGATE SCLK
          EA 50         15  F3  0597    559         AOBLEQ  #21,R0,2000$            ; 21 SCLK'S TO CAUSE DATA LATE
                                059B    560                                         ; BECAUSE  SBI COMMANDS ARE BLOCKED
0000'C2   00000000'8F   C8  059B    561         BISL    #SIM_EBL,DR(R2)         ; END OF BLOCK
0000'C2   00000000'8F   CA  05A4    562         BICL    #SIM_EBL,DR(R2)         ; NEGATE END OF BLOCK
                                05AD    563         $DS_WAITUS_S    #10             ; WAIT FOR DATA TO GET TO MEMORY
                        00  DD  05AD                    PUSHL   #0
                        0A  DD  05AF                    PUSHL   #10
    00000000'9F         02  FB  05B1                    CALLS   #2, a#DS$WAITUS
    54   00000000'EF    C8  05B8    564         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
         53   0000'C2   D0  05BF    565         MOVL    SR(R2),R3               ; READ STATUS REGISTER
1B 00000000'EF   00     E4  05C4    566         BBSC    #0,INT_OCCURRED,185$    ; BRANCH IF INTERRUPT OCCURRED
                                05CC    567         $DS_ERRHARD_S   #1,LUN,-
                                05CC    568                         ARRAY_MSG59,-
                                05CC    569                         PRINT_INTERR    ; REPORT INTERRUPT FAILURE
         00000000'EF    DF  05CC                    PUSHAL  PRINT_INTERR
         00000000'EF    DF  05D2                    PUSHAL  ARRAY_MSG59
         00000000'EF    DD  05D8                    PUSHL   LUN
                        01  DD  05DE                PUSHL   #1
    00000000'9F         04  FB  05E0                CALLS   #$$M, a#DS$ERRHARD
              54        53  D1  05E7    570 185$:   CMPL    R3,R4                   ; IS DATA LATE SET?
                        1B  13  05EA    571         BEQL    190$                    ; BRANCH IF SO
                                05EC    572         $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                05EC    573                         ARRAY_MSG6,-
                                05EC    574                         PRINT_SBE
```

```
                00000000'EF   DF  05EC                          PUSHAL  PRINT_SBE
                00000000'EF   DF  05F2                          PUSHAL  ARRAY_MSG6
                00000000'EF   DD  05F8                          PUSHL   LUN
                          02  DD  05FE                          PUSHL   #2
            00000000'9F   04  FB  0600                          CALLS   #$$M, a#DS$ERRHARD
                              0607  575 190$:   $DS_CKLOOP      180$                    ; SCOPE LOOP?
      00000000'9F   FEFB CF   FA  0607                          CALLG   180$, a#DS$CKLOOP
00000000'FF   00000000'8F   D0  0610  576 200$:   MOVL    #READ,@#DRIVE0              ; ISSUE READ FROM DRIVE
              53    0000'C2   D0  061B  577         MOVL    SR(R2),R3                   ; READ STATUS BIS
          54  00000000'8F   D0  0620  578         MOVL    #DT_BUSY,R4                 ; SET UP EXPECTED STATUS
          54  00000000'EF   C8  0627  579         BISL    RH11TB_MSK,R4               ; LOAD EXPECTED MASK BIT
                  54    53  D1  062E  580         CMPL    R3,R4                       ; DID STATUS BITS CLEAR
                        1B  13  0631  581         BEQL    210$                        ; BRANCH IF STATUS IS OK
                            0633  582         $DS_ERRHARD_S   #3,LUN,-             ; PRINT ERROR
                            0633  583                         ARRAY_MSG6,-
                            0633  584                         PRINT_SBE
                00000000'EF   DF  0633                          PUSHAL  PRINT_SBE
                00000000'EF   DF  0639                          PUSHAL  ARRAY_MSG6
                00000000'EF   DD  063F                          PUSHL   LUN
                          03  DD  0645                          PUSHL   #3
            00000000'9F   04  FB  0647                          CALLS   #$$M, a#DS$ERRHARD
                              064E  585 210$:   $DS_CKLOOP      180$                    ; SCOPE LOOP?
      00000000'9F   FEB4 CF   FA  064E                          CALLG   180$, a#DS$CKLOOP
                              0657  586         $DS_ENDSUB
                              0657      T17_S5_X:
00000000'9F   00000048'EF   FA  0657                          CALLG   $S$, a#DS$ENDSUB
```

```
                                0662   588 ;++
                                0662   589 ; ERROR CONFIRMATION/IS TIMEOUT TEST
                                0662   590 ;--
                                0662   591         $DS_BGNSUB
                                0662       T17_S6::
00000000'9F   00000054'EF   FA   0662                 CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8   066D   592 120$:    BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8   0676   593          BISL    #MAINT_MODE,CR(R2)        ; PUT INTO MAINT MODE
                            067F   594          EXPECT_INTER                      ; ARM INTERRUPTS
        00000000'8F   00    DA   067F                 MTPR    #0,#IPLR
        00000000'EF   01    90   0686                 MOVB    #1,EXP_INTERRUPT
              00000000'EF   94   068D                 CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8   0693                 BISL    #INTERRUPT_EN,CR(R2)
          57   52   D0   069C   595          MOVL    R2,R7                     ; GET ADDRESS OF RH750
     58   57   00'8F   78   069F   596          ASHL    #-PF_FIELD,R7,R8          ; PUT PFN INTO R8
    58   00000000'8F   C8   06A4   597          BISL    #VALID_BIT,R8             ; VALIDATE MAP ENTRY
59  57   00000000'8F   CB   06AB   598          BICL3   #MAP_PTR_MSK,R7,R9        ; CREATE VIRTUAL ADDRESS IN R9
        0000'C2   59   D0   06B3   599          MOVL    R9,VAR(R2)                ; WRITE VIRTUAL ADDRESS
        0800 C2   58   D0   06B8   600          MOVL    R8, X800(R2)              ; WRITE MAP REG 0
        0000'C2   01   CE   06BD   601          MNEGL   #1,BCR(R2)                ; WRITE BYTE COUNT
         00000000'8F   DD   06C2   602          PUSHL   #READ                     ; PUSH WRITE COMMAND FOR XFER
                  01   DD   06C8   603          PUSHL   #1                        ; PUSH BYTE COUNT
                  52   DD   06CA   604          PUSHL   R2                        ; PUSH RH750 ADDRESS
    00000000'EF   03   FB   06CC   605          CALLS   #3,MM_XFER                ; INITIATE XFER
        53   0000'C2   D0   06D3   606          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
    54   00000000'8F   D0   06D8   607          MOVL    #DATA_XFER_DONE!DATA_XFER_ABRT,R4
    54   00000000'8F   C8   06DF   608          BISL    #ERR_STAT,R4              ; EXPECT THESE BITS
    54   00000000'EF   C8   06E6   609          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
1B  00000000'EF   00   E4   06ED   610          BBSC    #0,INT_OCCURRED,125$      ; BRANCH IF INTERRUPT OCCURRED
                            06F5   611          $DS_ERRHARD_S   #1,LUN,-
                            06F5   612                          ARRAY_MSG3,-
                            06F5   613                          PRINT_INTERR      ; REPORT INTERRUPT FAILURE
         00000000'EF   DF   06F5                       PUSHAL  PRINT_INTERR
         00000000'EF   DF   06FB                       PUSHAL  ARRAY_MSG3
         00000000'EF   DD   0701                       PUSHL   LUN
                  01   DD   0707                       PUSHL   #1
    00000000'9F   04   FB   0709                       CALLS   #$$M, a#DS$ERRHARD
          54   53   D1   0710   614 125$:    CMPL    R3,R4                     ; IS ERROR CONFIRMATION SET?
               1B   13   0713   615          BEQL    130$                      ; BRANCH IF ERR CONF GENERATED
                            0715   616          $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                            0715   617                          ARRAY_MSG3,-
                            0715   618                          PRINT_SBE
         00000000'EF   DF   0715                       PUSHAL  PRINT_SBE
         00000000'EF   DF   071B                       PUSHAL  ARRAY_MSG3
         00000000'EF   DD   0721                       PUSHL   LUN
                  02   DD   0727                       PUSHL   #2
    00000000'9F   04   FB   0729                       CALLS   #$$M, a#DS$ERRHARD
                            0730   619 130$:    $DS_CKLOOP      120$              ; SCOPE LOOP?
00000000'9F   FF39 CF   FA   0730                       CALLG   120$, a#DS$CKLOOP
    0000'C2   59   D0   0739   620          MOVL    R9,VAR(R2)                ; RESTORE THE VAR
    0000'C2   01   CE   073E   621          MNEGL   #1,BCR(R2)                ; AND BYTE COUNT
00000000'FF   00000000'8F   D0   0743   622          MOVL    #READ,aDRIVE0             ; ISSUE READ COMMAND
    54   00000000'8F   D0   074E   623          MOVL    #DT_BUSY,R4               ; SET UP EXPECTED RESULTS
    54   00000000'EF   C8   0755   624          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
        53   0000'C2   D0   075C   625          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
          54   53   D1   0761   626          CMPL    R3,R4                     ; COMPARE EXPECTED AND RECEIVED STATUS
               1B   13   0764   627          BEQL    135$                      ; BRANCH IF NO ERRORS
```

```
                                  0766   628          $DS_ERRHARD_S   #3,LUN,-          ; PRINT ERROR
                                  0766   629                          ARRAY_MSG3,-
                                  0766   630                          PRINT_SBE
              00000000'EF   DF    0766                        PUSHAL  PRINT_SBE
              00000000'EF   DF    076C                        PUSHAL  ARRAY_MSG3
              00000000'EF   DD    0772                        PUSHL   LUN
                        03  DD    0778                        PUSHL   #3
         00000000'9F    04  FB    077A                        CALLS   #$$M, a#DS$ERRHARD
                                  0781   631 135$:    $DS_CKLOOP      120$              ; SCOPE LOOP?
     00000000'9F   FEE8 CF   FA   0781                        CALLG   120$, a#DS$CKLOOP
                                  078A   632          $DS_ENDSUB
                                  078A       T17_S6_X:
  00000000'9F   00000054'EF   FA  078A                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0795   634 ;++
                                0795   635 ; PROGRAM ERROR INTERRUPT TEST
                                0795   636 ;--
                                0795   637          $DS_BGNSUB
                                0795          T17_S7::
    00000000'9F   00000060'EF   FA  0795                CALLG   $S$, a#DS$BGNSUB
        0000'C2   00000000'8F   C8  07A0   638 220$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH750
                        50      D4  07A9   639          CLRL    R0                        ; CLEAR DELAY COUNTER
                    0000'C2     D4  07AB   640          CLRL    VAR(R2)                   ; CLEAR VIRT ADR REG
                0000'C2   01    CE  07AF   641          MNEGL   #1,BCR(R2)                ; WRITE BYTE COUNTER
            54   00000000'8F    D0  07B4   642          MOVL    #PROGRAM_ERROR!DATA_XFER_DONE,R4 ; SET UP EXPECTED STATUS
        0000'C2   00000000'8F   C8  07BB   643          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
                                07C4   644          EXPECT_INTER                      ; ARM INTERRUPTS
            00000000'8F   00    DA  07C4                MTPR    #0,#IPLR
            00000000'EF   01    90  07CB                MOVB    #1,EXP_INTERRUPT
                00000000'EF     94  07D2                CLRB    INT_OCCURRED
        0000'C2   00000000'8F   C8  07D8                BISL    #INTERRUPT_EN,CR(R2)
            66   00000000'8F    D0  07E1   645          MOVL    #WRITE,(R6)               ; ISSUE WRITE COMMAND TO DRIVE
        0000'C2   00000000'8F   D0  07E8   646          MOVL    #SIM_OCC,DR(R2)           ; SET OCCUPIED
            66   00000000'8F    D0  07F1   647          MOVL    #READ,(R6)                ; ISSUE  READ
        0000'C2   00000000'8F   C8  07F8   648          BISL    #SIM_SCLK,DR(R2)          ; ASSERT SCLK
        0000'C2   00000000'8F   CA  0801   649          BICL    #SIM_SCLK,DR(R2)          ; DEASSERT SCLK
        0000'C2   00000000'8F   C8  080A   650          BISL    #SIM_EBL,DR(R2)           ; ASSERT EBL
        0000'C2   00000000'8F   CA  0813   651          BICL    #SIM_EBL,DR(R2)           ; DEASSERT EBL
            54   00000000'EF    C8  081C   652          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                53   0000'C2    D0  0823   653          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                  FC 50   0A    F3  0828   654 221$:    AOBLEQ  #10,R0,221$               ; WAIT FOR INTERRUPT TO OCCUR
     1B 00000000'EF   00        E4  082C   655          BBSC    #0,INT_OCCURRED,225$      ; IF INTERRUPT OCCURRED SKIP
                                0834   656          $DS_ERRHARD_S   #1,LUN,-
                                0834   657                          ARRAY_MSG58,-
                                0834   658                          PRINT_INTERR      ; PRINT ERROR
            00000000'EF   DF    0834                        PUSHAL  PRINT_INTERR
            00000000'EF   DF    083A                        PUSHAL  ARRAY_MSG58
            00000000'EF   DD    0840                        PUSHL   LUN
                        01      DD  0846                    PUSHL   #1
        00000000'9F   04        FB  0848                    CALLS   #$$M, a#DS$ERRHARD
                54   53         D1  084F   659 225$:    CMPL    R3,R4                     ; EXPECTED = RECEIVED?
                        1B      13  0852   660          BEQL    230$                      ; BRANCH IF PGE IS SET
                                0854   661          $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                                0854   662                          ARRAY_MSG5,-
                                0854   663                          PRINT_SBE
            00000000'EF   DF    0854                        PUSHAL  PRINT_SBE
            00000000'EF   DF    085A                        PUSHAL  ARRAY_MSG5
            00000000'EF   DD    0860                        PUSHL   LUN
                        02      DD  0866                    PUSHL   #2
        00000000'9F   04        FB  0868                    CALLS   #$$M, a#DS$ERRHARD
                                086F   664 230$:    $DS_CKLOOP      220$              ; SCOPE LOOP?
00000000'9F   FF2D CF           FA  086F                    CALLG   220$, a#DS$CKLOOP
            0000'C2   53        D0  0878   665          MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
            53   0000'C2        D0  087D   666          MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                        54      D4  0882   667          CLRL    R4                        ; CLEAR EXP RESULTS REG
        54   00000000'EF        C8  0884   668          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                54   53         D1  088B   669          CMPL    R3,R4                     ; CHECK RESULTS
                        1B      13  088E   670          BEQL    250$                      ; BRANCH IF STATUS IS OK
                                0890   671          $DS_ERRHARD_S   #3,LUN,-           ; PRINT ERROR
                                0890   672                          ARRAY_MSG5,-
                                0890   673                          PRINT_SBE
```

```
                00000000'EF   DF  0890                          PUSHAL  PRINT_SBE
                00000000'EF   DF  0896                          PUSHAL  ARRAY_MSG5
                00000000'EF   DD  089C                          PUSHL   LUN
                          03  DD  08A2                          PUSHL   #3
             00000000'9F  04   FB  08A4                          CALLS   #$$M, a#DS$ERRHARD
                               08AB  674 250$:   $DS_CKLOOP      220$                ; SCOPE LOOP?
        00000000'9F   FEF1 CF   FA  08AB                          CALLG   220$, a#DS$CKLOOP
                               08B4  675         $DS_ENDSUB
                               08B4      T17_S7_X:
    00000000'9F   00000060'EF   FA  08B4                          CALLG   $S$, a#DS$ENDSUB
```

```
                                    08BF    677 ;++
                                    08BF    678 ; DATA LATE--READ FROM THE DRIVE, SILO FULL AND SCLK RECEIVED
                                    08BF    679 ;--
                                    08BF    680         $DS_BGNSUB
                                    08BF            T17_S8::
    00000000'9F    0000006C'EF  FA  08BF                    CALLG   $S$, a#DS$BGNSUB
        0000'C2    00000000'8F  C8  08CA    681 260$:      BISL    #PGM_INIT,CR(R2)        ; INIT THE RH
        0000'C2    00000000'8F  C8  08D3    682            BISL    #MAINT_MODE,CR(R2)      ; PUT INTO MAINTENANCE MODE
                 0000'C2    32  CE  08DC    683            MNEGL   #50,BCR(R2)             ; SET UP DUMMY BYTE COUNT
        0000'C2    00000000'8F  C8  08E1    684            BISL    #BLKSND_COMD,DR(R2)     ; SET UP TO CAUSE OVERFLOW DATA LATE
                          50    D4  08EA    685            CLRL    R0                      ; CLEAR SCLK COUNTER
                     0000'C2    D4  08EC    686            CLRL    VAR(R2)                 ; CAUSE SELECTION OF MAP0
                                    08F0    687            EXPECT_INTER                    ; ARM INTERRUPTS
             00000000'8F    00  DA  08F0                    MTPR    #0,#IPLR
             00000000'EF    01  90  08F7                    MOVB    #1,EXP_INTERRUPT
                  00000000'EF   94  08FE                    CLRB    INT_OCCURRED
                  00000000'8F   C8  0904                    BISL    #INTERRUPT_EN,CR(R2)
        0000'C2   00000000'8F   C8  090D    688            MOVL    #READ,(R6)              ; ISSUE READ COMMAND
        0000'C2   00000000'8F   C8  0914    689            BISL    #SIM_OCC,DR(R2)         ; PREVNT MISSED XFER
        0000'C2   00000000'8F   C8  091D    690 270$:      BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
        0000'C2   00000000'8F   CA  0926    691            BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
                   EA 50    15  F3  092F    692            AOBLEQ  #21,R0,270$             ; ISSUE 22 SCLK
        0000'C2   00000000'8F   C8  0933    693            BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
        0000'C2   00000000'8F   CA  093C    694            BICL    #SIM_EBL,DR(R2)         ; NEGATE EBL
             54   00000000'8F   D0  0945    695            MOVL    #DATA_XFER_LATE,R4      ; SET UP EXPECTED STATUS
             54   00000000'8F   C8  094C    696            BISL    #DATA_XFER_DONE!DATA_XFER_ABRT,R4  ; MERGE THESE BITS
             54   00000000'EF   C8  0953    697            BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                53    0000'C2   D0  095A    698            MOVL    SR(R2),R3               ; READ STATUS
          1B 00000000'EF    00   E4  095F    699            BBSC    #0,INT_OCCURRED,275$    ; NO ERROR IF INTERRUPT OCCURRED
                                    0967    700            $DS_ERRHARD_S   #1,LUN,-
                                    0967    701                            ARRAY_MSG59,-
                                    0967    702                            PRINT_INTERR    ;  INTERRUPT ERROR
                  00000000'EF   DF  0967                        PUSHAL  PRINT_INTERR
                  00000000'EF   DF  096D                        PUSHAL  ARRAY_MSG59
                  00000000'EF   DD  0973                        PUSHL   LUN
                          01    DD  0979                        PUSHL   #1
             00000000'9F    04   FB  097B                        CALLS   #$$M, a#DS$ERRHARD
                     54     53   D1  0982    703 275$:      CMPL    R3,R4                   ; SEE IF STATUS IS CORRECT
                            1B   13  0985    704            BEQL    280$                    ; BRANCH IF STATUS IS OK
                                    0987    705            $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                    0987    706                            ARRAY_MSG6,-
                                    0987    707                            PRINT_SBE
                  00000000'EF   DF  0987                        PUSHAL  PRINT_SBE
                  00000000'EF   DF  098D                        PUSHAL  ARRAY_MSG6
                  00000000'EF   DD  0993                        PUSHL   LUN
                          02    DD  0999                        PUSHL   #2
             00000000'9F    04   FB  099B                        CALLS   #$$M, a#DS$ERRHARD
                                    09A2    708 280$:      $DS_CKLOOP      260$            ; SCOPE LOOP?
        00000000'9F    FF24 CF  FA  09A2                        CALLG   260$, a#DS$CKLOOP
    00000000'FF    00000000'8F  D0  09AB    709 290$:      MOVL    #READ,@DRIVE0           ; ISSUE DRIVE COMMAND
             54    00000000'8F  D0  09B6    710            MOVL    #DT_BUSY,R4             ; SET UP EXPECTED VALUE
             54    00000000'EF  C8  09BD    711            BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                53    0000'C2   D0  09C4    712            MOVL    SR(R2),R3               ; READ RH750 STATUS
                      54    53  D1  09C9    713            CMPL    R3,R4                   ; CHECK STATUS
                            1B   13  09CC    714            BEQL    300$                    ; BRANCH IF STATUS OK
                                    09CE    715            $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                                    09CE    716                            ARRAY_MSG6,-
```

```
                                  09CE   717                          PRINT_SBE
              00000000'EF   DF    09CE                       PUSHAL  PRINT_SBE
              00000000'EF   DF    09D4                       PUSHAL  ARRAY_MSG6
              00000000'EF   DD    09DA                       PUSHL   LUN
                        03  DD    09E0                       PUSHL   #3
         00000000'9F    04  FB    09E2                       CALLS   #$$M, a#DS$ERRHARD
                                  09E9   718 300$:   $DS_CKLOOP      260$              ; SCOPE LOOP
     00000000'9F   FEDD CF   FA   09E9                       CALLG   260$, a#DS$CKLOOP
                                  09F2   719         $DS_ENDSUB
                                  09F2       T17_S8_X:
  00000000'9F   0000006C'EF   FA   09F2                       CALLG   $S$, a#DS$ENDSUB
```

```
                                09FD   721 ;**
                                09FD   722 ; MASS BUS DATA PARITY ERROR DETECTION TEST
                                09FD   723 ;--
                                09FD   724          $DS_BGNSUB
                                09FD           T17_S9::
00000000'9F   00000078'EF   FA  09FD                   CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0A08   725 80$:    BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
    0000'C2   00000000'8F   C8  0A11   726          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
    0000'C2   00000000'8F   C8  0A1A   727          BISL    #INVRT_MB_DPAR,DR(R2)     ; SET UP TO INV DATA PARITY ON MASSBUS
                                0A23   728
        57   00000000'EF   DE  0A23   729          MOVAL   MIOBUFFER,R7              ; GET BUFFER ADDRESS
         59   57   00'8F   78  0A2A   730          ASHL    #-PF_FIELD,R7,R9          ; PUT PFN INTO R9
        59   00000000'8F   C8  0A2F   731          BISL    #VALID_BIT,R9             ; SET V BIT
            0800 C2   59   D0  0A36   732          MOVL    R9, X800(R2)              ; WRITE MAP REGISTER 0
        57   00000000'8F   CA  0A3B   733          BICL    #MAP_PTR_MSK,R7           ; CLEAR MAP SELECT FIELD IN VAR ENTRY
            0000'C2   57   D0  0A42   734          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS FOR READ
            0000'C2   04   CE  0A47   735          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT = 4 INTO BCR
             00000000'8F   DD  0A4C   736          PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                   04   DD  0A52   737          PUSHL   #4                        ; PUSH BYTE COUNT
                   52   DD  0A54   738          PUSHL   R2                        ; PUSH RH750 ADDRESS
        00000000'EF   03   FB  0A56   739          CALLS   #3,MM_XFER                ; INITIATE TRANSFER
    0000'C2   00000000'8F   CA  0A5D   740          BICL    #INVRT_MB_DPAR,DR(R2)     ; CLEAR INVERT BIT
                                0A66   741
                                0A66   742          EXPECT_INTER                      ; ARM INTERRUPTS
        00000000'8F   00   DA  0A66                   MTPR    #0,#IPLR
        00000000'EF   01   90  0A6D                   MOVB    #1,EXP_INTERRUPT
            00000000'EF   94  0A74                   CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8  0A7A                   BISL    #INTERRUPT_EN,CR(R2)
            0000'C2   57   D0  0A83   743          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS FOR READ
            0000'C2   04   CE  0A88   744          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT = 4 INTO BCR
             00000000'8F   DD  0A8D   745          PUSHL   #READ                     ; PUSH READ COMMAND FOR MM_XFER
                   04   DD  0A93   746          PUSHL   #4                        ; PUSH BYTE COUNT
                   52   DD  0A95   747          PUSHL   R2                        ; PUSH RH750 ADDRESS
    00000000'EF   03   FB  0A97   748          CALLS   #3,MM_XFER                ; INITIATE TRANSFER
                                0A9E   749
    54   00000000'8F   D0  0A9E   750          MOVL    #MASS_DATA_PE!DATA_XFER_ABRT,R4 ; EXPECTED VALUE
    54   00000000'8F   C8  0AA5   751          BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
                                0AAC   752          $DS_WAITUS_S    #75               ; WAIT 750 US
                   00   DD  0AAC                   PUSHL   #0
         0000004B 8F   DD  0AAE                   PUSHL   #75
    00000000'9F   02   FB  0AB4                   CALLS   #2, a#DS$WAITUS
    54   00000000'EF   C8  0ABB   753          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
        53   0000'C2   D0  0AC2   754          MOVL    SR(R2),R3                 ; READ STATUS
1B 00000000'EF   00   E4  0AC7   755          BBSC    #0,INT_OCCURRED,85$       ; BRANCH IF INTERRUPT RECEIVED
                                0ACF   756          $DS_ERRHARD_S   #1,LUN,-
                                0ACF   757                          ARRAY_MSG59,-
                                0ACF   758                          PRINT_INTERR      ; PRINT ERROR
        00000000'EF   DF  0ACF                   PUSHAL  PRINT_INTERR
        00000000'EF   DF  0AD5                   PUSHAL  ARRAY_MSG59
        00000000'EF   DD  0ADB                   PUSHL   LUN
                   01   DD  0AE1                   PUSHL   #1
    00000000'9F   04   FB  0AE3                   CALLS   #$$M, a#DS$ERRHARD
            54   53   D1  0AEA   759 85$:    CMPL    R3,R4                     ; IS MASSBUS DATA PE SET?
                 1B   13  0AED   760          BEQL    90$                       ; BRANCH IF OK
                                0AEF   761          $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                                0AEF   762                          ARRAY_MSG6,-
                                0AEF   763                          PRINT_SBE
```

```
                00000000'EF   DF  0AEF                          PUSHAL  PRINT_SBE
                00000000'EF   DF  0AF5                          PUSHAL  ARRAY_MSG6
                00000000'EF   DD  0AFB                          PUSHL   LUN
                          02  DD  0B01                          PUSHL   #2
              00000000'9F   04  FB  0B03                          CALLS   #$$M, a#DS$ERRHARD
                                0B0A    764 90$:    $DS_CKLOOP      80$                 ; SCOPE LOOP?
          00000000'9F   FEFA CF   FA  0B0A                          CALLG   80$, a#DS$CKLOOP
                0000'C2   53    D0  0B13    765         MOVL    R3,SR(R2)               ; WRITE STATUS BITS TO CLEAR
                53   0000'C2    D0  0B18    766         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                          54    D4  0B1D    767         CLRL    R4                      ; CLEAR EXP RESULTS REG
              54   00000000'EF   C8  0B1F    768         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                     54   53    D1  0B26    769         CMPL    R3,R4                   ; CHECK RESULTS
                          1B    13  0B29    770         BEQL    95$                     ; IF BITS CLEARED THEN BRANCH
                                0B2B    771         $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                                0B2B    772                         ARRAY_MSG6,-
                                0B2B    773                         PRINT_SBE
                00000000'EF   DF  0B2B                          PUSHAL  PRINT_SBE
                00000000'EF   DF  0B31                          PUSHAL  ARRAY_MSG6
                00000000'EF   DD  0B37                          PUSHL   LUN
                          03  DD  0B3D                          PUSHL   #3
              00000000'9F   04  FB  0B3F                          CALLS   #$$M, a#DS$ERRHARD
                                0B46    774 95$:    $DS_CKLOOP      80$                 ; SCOPE LOOP?
          00000000'9F   FEBE CF   FA  0B46                          CALLG   80$, a#DS$CKLOOP
                                0B4F    775         $DS_ENDSUB
                                0B4F        T17_S9_X:
    00000000'9F   00000078'EF   FA  0B4F                          CALLG   $S$, a#DS$ENDSUB
```

```
                                0B5A   777 ;++
                                0B5A   778 ; MASS BUS CONTROL PARITY ERROR DETECTION TEST
                                0B5A   779 ;--
                                0B5A   780          $DS_BGNSUB
                                0B5A               T17_S10::
00000000'9F   00000084'EF   FA  0B5A                       CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0B65   781 100$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH
    0000'C2   00000000'8F   C8  0B6E   782          BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINT MODE
    0000'C2   00000000'8F   C8  0B77   783          BISL    #INVRT_MB_CPAR,DR(R2)     ; SET UP TO INVERT MB CONTROL PARITY
                                0B80   784          EXPECT_INTER                      ; ARM INTERRUPTS
        00000000'8F   00    DA  0B80                       MTPR    #0,#IPLR
        00000000'EF   01    90  0B87                       MOVB    #1,EXP_INTERRUPT
              00000000'EF   94  0B8E                       CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8  0B94                       BISL    #INTERRUPT_EN,CR(R2)
        56    00000000'EF   D0  0B9D   785          MOVL    DRIVE0,R6                 ; MOVE ADDRESS OF DRIVE 0 TO R6
                 53   66    D0  0BA4   786          MOVL    (R6),R3                   ; READ REG 0 DRIVE 0
              53    0000'C2  D0  0BA7   787          MOVL    SR(R2),R3                 ; READ STATUS REGISTER--MASSBUS
                                0BAC   788                                            ; CONTROL PARITY ERROR SHOULD BE SET
                    0000'C2   D4  0BAC   789          CLRL    DR(R2)                    ; CLEAR INVERT MB CONTROL PARITY
        54    00000000'8F   D0  0BB0   790          MOVL    #MASS_CNTRL_PE,R4         ; SET UP EXPECTED VALUE REGISTER
        54    00000000'EF   C8  0BB7   791          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
     1B 00000000'EF   00    E4  0BBE   792          BBSC    #0,INT_OCCURRED,105$      ; BRANCH IF INTERRUPT OCCURRED
                                0BC6   793          $DS_ERRHARD_S   #1,LUN,-
                                0BC6   794                          ARRAY_MSG57,-
                                0BC6   795                          PRINT_INTERR      ; REPORT INTERRUPT FAILURE
              00000000'EF   DF  0BC6                       PUSHAL  PRINT_INTERR
              00000000'EF   DF  0BCC                       PUSHAL  ARRAY_MSG57
              00000000'EF   DD  0BD2                       PUSHL   LUN
                    01      DD  0BD8                       PUSHL   #1
        00000000'9F   04    FB  0BDA                       CALLS   #$$M, a#DS$ERRHARD
              54    53    D1  0BE1   796 105$:    CMPL    R3,R4                     ; IS CONTROL PARITY ERROR SET?
                    1B    13  0BE4   797          BEQL    110$                      ; BRANCH IF ERROR BIT GENERATED
                                0BE6   798          $DS_ERRHARD_S   #2,LUN,-       ; PRINT ERROR
                                0BE6   799                          ARRAY_MSG4,-
                                0BE6   800                          PRINT_SBE
           00000000'EF   DF  0BE6                       PUSHAL  PRINT_SBE
           00000000'EF   DF  0BEC                       PUSHAL  ARRAY_MSG4
           00000000'EF   DD  0BF2                       PUSHL   LUN
                    02   DD  0BF8                       PUSHL   #2
        00000000'9F   04    FB  0BFA                       CALLS   #$$M, a#DS$ERRHARD
                                0C01   801 110$:    $DS_CKLOOP      100$              ; SCOPE LOOP?
00000000'9F   FF60  CF      FA  0C01                       CALLG   100$, a#DS$CKLOOP
          0000'C2   53      D0  0C0A   802          MOVL    R3,SR(R2)                 ; WRITE STATUS BITS TO CLEAR
          53   0000'C2      D0  0C0F   803          MOVL    SR(R2),R3                 ; READ STATUS REGISTER-- SHOULD BE 0
                    54      D4  0C14   804          CLRL    R4                        ; CLEAR EXP RESULTS REG
      54    00000000'EF     C8  0C16   805          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
              54    53      D1  0C1D   806          CMPL    R3,R4                     ; CHECK RESULTS
                    1B      13  0C20   807          BEQL    115$                      ; BRANCH IF SO
                                0C22   808          $DS_ERRHARD_S   #3,LUN,-       ; PRINT ERROR
                                0C22   809                          ARRAY_MSG4,-
                                0C22   810                          PRINT_SBE
           00000000'EF   DF  0C22                       PUSHAL  PRINT_SBE
           00000000'EF   DF  0C28                       PUSHAL  ARRAY_MSG4
           00000000'EF   DD  0C2E                       PUSHL   LUN
                    03   DD  0C34                       PUSHL   #3
        00000000'9F   04    FB  0C36                       CALLS   #$$M, a#DS$ERRHARD
                                0C3D   811 115$:    $DS_CKLOOP      100$              ; SCOPE LOOP?
```

```
                00000000'9F    FF24 CF    FA  0C3D                    CALLG   100$, a#DS$CKLOOP
                                          0C46   812          $DS_ENDSUB
                                          0C46       T17_S10_X:
    00000000'9F    00000084'EF    FA  0C46                    CALLG   $S$, a#DS$ENDSUB
```

```
                                0C51    814 ;++
                                0C51    815 ; MAP PARITY ERROR DETECTION TEST
                                0C51    816 ;--
                                0C51    817         $DS_BGNSUB
                                0C51            T17_S11::
00000000'9F   00000090'EF   FA  0C51                    CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  0C5C    818 140$:   BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
    0000'C2   00000000'8F   C8  0C65    819         BISL    #MAINT_MODE,CR(R2)      ; PUT MUT INTO MAINTENANCE MODE
    0000'C2   00000000'8F   C8  0C6E    820         BISL    #INVRT_MAP_PAR,DR(R2)   ; PREPARE FOR INVERTING MAP PARITY
         57   00000000'EF   DE  0C77    821         MOVAL   MIOBUFFER,R7            ; R7 <-- BUFFER ADDRESS
          58    57    F7 8F   78  0C7E    822         ASHL    #-9,R7,R8               ; NORMALIZE FOR MAP ENTRY
         58   00000000'8F   C8  0C83    823         BISL    #VALID_BIT,R8           ; VALIDATE MAP ENTRY
              0800 C2   58   D0  0C8A    824         MOVL    R8, X800(R2)            ; WRITE MAP 0
         57   00000000'8F   CA  0C8F    825         BICL    #MAP_PTR_MSK,R7         ; CLEAR MAP SELECTOR IN VA
                                0C96    826         EXPECT_INTER                    ; ARM INTERRUPTS
                                0C96                    MTPR    #0,#IPLR
         00000000'8F   00   DA  0C96                    MOVB    #1,EXP_INTERRUPT
              00000000'EF   01   90  0C9D                    CLRB    INT_OCCURRED
              00000000'EF   94   0CA4                    BISL    #INTERRUPT_EN,CR(R2)
    0000'C2   00000000'8F   C8  0CAA
              0000'C2   57   D0  0CB3    827         MOVL    R7,VAR(R2)              ; WRITE THE VAR
              0000'C2   04   CE  0CB8    828         MNEGL   #4,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
               00000000'8F   DD  0CBD    829         PUSHL   #WRITE                  ; PUSH WRITE COMMAND FOR MM_XFER
                        04   DD  0CC3    830         PUSHL   #4                      ; PUSH BYTE COUNT
                        52   DD  0CC5    831         PUSHL   R2                      ; PUSH RH750 ADDRESS
         00000000'EF   03   FB  0CC7    832         CALLS   #3,MM_XFER              ; INITIATE TRANSFER
         54   00000000'8F   D0  0CCE    833         MOVL    #MAP_PE!DATA_XFER_ABRT,R4 ; SET UP EXPECTED RESULTS
         54   00000000'8F   C8  0CD5    834         BISL    #DATA_XFER_DONE,R4      ; EXPECT DT COMPLETE
         54   00000000'EF   C8  0CDC    835         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              53   0000'C2   D0  0CE3    836         MOVL    SR(R2),R3               ; READ STATUS EXPECT MAP PARITY ERROR
   1B 00000000'EF   00   E4  0CE8    837         BBSC    #0,INT_OCCURRED,145$    ; BRANCH IF INTERRUPT RECEIVED
                                0CF0    838         $DS_ERRHARD_S   #1,LUN,-
                                0CF0    839                         ARRAY_MSG57,-
                                0CF0    840                         PRINT_INTERR    ; PRINT ERROR
              00000000'EF   DF  0CF0                        PUSHAL  PRINT_INTERR
              00000000'EF   DF  0CF6                        PUSHAL  ARRAY_MSG57
              00000000'EF   DD  0CFC                        PUSHL   LUN
                        01   DD  0D02                        PUSHL   #1
         00000000'9F   04   FB  0D04                        CALLS   #$$M, a#DS$ERRHARD
              54   53   D1  0D0B    841 145$:   CMPL    R3,R4                   ; IS STATUS AS EXPECTED?
                   1B   13  0D0E    842         BEQL    150$                    ; BRANCH IF OK
                                0D10    843         $DS_ERRHARD_S   #2,LUN,-        ; PRINT ERROR
                                0D10    844                         ARRAY_MSG4,-
                                0D10    845                         PRINT_SBE
              00000000'EF   DF  0D10                        PUSHAL  PRINT_SBE
              00000000'EF   DF  0D16                        PUSHAL  ARRAY_MSG4
              00000000'EF   DD  0D1C                        PUSHL   LUN
                        02   DD  0D22                        PUSHL   #2
         00000000'9F   04   FB  0D24                        CALLS   #$$M, a#DS$ERRHARD
                                0D2B    846 150$:   $DS_CKLOOP      140$            ; SCOPE LOOP?
00000000'9F   FF2D CF   FA  0D2B                        CALLG   140$, a#DS$CKLOOP
         0000'C2   53   D0  0D34    847         MOVL    R3,SR(R2)               ; WRITE STATUS BITS TO CLEAR
         53   0000'C2   D0  0D39    848         MOVL    SR(R2),R3               ; READ STATUS REGISTER
                   54   D4  0D3E    849         CLRL    R4                      ; CLEAR EXP RESULTS REG
    54   00000000'EF   C8  0D40    850         BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
              54   53   D1  0D47    851         CMPL    R3,R4                   ; CHECK RESULTS
                   1B   13  0D4A    852         BEQL    155$                    ; IF STATUS IS CORRECT THEN BRANCH
                                0D4C    853         $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
```

```
                                0D4C   854                          ARRAY_MSG4,-
                                0D4C   855                          PRINT_SBE
                00000000'EF     DF     0D4C                 PUSHAL  PRINT_SBE
                00000000'EF     DF     0D52                 PUSHAL  ARRAY_MSG4
                00000000'EF     DD     0D58                 PUSHL   LUN
                          03    DD     0D5E                 PUSHL   #3
             00000000'9F    04   FB     0D60                 CALLS   #$$M, a#DS$ERRHARD
                                0D67   856 155$:   $DS_CKLOOP       140$                ; SCOPE LOOP?
          00000000'9F   FEF1 CF   FA     0D67                 CALLG   140$, a#DS$CKLOOP
                                0D70   857         $DS_ENDSUB
                                0D70       T17_S11_X:
    00000000'9F   00000090'EF   FA     0D70                 CALLG   $S$, a#DS$ENDSUB
```

```
                                0D7B   859          $DS_BGNSUB
                                0D7B          T17_S12::
    00000000'9F   0000009C'EF   FA  0D7B                       CALLG   $S$, a#DS$BGNSUB
                                0D86   860 ;++
                                0D86   861 ; INVALID MAP DETECTION TEST
                                0D86   862 ;--
        0000'C2   00000000'8F   C8  0D86   863 160$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH
        0000'C2   00000000'8F   C8  0D8F   864         BISL    #MAINT_MODE,CR(R2)        ; PUT MUT INTO MAINT MODE
             57   00000000'EF   DE  0D98   865         MOVAL   MIOBUFFER,R7              ; SET UP VA AND MAP
              58    57    F7 8F   78  0D9F   866         ASHL    #-9,R7,R8                 ; PUT PFN INTO R8
             57   00000000'8F   CA  0DA4   867         BICL    #MAP_PTR_MSK,R7           ; CREATE VIRTUAL ADDRESS
             58   00000000'8F   CA  0DAB   868         BICL    #VALID_BIT,R8             ; CLEAR VALID BIT IN MAP ENTRY
                0800 C2   58   D0  0DB2   869         MOVL    R8, X800(R2)              ; INVALIDATE MAP REG 0 ENTRY
                                0DB7   870         EXPECT_INTER                      ; ARM INTERRUPTS
             00000000'8F   00   DA  0DB7                       MTPR    #0,#IPLR
             00000000'EF   01   90  0DBE                       MOVB    #1,EXP_INTERRUPT
                  000000C0'EF   94  0DC5                       CLRB    INT_OCCURRED
        0000'C2   00000000'8F   C8  0DCB                       BISL    #INTERRUPT_EN,CR(R2)
                0000'C2   57   D0  0DD4   871         MOVL    R7,VAR(R2)                ; WRITE THE VAR
                0000'C2   04   CE  0DD9   872         MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
                0000'C2   57   D0  0DDE   873         MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
                 00000000'8F   DD  0DE3   874         PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                           04   DD  0DE9   875         PUSHL   #4                        ; PUSH BYTE COUNT
                           52   DD  0DEB   876         PUSHL   R2                        ; PUSH RH750 ADDRESS
             00000000'EF   03   FB  0DED   877         CALLS   #3,MM_XFER                ; INITIATE TRANSFER WITH INVALID MAP
             54   00000000'8F   D0  0DF4   878         MOVL    #DATA_XFER_ABRT!MAP_INVALID,R4 ; EXPECTED DATA
             54   00000000'8F   C8  0DFB   879         BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
             54   00000000'EF   C8  0E02   880         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
                53   0000'C2   D0  0E09   881         MOVL    SR(R2),R3                 ; MOVE STATUS REG INTO R3
          1B 00000000'EF   00   E4  0E0E   882         BBSC    #0,INT_OCCURRED,165$      ; BRANCH IF INTERRUPT OCCURRED
                                0E16   883         $DS_ERRHARD_S   #1,LUN,-
                                0E16   884                         ARRAY_MSG57,-
                                0E16   885                         PRINT_INTERR      ; REPORT INTERRUPT FAILURE
                00000000'EF   DF  0E16                       PUSHAL  PRINT_INTERR
                00000000'EF   DF  0E1C                       PUSHAL  ARRAY_MSG57
                00000000'EF   DD  0E22                       PUSHL   LUN
                           01   DD  0E28                       PUSHL   #1
             00000000'9F   04   FB  0E2A                       CALLS   #$$M, a#DS$ERRHARD
                     54   53   D1  0E31   886 165$:   CMPL    R3,R4                     ; IS STATUS AS EXPECTED?
                          1B   13  0E34   887         BEQL    170$                      ; BRANCH IF STATUS IS OK
                                0E36   888         $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                                0E36   889                         ARRAY_MSG4,-
                                0E36   890                         PRINT_SBE
                00000000'EF   DF  0E36                       PUSHAL  PRINT_SBE
                00000000'EF   DF  0E3C                       PUSHAL  ARRAY_MSG4
                00000000'EF   DD  0E42                       PUSHL   LUN
                           02   DD  0E48                       PUSHL   #2
             00000000'9F   04   FB  0E4A                       CALLS   #$$M, a#DS$ERRHARD
                                0E51   891 170$:   $DS_CKLOOP       160$              ; SCOPE LOOP?
    00000000'9F   FF31 CF   FA  0E51                       CALLG   160$, a#DS$CKLOOP
00000000'FF   00000000'8F   D0  0E5A   892         MOVL    #READ,aDRIVE0             ; ISSUE READ COMMAND TO CLEAR
        54   00000000'8F   D0  0E65   893         MOVL    #DT_BUSY,R4               ; EXPECT DATA TRANSFER BUSY
        54   00000000'EF   C8  0E6C   894         BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
            53   0000'C2   D0  0E73   895         MOVL    SR(R2),R3                 ; READ STATUS REGISTER
                 54   53   D1  0E78   896         CMPL    R3,R4                     ; IS STATUS INDICATING BUSY
                      1B   13  0E7B   897         BEQL    175$                      ; BRANCH IF STATUS OK
                                0E7D   898         $DS_ERRHARD_S   #3,LUN,-           ; PRINT ERROR
```

```
                                0E7D   899                          ARRAY_MSG4,-
                                0E7D   900                          PRINT_SBE
            00000000'EF   DF    0E7D                        PUSHAL  PRINT_SBE
            00000000'EF   DF    0E83                        PUSHAL  ARRAY_MSG4
            00000000'EF   DD    0E89                        PUSHL   LUN
                      03  DD    0E8F                        PUSHL   #3
      00000000'9F   04    FB    0E91                        CALLS   #$$M, a#DS$ERRHARD
                                0E98   901 175$:   $DS_CKLOOP       160$                ; SCOPE LOOP?
00000000'9F   FEEA CF    FA    0E98                        CALLG   160$, a#DS$CKLOOP
                                0EA1   902         $DS_ENDSUB
                                0EA1       T17_S12_X:
00000000'9F   0000009C'EF    FA 0EA1                        CALLG   $S$, a#DS$ENDSUB
```

```
                                    0EAC    904          $DS_BGNSUB
                                    0EAC                 T17_S13::
    00000000'9F   000000A8'EF   FA  0EAC                      CALLG   $S$, a#DS$BGNSUB
                                    0EB7    905 ;++
                                    0EB7    906 ; WRITE CHECK ERROR LOW TEST
                                    0EB7    907 ;--
        0000'C2   00000000'8F   C8  0EB7    908 180$:    BISL    #PGM_INIT,CR(R2)          ; INIT RH
        0000'C2   00000000'8F   C8  0EC0    909           BISL    #MAINT_MODE,CR(R2)        ; PUT RH750 INTO MAINTENANCE MODE
                                    0EC9    910           EXPECT_INTER                      ; ARM INTERRUPTS
             00000000'8F   00   DA  0EC9                      MTPR    #0,#IPLR
             00000000'EF   01   90  0ED0                      MOVB    #1,EXP_INTERRUPT
                  00000000'EF   94  0ED7                      CLRB    INT_OCCURRED
        0000'C2   00000000'8F   C8  0EDD                      BISL    #INTERRUPT_EN,CR(R2)
             57   00000000'EF   DE  0EE6    911           MOVAL   MIOBUFFER,R7              ; MOVE BUFFER ADDRESS TO R7
                       67        7C  0EED    912           CLRQ    (R7)                      ; CLEAR FIRST 8 BYTES OF BUFFER
              58   57   F7 8F   78  0EEF    913           ASHL    #-9,R7,R8                 ; PUT PAGE FRAME NUMBER INTO R8
             57   00000000'8F   CA  0EF4    914           BICL    #MAP_PTR_MSK,R7           ; CREATE VIRTUAL ADDR REGISTER ENTRY
                  0000'C2   57   D0  0EFB    915           MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
                  0000'C2   02   CE  0F00    916           MNEGL   #2,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
             58   00000000'8F   C8  0F05    917           BISL    #VALID_BIT,R8             ; SET V BIT IN MAP ENTRY
                  800 C2   58   D0  0F0C    918           MOVL    R8, X800(R2)              ; WRITE VALID MAP ENTRY INTO MAP REG 0
                  00000000'8F   DD  0F11    919           PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                       02       DD  0F17    920           PUSHL   #2                        ; PUSH BYTE COUNT
                       52       DD  0F19    921           PUSHL   R2                        ; PUSH RH750 ADDRESS
             00000000'EF   03   FB  0F1B    922           CALLS   #3,MM_XFER                ; INITIATE XFER--PUT 0 ONTO MASSBUS
                 0000'C2   57   D0  0F22    923           MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
                 0000'C2   04   CE  0F27    924           MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
             00000000'EF   03   C8  0F2C    925           BISL    #3,MIOBUFFER              ; SET TWO BITS IN OUTPUT BUFFER--
                                    0F33    926                                             ; DATA FOR WRITE CHECK WILL BE
                                    0F33    927                                             ; DIFFERENT THAN THAT ON THE MASSBUS
                  00000000'8F   DD  0F33    928           PUSHL   #WRITE_CHECK              ; PUSH COMMAND FOR MM_XFER
                       04       DD  0F39    929           PUSHL   #4                        ; PUSH BYTE COUNT
                       52       DD  0F3B    930           PUSHL   R2                        ; PUSH RH750 ADDRESS
             00000000'EF   03   FB  0F3D    931           CALLS   #3,MM_XFER                ; INITIATE WRITE CHECK
        54   00000000'8F   D0   0F44    932           MOVL    #WRITE_CHK_LOW!DATA_XFER_ABRT,R4 ; EXPECTED DATA
        54   00000000'8F   C8   0F4B    933           BISL    #DATA_XFER_DONE,R4        ; EXPECT DT COMPLETE
        54   00000000'EF   C8   0F52    934           BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
             53   0000'C2   D0   0F59    935           MOVL    SR(R2),R3                 ; READ STATUS AFTER WRITE CHECK
    1B 00000000'EF   00   E4   0F5E    936           BBSC    #0,INT_OCCURRED,185$      ; IF INTERRUPT OCCURRED BRANCH
                                    0F66    937           $DS_ERRHARD_S   #1,LUN,-
                                    0F66    938                           ARRAY_MSG57,-
                                    0F66    939                           PRINT_INTERR      ; PRINT ERROR
                  00000000'EF   DF  0F66                      PUSHAL  PRINT_INTERR
                  00000000'EF   DF  0F6C                      PUSHAL  ARRAY_MSG57
                  00000000'EF   DD  0F72                      PUSHL   LUN
                       01       DD  0F78                      PUSHL   #1
        00000000'9F   04   FB  0F7A                      CALLS   #$$M, a#DS$ERRHARD
                  54   53   D1  0F81    940 185$:    CMPL    R3,R4                     ; IS WRITE CHECK LOWER ERROR SET?
                       1B   13  0F84    941           BEQL    190$                      ; BRANCH IF STATUS IS OK
                                    0F86    942           $DS_ERRHARD_S   #2,LUN,-          ; PRINT ERROR
                                    0F86    943                           ARRAY_MSG4,-
                                    0F86    944                           PRINT_SBE
                  00000000'EF   DF  0F86                      PUSHAL  PRINT_SBE
                  00000000'EF   DF  0F8C                      PUSHAL  ARRAY_MSG4
                  00000000'EF   DD  0F92                      PUSHL   LUN
                       02       DD  0F98                      PUSHL   #2
        00000000'9F   04   FB  0F9A                      CALLS   #$$M, a#DS$ERRHARD
```

```
                                  0FA1   945 190$:   $DS_CKLOOP      180$                ; SCOPE LOOP?
     00000000'9F   FF12 CF   FA   0FA1                       CALLG   180$, a#DS$CKLOOP
  00000000'FF   00000000'8F   D0   0FAA   946         MOVL    #READ,aDRIVE0               ; ISSUE DATA XFER COMMAND
           54   00000000'8F   D0   0FB5   947         MOVL    #DT_BUSY,R4                 ; EXPECT DATA TRANSFER BUSY
           54   00000000'EF   C8   0FBC   948         BISL    RH11TB_MSK,R4               ; LOAD EXPECTED MASK BIT
               53   0000'C2   D0   0FC3   949         MOVL    SR(R2),R3                   ; READ STATUS REGISTER SB 0
                    54   53   D1   0FC8   950         CMPL    R3,R4                       ; STATUS BITS OKAY?
                         1B   13   0FCB   951         BEQL    195$                        ; BRANCH IF ONLY DTBUSY SET
                                  0FCD   952         $DS_ERRHARD_S   #3,LUN,-            ; PRINT ERROR
                                  0FCD   953                         ARRAY_MSG4,-
                                  0FCD   954                         PRINT_SBE
               00000000'EF   DF   0FCD                       PUSHAL  PRINT_SBE
               00000000'EF   DF   0FD3                       PUSHAL  ARRAY_MSG4
               00000000'EF   DD   0FD9                       PUSHL   LUN
                         03   DD   0FDF                       PUSHL   #3
           00000000'9F   04   FB   0FE1                       CALLS   #$$M, a#DS$ERRHARD
                                  0FE8   955 195$:   $DS_CKLOOP      180$                ; SCOPE LOOP?
     00000000'9F   FECB CF   FA   0FE8                       CALLG   180$, a#DS$CKLOOP
```

```
                                    0FF1   957          $DS_ENDSUB
                                    0FF1          T17_S13_X:
00000000'9F   000000A8'EF    FA     0FF1                       CALLG   $S$, a#DS$ENDSUB
                                    0FFC   958 ;++
                                    0FFC   959 ; WRITE CHECK ERROR HIGH TEST
                                    0FFC   960 ;--
                                    0FFC   961          $DS_BGNSUB
                                    0FFC          T17_S14::
00000000'9F   000000B4'EF    FA     0FFC                       CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F    C8     1007   962 200$:   BISL    #PGM_INIT,CR(R2)          ; INIT THE RH750 UNDER TEST
    0000'C2   00000000'8F    C8     1010   963          BISL    #MAINT_MODE,CR(R2)        ; PUT THE MUT INTO MAINTENANCE MODE
                                    1019   964          EXPECT_INTER                      ; ARM INTERRUPTS
         00000000'8F   00    DA     1019                       MTPR    #0,#IPLR
         00000000'EF   01    90     1020                       MOVB    #1,EXP_INTERRUPT
              00000000'EF    94     1027                       CLRB    INT_OCCURRED
    0000'C2   00000000'8F    C8     102D                       BISL    #INTERRUPT_EN,CR(R2)
              00000000'EF    D4     1036   965          CLRL    MIOBUFFER                 ; EMPTY MIDDLE I/O BUFFER
             0000'C2   57    D0     103C   966          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
             0000'C2   02    CE     1041   967          MNEGL   #2,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
              00000000'8F    DD     1046   968          PUSHL   #WRITE                    ; PUSH WRITE COMMAND FOR MM_XFER
                        02    DD     104C   969          PUSHL   #2                        ; PUSH BYTE COUNT FOR MM_XFER
                        52    DD     104E   970          PUSHL   R2                        ; PUSH RH750 ADDRESS
         00000000'EF   03    FB     1050   971          CALLS   #3,MM_XFER                ; INITIATE WRITE--PUT 0 ON THE MASSBUS
              0000'C2   57    D0     1057   972          MOVL    R7,VAR(R2)                ; WRITE VIRTUAL ADDRESS INTO VAR
              0000'C2   04    CE     105C   973          MNEGL   #4,BCR(R2)                ; WRITE BYTE COUNT INTO BCR
         00000001'EF   03    D0     1061   974          MOVL    #3,MIOBUFFER + 1          ; SET UP BAD DATA IN UPPER HALF OF
                                    1068   975                                            ; WRITE CHECK WORD
              00000000'8F   DD     1068   976          PUSHL   #WRITE_CHECK              ; PUSH COMMAND FOR MM_XFER
                        04    DD     106E   977          PUSHL   #4                        ; PUSH BYTE COUNT
                        52    DD     1070   978          PUSHL   R2                        ; PUSH RH750 ADDRESS
    00000000'EF   03    FB     1072   979          CALLS   #3,MM_XFER                ; INITIATE WRITE CHECK WITH BAD DATA
                                    1079   980                                            ; IN MEMORY
    54   00000000'8F    D0     1079   981          MOVL    #WRITE_CHK_HIGH!DATA_XFER_ABRT,R4 ; EXPECTED STATUS
    54   00000000'8F    C8     1080   982          BISL    #DATA_XFER_DONE,R4        ; EXPECT THIS BIT ALSO
    54   00000000'EF    C8     1087   983          BISL    RH11TB_MSK,R4             ; LOAD EXPECTED MASK BIT
        53   0000'C2    D0     108E   984          MOVL    SR(R2),R3                 ; READ STATUS REGISTER--
                                    1093   985                                            ; SHOULD HAVE WRITE CHECK ERROR HIGH
1B 00000000'EF   00    E4     1093   986          BBSC    #0,INT_OCCURRED,205$      ; BRANCH IF INTERRUPT OCCURRED
                                    109B   987          $DS_ERRHARD_S   #1,LUN,-
                                    109B   988                          ARRAY_MSG57,-
                                    109B   989                          PRINT_INTERR      ; PRINT ERROR
         00000000'EF   DF    109B                             PUSHAL  PRINT_INTERR
         00000000'EF   DF    10A1                             PUSHAL  ARRAY_MSG57
         00000000'EF   DD    10A7                             PUSHL   LUN
                   01  DD    10AD                             PUSHL   #1
    00000000'9F    04   FB    10AF                             CALLS   #$$M, a#DS$ERRHARD
              54   53   D1    10B6   990 205$:   CMPL    R3,R4                     ; IS STATUS AS EXPECTED
                   1B   13    10B9   991          BEQL    210$                      ; BRANCH IF OK
                                    10BB   992          $DS_ERRHARD_S   #2,LUN,-           ; PRINT ERROR
                                    10BB   993                          ARRAY_MSG4,-
                                    10BB   994                          PRINT_SBE
         00000000'EF   DF    10BB                             PUSHAL  PRINT_SBE
         00000000'EF   DF    10C1                             PUSHAL  ARRAY_MSG4
         00000000'EF   DD    10C7                             PUSHL   LUN
                   02  DD    10CD                             PUSHL   #2
    00000000'9F    04   FB    10CF                             CALLS   #$$M, a#DS$ERRHARD
                                    10D6   995 210$:   $DS_CKLOOP      200$              ; SCOPE LOOP?
```

```
      00000000'9F   FF2D CF   FA  10D6                          CALLG   200$, a#DS$CKLOOP
   00000000'FF   00000000'8F  D0  10DF   996          MOVL    #READ,aDRIVE0              ; ISSUE DT COMMAND
              53   0000'C2    D0  10EA   997          MOVL    SR(R2),R3                  ; READ STATUS REGISTER SB 0
          54   00000000'8F    D0  10EF   998          MOVL    #DT_BUSY,R4                ; SET UP EXPECTED VALUE
          54   00000000'EF    C8  10F6   999          BISL    RH11TB_MSK,R4              ; LOAD EXPECTED MASK BIT
                    54   53   D1  10FD  1000          CMPL    R3,R4                      ; EXPECTED = RECEIVED?
                         1B   13  1100  1001          BEQL    215$                       ; BRANCH IF SO
                                  1102  1002          $DS_ERRHARD_S   #3,LUN,-           ; PRINT ERROR
                                  1102  1003                          ARRAY_MSG4,-
                                  1102  1004                          PRINT_SBE
              00000000'EF    DF  1102                          PUSHAL  PRINT_SBE
              00000000'EF    DF  1108                          PUSHAL  ARRAY_MSG4
              00000000'EF    DD  110E                          PUSHL   LUN
                         03   DD  1114                          PUSHL   #3
          00000000'9F    04   FB  1116                          CALLS   #$$M, a#DS$ERRHARD
                                  111D  1005 215$:    $DS_CKLOOP      200$               ; SCOPE LOOP?
      00000000'9F   FEE6 CF   FA  111D                          CALLG   200$, a#DS$CKLOOP
```

```
                                1126  1007          $DS_ENDSUB
                                1126        T17_S14_X:
00000000'9F   000000B4'EF   FA  1126                CALLG   $S$, a#DS$ENDSUB
                                1131  1008 ;++
                                1131  1009 ; READ DATA TIME OUT TEST
                                1131  1010 ;--
                                1131  1011          $DS_BGNSUB
                                1131        T17_S15::
00000000'9F   000000C0'EF   FA  1131                CALLG   $S$, a#DS$BGNSUB
    0000'C2   00000000'8F   C8  113C  1012 220$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE RH
    0000'C2   00000000'8F   C8  1145  1013          BISL    #MAINT_MODE,CR(R2)      ; PUT INTO MAINTENANCE MODE
                            C8  114E  1014          EXPECT_INTER                    ; ARM INTERRUPTS
        00000000'8F   00    DA  114E                MTPR    #0,#IPLR
        00000000'EF   01    90  1155                MOVB    #1,EXP_INTERRUPT
            00000000'EF     94  115C                CLRB    INT_OCCURRED
    0000'C2   00000000'8F   C8  1162                BISL    #INTERRUPT_EN,CR(R2)
                  57   52   D0  116B  1015          MOVL    R2,R7                   ; SET UP TO TRANSFER FROM NON-EXISTENT
                            D0  116E  1016                                          ; NEXUS
        57   00001000 8F   C8  116E  1017          BISL    #BIT12,R7               ; SET PHYSICAL ADDRESS BIT 10
         58    57   00'8F   78  1175  1018          ASHL    #-PF_FIELD,R7,R8        ; GET PFN
        58   00000000'8F   C8  117A  1019          BISL    #VALID_BIT,R8           ; SET VALID BIT IN MAP ENTRY
        57   00000000'8F   CA  1181  1020          BICL    #MAP_PTR_MSK,R7         ; CLEAR MAP POINTER IN VIRTUAL ADDRESS
            0800 C2   58    D0  1188  1021          MOVL    R8, X800(R2)            ; WRITE VALIDATED PFN INTO MAP 0
            0000'C2   57    D0  118D  1022          MOVL    R7,VAR(R2)              ; WRITE VIRTUAL ADDRESS INTO VAR
            0000'C2   08    CE  1192  1023          MNEGL   #8,BCR(R2)              ; WRITE BYTE COUNT INTO BCR
             00000000'8F    DD  1197  1024          PUSHL   #WRITE                  ; PUSH WTD COMMAND FOR MM_XFER
                       08    DD  119D  1025          PUSHL   #8                      ; PUSH BYTE COUNT
                       52    DD  119F  1026          PUSHL   R2                      ; PUSH RH750 ADDRESS
        00000000'EF    03    FB  11A1  1027          CALLS   #3,MM_XFER              ; INITIATE WRITE TO DRIVE
                            11A8  1028                                              ; A READ DATA TIME OUT WILL OCCUR
        54   00000000'8F   D0  11A8  1029          MOVL    #NO_RESPONSE!DATA_XFER_ABRT,R4 ; EXPECTED STATUS
        54   00000000'8F   C8  11AF  1030          BISL    #DATA_XFER_DONE,R4      ; EXPECTED
        54   00000000'EF   C8  11B6  1031          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
            53   0000'C2   D0  11BD  1032          MOVL    SR(R2),R3               ; READ STATUS REGISTER
    1B 00000000'EF   00    E4  11C2  1033          BBSC    #0,INT_OCCURRED,225$    ; BRANCH IF INTERRUPT OCCURRED
                            11CA  1034          $DS_ERRHARD_S   #1,LUN,-
                            11CA  1035                          ARRAY_MSG3,-
                            11CA  1036                          PRINT_INTERR        ; REPORT INTERRUPT FAILURE
            00000000'EF     DF  11CA                    PUSHAL  PRINT_INTERR
            00000000'EF     DF  11D0                    PUSHAL  ARRAY_MSG3
            00000000'EF     DD  11D6                    PUSHL   LUN
                       01   DD  11DC                    PUSHL   #1
        00000000'9F    04   FB  11DE                    CALLS   #$$M, a#DS$ERRHARD
                  54   53   D1  11E5  1037 225$:   CMPL    R3,R4                   ; IS STATUS AS EXPECTED?
                       1B   13  11E8  1038          BEQL    230$                    ; BRANCH IF STATUS IS OK
                            11EA  1039          $DS_ERRHARD_S   #2,LUN,-         ; PRINT ERROR
                            11EA  1040                          ARRAY_MSG3,-
                            11EA  1041                          PRINT_SBE
            00000000'EF     DF  11EA                    PUSHAL  PRINT_SBE
            00000000'EF     DF  11F0                    PUSHAL  ARRAY_MSG3
            00000000'EF     DD  11F6                    PUSHL   LUN
                       02   DD  11FC                    PUSHL   #2
        00000000'9F    04   FB  11FE                    CALLS   #$$M, a#DS$ERRHARD
                            1205  1042 230$:    $DS_CKLOOP      220$                ; SCOPE LOOP?
    00000000'9F   FF33 CF   FA  1205                    CALLG   220$, a#DS$CKLOOP
00000000'FF    00000000'8F  D0  120E  1043          MOVL    #READ,aDRIVE0           ; ISSUE DT COMMAND
            53   0000'C2   D0  1219  1044          MOVL    SR(R2),R3               ; READ STATUS REGISTER
```

```
                54   00000000'8F   D0  121E  1045          MOVL    #DT_BUSY,R4             ; SET UP EXPECTED RESULTS
                54   00000000'EF   C8  1225  1046          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
                     54      53   D1  122C  1047          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                             1B   13  122F  1048          BEQL    240$                    ; BRANCH IF SO
                                      1231  1049          $DS_ERRHARD_S   #3,LUN,-       ; PRINT ERROR
                                      1231  1050                          ARRAY_MSG3,-
                                      1231  1051                          PRINT_SBE
                     00000000'EF   DF  1231                PUSHAL  PRINT_SBE
                     00000000'EF   DF  1237                PUSHAL  ARRAY_MSG3
                     00000000'EF   DD  123D                PUSHL   LUN
                             03    DD  1243                PUSHL   #3
                00000000'9F   04   FB  1245                CALLS   #$$M, a#DS$ERRHARD
                                      124C  1052 240$:      $DS_CKLOOP      220$            ; SCOPE LOOP?
           00000000'9F   FEEC CF   FA  124C                CALLG   220$, a#DS$CKLOOP
                                      1255  1053          $DS_ENDSUB
                                      1255          T17_S15_X:
      00000000'9F   000000C0'EF   FA  1255                CALLG   $S$, a#DS$ENDSUB
                                      1260  1054 $DS_ENDTEST
                     50   01   D0  1260                MOVL    #1, R0                  ; NORMAL EXIT
                                      1263          TEST_017_X::
                00000000'9F   6E   FA  1263                CALLG   (SP), a#DS$BREAK
                                   04  126A                RET                             ; RETURN TO TEST SEQUENCER
                                      126B  1055 $DS_PAGE
```

```
                        126B                .SBTTL  TEST 18:  ATTENTION SUMMARY REGISTER TESTS
                     00000000                       .PSECT  TEST_018, PAGE, NOWRT
                         0024
                         0024  1058 $DS_BGNTEST     <DEFAULT,ALL>,,LONG
                         0024       DATA_018:
                     00000000    0024               .LONG   0                       ; TEST ARGUMENT TABLE TERMINATOR
                         0028       TEST_018::
                        0000 0028                   .WORD    M<>                    ; ENTRY MASK
                         002A  1059 ;++
                         002A  1060 ; TEST DESCRIPTION:
                         002A  1061 ;
                         002A  1062 ; THIS TEST VERIFYS ADDITIONAL RH750 LOGIC NOT TESTED IN
                         002A  1063 ; OTHER TESTS. THE FIRST SUBTEST CHECKS THAT SR<18> NED
                         002A  1064 ; DOES NOT GET SET WHEN READING THE ATTENTION SUMMARY REGISTER.
                         002A  1065 ; THE SECOND SUBTEST LOOKS FOR A NON-EXISTENT DRIVE, IF FOUND,
                         002A  1066 ; CLEARS NED, READS THE ATTENTION SUMMARY REGISTER AND CHECKS
                         002A  1067 ; THAT NED IS CLEAR.
                         002A  1068 ;
                         002A  1069 ;
                         002A  1070 ;*****************************************************************************
                         002A  1071 ; CAUTION:
                         002A  1072 ;
                         002A  1073 ;            DO NOT SINGLE STEP THROUGH THIS PROGRAM.
                         002A  1074 ;            DOING SO WILL CAUSE A MISSED TRANSFER ERROR.
                         002A  1075 ;
                         002A  1076 ;*****************************************************************************
                         002A  1077 ;--
                         002A  1078 $DS_BGNSUB
                         002A       T18_S1::
00000000'9F   000000CC'EF   FA   002A                       CALLG   $S$, a#DS$BGNSUB
                    FFC8'   30   0035  1079         BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
          52   00000000'EF   D0   0038  1080         MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
     0000'C2   00000000'8F   C8   003F  1081 10$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE RH
                         0048  1082         $DS_WAITUS_S    #10                     ; WAIT 100 US FOR RH11-TB
                      00   DD   0048                       PUSHL   #0
                      0A   DD   004A                       PUSHL   #10
          00000000'9F   02   FB   004C                       CALLS   #2, a#DS$WAITUS
                         0053  1083                                               ; (IF PRESENT) TO SETTLE
              55   0790 C2   D0   0053  1084         MOVL     X790(R2),R5            ; READ ATTENTION SUMMARY REGISTER
                      54   D4   0058  1085         CLRL    R4                      ; CLEAR EXPECTED RESULTS REGISTER
                      55   B5   005A  1086         TSTW    R5                      ; ANY ATTENTION BITS SET?
                      07   13   005C  1087         BEQL    15$                     ; SKIP IF NO ATTENTION
          54   00000000'8F   C8   005E  1088         BISL    #ATTENTION,R4           ; EXPECT ATTENTION IN STATUS REGISTER
               53   0000'C2   D0   0065  1089 15$:    MOVL    SR(R2),R3               ; READ STATUS REGISTER
                    54   53   D1   006A  1090         CMPL    R3,R4                   ; IS STATUS OK?
                      1B   13   006D  1091         BEQL    20$                     ; BRANCH IF NO NED
                         006F  1092         $DS_ERRHARD_S   #1,LUN,-                ; PRINT ERROR
                         006F  1093                         ARRAY_MSG58,-
                         006F  1094                         PRINT_SBE
               00000000'EF   DF   006F                       PUSHAL  PRINT_SBE
               00000000'EF   DF   0075                       PUSHAL  ARRAY_MSG58
               00000000'EF   DD   007B                       PUSHL   LUN
                      01   DD   0081                       PUSHL   #1
          00000000'9F   04   FB   0083                       CALLS   #$$M, a#DS$ERRHARD
                         008A  1095 20$:    $DS_CKLOOP      10$                     ; SCOPE LOOP?
    00000000'9F   B2 AF   FA   008A                       CALLG   10$, a#DS$CKLOOP
                         0092  1096 $DS_ENDSUB
```

```
                        0092      T18_S1_X:
00000000'9F   000000CC'EF   FA  0092                CALLG   $S$, a#DS$ENDSUB
```

```
                                009D  1098 $DS_BGNSUB
                                009D       T18_S2::
00000000'9F   000000D8'EF   FA  009D                 CALLG   $S$, a#DS$BGNSUB
                       5A   D4  00A8  1099           CLRL    R10                     ; CLEAR DRIVE DETECTED FLAG
            57   00000000'EF  D0  00AA  1100           MOVL    DRIVE0,R7               ; PUT DRIVE 0'S ADDRESS INTO R7
                       56   D4  00B1  1101           CLRL    R6                      ; CLEAR DRIVE COUNTER
      0000'C2   00000000'8F   C8  00B3  1102 10$:      BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                  5B   67      D0  00BC  1103           MOVL    (R7),R11                ; TRY TO READ EXT REG 0
              53   0000'C2     D0  00BF  1104           MOVL    SR(R2),R3               ; READ STATUS REGISTER
                1C 53   12     E0  00C4  1105           BBS     #18,R3,20$              ; IF NED IS SET BRANCH
            57   00000000'8F   C0  00C8  1106           ADDL    #DRIVE_OFFSET,R7        ; INCREMENT DRIVE ADDR POINTER
          FFDE 56   01   07     3D  00CF  1107           ACBW    #7,#1,R6,10$            ; CHECK NEXT DRIVE FOR NED
                                    00D5  1108           $DS_PRINTX_S    FMT_ALLDRVPRES  ; ALL DRIVES PRESENT,
                00000000'EF   9F  00D5                   PUSHAB  FMT_ALLDRVPRES
             00000000'9F   01   FB  00DB                 CALLS   #$$N, a#DS$PRINTX
                                    00E2  1109                                       ; CANNOT PERFORM SUBTEST 2
                       35   11  00E2  1110               BRB     30$                 ; BRANCH TO END OF TEST
      0000'C2   00000000'8F   C8  00E4  1111 20$:      BISL    #NON_XIST_DRIVE,SR(R2)  ; CLEAR NED BIT (WRITE ONE TO CLEAR)
              55   0790 C2     D0  00ED  1112           MOVL     X790(R2),R5            ; READ ATTN SUMMARY REGISTER
                       54   D4  00F2  1113           CLRL    R4                      ; CLEAR EXPECTED RESULTS REG
              53   0000'C2     D0  00F4  1114           MOVL    SR(R2),R3               ; READ STATUS REGISTER
                   54   53     D1  00F9  1115           CMPL    R3,R4                   ; NED SHOULD BE CLEAR
                       1B   13  00FC  1116           BEQL    30$                     ; BRANCH IF NED IS CLEAR
                                    00FE  1117           $DS_ERRHARD_S   #1,LUN,-     ; PRINT ERROR
                                    00FE  1118                           ARRAY_MSG5,-
                                    00FE  1119                           PRINT_SBE
                00000000'EF   DF  00FE                   PUSHAL  PRINT_SBE
                00000000'EF   DF  0104                   PUSHAL  ARRAY_MSG5
                00000000'EF   DD  010A                   PUSHL   LUN
                       01     DD  0110                   PUSHL   #1
             00000000'9F   04   FB  0112                 CALLS   #$$M, a#DS$ERRHARD
                                    0119  1120 30$:      $DS_CKLOOP              10$             ; SCOPE LOOP?
          00000000'9F   97 AF   FA  0119                 CALLG   10$, a#DS$CKLOOP
                                    0121  1121 $DS_ENDSUB
                                    0121       T18_S2_X:
00000000'9F   000000D8'EF   FA  0121                 CALLG   $S$, a#DS$ENDSUB
                                    012C  1122 $DS_ENDTEST
                    50   01   D0  012C                 MOVL    #1, R0                  ; NORMAL EXIT
                                    012F       TEST_018_X::
             00000000'9F   6E   FA  012F                 CALLG   (SP), a#DS$BREAK
                                04  0136                 RET                             ; RETURN TO TEST SEQUENCER
                                    0137  1123 $DS_PAGE
```

```
                                0137                .SBTTL  TEST 19:  IGNORE BYTE COUNTER MODE TEST
                         00000000                   .PSECT  TEST_019, PAGE, NOWRT
                                0020
                                0020  1126 $DS_BGNTEST      <DEFAULT,ALL>,,LONG
                                0020       DATA_019:
                         00000000  0020                   .LONG   0                   ; TEST ARGUMENT TABLE TERMINATOR
                                0024       TEST_019::
                             0000  0024                   .WORD    M<>                ; ENTRY MASK
                                0026  1127 ;++
                                0026  1128 ; TEST DESCRIPTION:
                                0026  1129 ;
                                0026  1130 ;       THIS TEST CHECKS THE IGNORE BYTE COUNTER MODE WHICH ALLOWS
                                0026  1131 ;       FOR TRANSFERS OF MORE THAN 65 KILOBYTES. THIS MODE IS
                                0026  1132 ;       CONTROLLED BY BIT 4 OF THE CONTROL REGISTER, CR<04>, (IBC).
                                0026  1133 ;       THE TEST IS COMPRISED OF 4 SUBTESTS.
                                0026  1134 ;
                                0026  1135 ;--
                                0026  1136
                                0026  1137 $DS_BGNSUB
                                0026       T19_S1::
    00000000'9F   000000E4'EF   FA  0026                   CALLG   $S$, a#DS$BGNSUB
                                0031  1138 ;
                                0031  1139 ; Subtest one checks that the BCR and VAR continue incrementing after
                                0031  1140 ; BCR overflow while in the IBC mode.
                                0031  1141 ;
                                0031  1142
                    FFCC'   30  0031  1143          BSBW    RH11TB_PRES             ; CHECK FOR RH11TB
         52   00000000'EF   D0  0034  1144          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
0000'C2   00000000'8F   C8  003B  1145 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
         56   00000000'EF   DE  0044  1146          MOVAL   MIOBUFFER,R6            ; STORE ADDRESS OF BUFFER
          57   56    00'8F   78  004B  1147          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
         57   00000000'8F   C8  0050  1148          BISL    #VALID_BIT,R7           ; SET VALID BIT IN MAP ENTRY
5A       52   00000000'8F   C1  0057  1149          ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
                  6A   57   D0  005F  1150          MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
                  0000'C2   D4  0062  1151          CLRL    VAR(R2)                 ; CLEAR VAR
              0000'C2   02  CE  0066  1152          MNEGL   #2,BCR(R2)              ; SET BCR FOR 2 BYTE TRANSFER
         50   00000000'EF   D0  006B  1153          MOVL    DRIVE0,R0               ; GET ADDRESS OF DRIVE 0
0000'C2   00000000'8F   C8  0072  1154          BISL    #INH_BYTE_CNT,CR(R2)    ; SET IBC MODE
0000'C2   00000000'8F   C8  007B  1155          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
         60   00000000'8F   D0  0084  1156          MOVL    #WRITE,(R0)             ; ISSUE W-T-D
0000'C2   00000000 8F   D0  008B  1157          MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                0094  1158          $DS_WAITUS_S    #10             ; WAIT 100 MICRO-SECONDS
                     00   DD  0094                  PUSHL   #0
                     0A   DD  0096                  PUSHL   #10
         00000000'9F      02  FB  0098                  CALLS   #2, a#DS$WAITUS
0000'C2   00000000'8F   C8  009F  1159          BISL    #SIM_EBL,DR(R2)         ; ASSERT EBL
0000'C2   00000000'8F   CA  00A8  1160          BICL    #SIM_EBL,DR(R2)         ; DEASSERT EBL
              53   0000'C2   D0  00B1  1161          MOVL    BCR(R2),R3              ; READ BCR
         53   00000000'8F   CA  00B6  1162          BICL    #BYTE3!BYTE2,R3         ; CLEAR UPPER WORD OF BCR
                  53   22   D1  00BD  1163          CMPL    #34,R3                  ; CHECK FOR CORRECT BCR
                       26   13  00C0  1164          BEQL    20$                     ; BRANCH IF BCR CORRECT
         00000000'EF      01  D0  00C2  1165          MOVL    #1,IBC_MSG              ; GENERATE ERR MSG POINTER
                                00C9  1166          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                00C9  1167                          ARRAY_MSG61,-
                                00C9  1168                          PRINT_IBCERR,-
                                00C9  1169                          #34,R3
                     53   DD  00C9                  PUSHL   R3
```

```
                    22   DD  00CB                        PUSHL   #34
          00000000'EF    DF  00CD                        PUSHAL  PRINT_IBCERR
          00000000'EF    DF  00D3                        PUSHAL  ARRAY_MSG61
          00000000'EF    DD  00D9                        PUSHL   LUN
                    01   DD  00DF                        PUSHL   #1
     00000000'9F    06   FB  00E1                        CALLS   #$$M, a#DS$ERRHARD
                             00E8  1170 20$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
  00000000'9F   FF4F CF   FA  00E8                        CALLG   10$, a#DS$CKLOOP
          54    0000'C2   D0  00F1  1171         MOVL    VAR(R2),R4                  ; READ VAR
                54    28   D1  00F6  1172         CMPL    #40,R4                      ; CHECK FOR CORRECT VAR
                    26   13  00F9  1173         BEQL    30$                         ; BRANCH IF VAR CORRECT
     00000000'EF    02   D0  00FB  1174         MOVL    #2,IBC_MSG                  ; GENERATE IBC MSG POINTER
                             0102  1175         $DS_ERRHARD_S   #2,LUN,-            ; REPORT ERROR
                             0102  1176                         ARRAY_MSG61,-
                             0102  1177                         PRINT_IBCERR,-
                             0102  1178                         #40,R4
                    54   DD  0102                        PUSHL   R4
                    28   DD  0104                        PUSHL   #40
          00000000'EF    DF  0106                        PUSHAL  PRINT_IBCERR
          00000000'EF    DF  010C                        PUSHAL  ARRAY_MSG61
          00000000'EF    DD  0112                        PUSHL   LUN
                    02   DD  0118                        PUSHL   #2
     00000000'9F    06   FB  011A                        CALLS   #$$M, a#DS$ERRHARD
                             0121  1179 30$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
  00000000'9F   FF16 CF   FA  0121                        CALLG   10$, a#DS$CKLOOP
                             012A  1180 $DS_ENDSUB
                             012A       T19_S1_X:
 00000000'9F   000000E4'EF  FA  012A                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0135  1182 $DS_BGNSUB
                                0135       T19_S2::
00000000'9F   000000F0'EF   FA  0135                 CALLG   $S$, a#DS$BGNSUB
                                0140  1183 ;
                                0140  1184 ; Subtest two checks that a PGE occurs when DT_BUSY is set and an attempt
                                0140  1185 ; is made to set IBC.
                                0140  1186 ;
                                0140  1187
      52   00000000'EF    D0  0140  1188          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
0000'C2    00000000'8F    C8  0147  1189 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
      54   00000000'8F    D0  0150  1190          MOVL    #DT_BUSY!PROGRAM_ERROR,R4 ; STORE EXPECTED RESULTS
      56   00000000'EF    DE  0157  1191          MOVAL   MIOBUFFER,R6            ; MOVE ADDRESS OF BUFFER TO R6
       57    56   00'8F    78  015E  1192          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
      57   00000000'8F    C8  0163  1193          BISL    #VALID_BIT,R7           ; SET VALID BIT IN MAP ENTRY
5A    52   00000000'8F    C1  016A  1194          ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
               6A   57    D0  0172  1195          MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
             0000'C2      D4  0175  1196          CLRL    VAR(R2)                 ; CLEAR VAR
          0000'C2   29    CE  0179  1197          MNEGL   #41,BCR(R2)             ; SET BCR FOR 41-BYTE TRANSFER
      50   00000000'EF    D0  017E  1198          MOVL    DRIVE0,R0               ; GET ADDRESS OF DRIVE 0
0000'C2    00000000'8F    C8  0185  1199          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
      60   00000000'8F    D0  018E  1200          MOVL    #WRITE,(R0)             ; ISSUE W-T-D
0C00'C2    00000000'8F    D0  0195  1201          MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                019E  1202          $DS_WAITUS_S    #10             ; WAIT 100 MICRO-SECONDS
                     00    DD  019E                PUSHL   #0
                     0A    DD  01A0                PUSHL   #10
   00000000'9F       02    FB  01A2                CALLS   #2, a#DS$WAITUS
0000'C2    00000000'8F    C8  01A9  1203          BISL    #INH_BYTE_CNT,CR(R2)    ; ATTEMPT TO SET IBC
      54   00000000'EF    C8  01B2  1204          BISL    RH11TB_MSK,R4           ; LOAD EXPECTED MASK BIT
          53   0000'C2    D0  01B9  1205          MOVL    SR(R2),R3               ; READ SR
               54   53    D1  01BE  1206          CMPL    R3,R4                   ; CHECK IF PGE IS SET
                    22    13  01C1  1207          BEQL    20$                     ; BRANCH IF PGE IS SET
   00000000'EF       03    D0  01C3  1208          MOVL    #3,IBC_MSG              ; GENERATE IBC MSG POINTER
                                01CA  1209          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                01CA  1210                          ARRAY_MSG5,-
                                01CA  1211                          PRINT_IBCERR,
          00000000'EF       DF  01CA                PUSHAL  PRINT_IBCERR
          00000000'EF       DF  01D0                PUSHAL  ARRAY_MSG5
          00000000'EF       DD  01D6                PUSHL   LUN
                     01     DD  01DC                PUSHL   #1
   00000000'9F       04    FB  01DE                CALLS   #$$M, a#DS$ERRHARD
                                01E5  1212 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
00000000'9F   FF5E CF   FA  01E5                    CALLG   10$, a#DS$CKLOOP
                                01EE  1213 $DS_ENDSUB
                                01EE       T19_S2_X:
00000000'9F   000000F0'EF   FA  01EE                CALLG   $S$, a#DS$ENDSUB
```

```
                                01F9  1215 $DS_BGNSUB
                                01F9       T19_S3::
00000000'9F   000000FC'EF   FA   01F9                 CALLG   $S$, a#DS$BGNSUB
                                0204  1216 ;
                                0204  1217 ; Subtest three checks that the BCR stops at the next BCR
                                0204  1218 ; overflow when IBC is cleared while a transfer is underway.
                                0204  1219 ;
                                0204  1220
      52   00000000'EF   D0     0204  1221          MOVL    RH_CUR_ADR,R2           ; STORE RH750 ADDRESS
 0000'C2   00000000'8F   C8     020B  1222 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
      56   00000000'EF   DE     0214  1223          MOVAL   MIOBUFFER,R6            ; STORE ADDRESS OF BUFFER
       57   56   00'8F   78     021B  1224          ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
      57   00000000'8F   C8     0220  1225          BISL    #VALID_BIT,R7           ; SET VALID BIT IN MAP ENTRY
5A    52   00000000'8F   C1     0227  1226          ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
               6A   57   D0     022F  1227          MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
           0000'C2   D4     0232  1228          CLRL    VAR(R2)                 ; CLEAR VAR
         0000'C2   29   CE     0236  1229          MNEGL   #41,BCR(R2)             ; LOAD BCR FOR 41 BYTE TRANSFER
      50   00000000'EF   D0     023B  1230          MOVL    DRIVE0,R0               ; GET ADDRESS OF DRIVE 0
 0000'C2   00000000'8F   C8     0242  1231          BISL    #INH_BYTE_CNT,CR(R2)    ; SET IBC
 0000'C2   00000000'8F   C8     024B  1232          BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
      60   00000000'8F   D0     0254  1233          MOVL    #WRITE,(R0)             ; ISSUE W-T-D
 0000'C2   00000000'8F   D0     025B  1234          MOVL    #SIM_OCC,DR(R2)         ; SET OCCUPIED
                                0264  1235          $DS_WAITUS_S    #10             ; WAIT 100 MICRO-SECONDS
                     00   DD     0264                PUSHL   #0
                     0A   DD     0266                PUSHL   #10
      00000000'9F    02   FB     0268                CALLS   #2, a#DS$WAITUS
 0000'C2   00000000'8F   CA     026F  1236          BICL    #INH_BYTE_CNT,CR(R2)    ; CLEAR IBC
                     53   D4     0278  1237          CLRL    R3                      ; CLEAR SYNC CLOCK COUNTER
 0000'C2   00000000'8F   C8     027A  1238 20$:     BISL    #SIM_SCLK,DR(R2)        ; ASSERT SCLK
 0000'C2   00000000'8F   CA     0283  1239          BICL    #SIM_SCLK,DR(R2)        ; DEASSERT SCLK
          EA 53    15   F3     028C  1240          AOBLEQ  #21,R3,20$              ; ASSERT SCLK 22 TIMES
         54   0000'C2   D0     0290  1241          MOVL    BCR(R2),R4              ; READ BCR
              54   00   D1     0295  1242          CMPL    #0,R4                   ; CHECK IF BCR IS ZERO
                   26   13     0298  1243          BEQL    30$                     ; BRANCH IF BCR IS ZERO
      00000000'EF   04   D0     029A  1244          MOVL    #4,IBC_MSG              ; GENERATE IBC MSG POINTER
                                02A1  1245          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                02A1  1246                          ARRAY_MSG61,-
                                02A1  1247                          PRINT_IBCERR,-
                                02A1  1248                          #0,R4
                     54   DD     02A1                PUSHL   R4
                     00   DD     02A3                PUSHL   #0
           00000000'EF    DF     02A5                PUSHAL  PRINT_IBCERR
           00000000'EF    DF     02AB                PUSHAL  ARRAY_MSG61
           00000000'EF    DD     02B1                PUSHL   LUN
                     01   DD     02B7                PUSHL   #1
      00000000'9F    06   FB     02B9                CALLS   #$$M, a#DS$ERRHARD
                                02C0  1249 30$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
00000000'9F   FF47   CF     FA   02C0                CALLG   10$, a#DS$CKLOOP
                                02C9  1250 $DS_ENDSUB
                                02C9       T19_S3_X:
00000000'9F   000000FC'EF   FA   02C9                 CALLG   $S$, a#DS$ENDSUB
```

```
                                02D4  1252 $DS_BGNSUB
                                02D4       T19_S4::
    00000000'9F   00000108'EF   FA 02D4                    CALLG   $S$, a*DS$BGNSUB
                                02DF  1253 ;
                                02DF  1254 ; Subtest four checks that writing to a Map Register in IBC mode
                                02DF  1255 ; clears the interrupt condition, and that a PGE is not caused as
                                02DF  1256 ; a result of writing to a Map Register while a transfer is in
                                02DF  1257 ; progress.
                                02DF  1258 ;
                                02DF  1259
          52   00000000'EF    D0 02DF  1260 10$:    MOVL    RH_CUR_ADR,R2           ; STORE ADDRESS OF RH750
                         55     D4 02E6  1261         CLRL    R5                      ; CLEAR DELAY COUNTER
                         59     D4 02E8  1262         CLRL    R9                      ; CLEAR INTR DELAY COUNTER
      0000'C2   00000000'8F     C8 02EA  1263         BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
          56   00000000'EF     DE 02F3  1264         MOVAL   DATA_IBUFFER,R6         ; STORE ADDRESS OF BUFFER
           57    56    00'8F    78 02FA  1265         ASHL    #-PF_FIELD,R6,R7        ; EXTRACT PFN
          57   00000000'8F     C8 02FF  1266         BISL    #VALID_BIT,R7           ; SET VALID BIT IN MAP ENTRY
      5A  52   00000000'8F     C1 0306  1267         ADDL3   #MAP_OFFSET,R2,R10      ; R10 POINTS TO MAP REG 0
                     6A   57    D0 030E  1268         MOVL    R7,(R10)                ; WRITE PFN INTO MAP REG 0
                58   6A   01    C1 0311  1269         ADDL3   #1,(R10),R8             ; STORE EXP RESULTS
          00000000'EF   01     88 0315  1270         BISB2   #1,EXP_INTERRUPT        ; SET EXPECTED INT FLAG
                00000000'EF     94 031C  1271         CLRB    INT_OCCURRED            ; CLEAR INTERRUPT OCCURRED FLAG
      0000'C2   00000000'8F     C8 0322  1272         BISL    #MAINT_MODE,CR(R2)      ; SET MAINT MODE
      0000'C2   00000000'8F     C8 032B  1273         BISL    #INH_BYTE_CNT,CR(R2)    ; SET IBC
              0000'C2   02      CE 0334  1274         MNEGL   #2,BCR(R2)              ; LOAD BCR FOR 2 BYTE TRANSFER
                   0000'C2      D4 0339  1275         CLRL    VAR(R2)                 ; CLEAR VAR
      0000'C2   00000000'8F     C8 033D  1276         BISL    #INTERRUPT_EN,CR(R2)    ; SET IE
          50   00000000'EF     D0 0346  1277         MOVL    DRIVE0,R0               ; GET ADDRESS OF DRIVE 0
          60   00000000'8F     D0 034D  1278         MOVL    #WRITE,(R0)             ; EXECUTE W-T-D
      0000'C2   00000000'8F     C8 0354  1279         BISL    #SIM_OCC,DR(R2)         ; ASSERT OCCUPIED
          00000000'8F   00     DA 035D  1280         MTPR    #0,#IPLR                ; LOWER IPL TO ALLOW INTERRRUPT
               FC 55    0A     F3 0364  1281 15$:    AOBLEQ  #10,R5,15$              ; WAIT FOR INTERRUPT
        22 00000000'EF   00     E4 0368  1282         BBSC    #0,INT_OCCURRED,20$     ; CHECK IF EXP INTERRUPT OCCURRED
          00000000'EF   05     D0 0370  1283         MOVL    #5,IBC_MSG              ; GENERATE EBC MSG POINTER
                                0377  1284         $DS_ERRHARD_S   #1,LUN,-         ; REPORT ERROR
                                0377  1285                         ARRAY_MSG3,-
                                0377  1286                         PRINT_IBCERR
                00000000'EF     DF 0377                    PUSHAL  PRINT_IBCERR
                00000000'EF     DF 037D                    PUSHAL  ARRAY_MSG3
                00000000'EF     DD 0383                    PUSHL   LUN
                         01     DD 0389                    PUSHL   #1
          00000000'9F   04     FB 038B                    CALLS   #$$M, a*DS$ERRHARD
                                0392  1287 20$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF49  CF    FA 0392                    CALLG   10$, a*DS$CKLOOP
           22 0000'C2   1F     E0 039B  1288         BBS     #31,SR(R2),30$          ; CHECK THAT DT_BUSY IS SET
          00000000'EF   06     D0 03A1  1289         MOVL    #6,IBC_MSG              ; GENERATE IBC MSG POINTER
                                03A8  1290         $DS_ERRHARD_S   #2,LUN,-         ; REPORT ERROR
                                03A8  1291                         ARRAY_MSG61,-
                                03A8  1292                         PRINT_IBCERR
                00000000'EF     DF 03A8                    PUSHAL  PRINT_IBCERR
                00000000'EF     DF 03AE                    PUSHAL  ARRAY_MSG61
                00000000'EF     DD 03B4                    PUSHL   LUN
                         02     DD 03BA                    PUSHL   #2
          00000000'9F   04     FB 03BC                    CALLS   #$$M, a*DS$ERRHARD
                                03C3  1293 30$:    $DS_CKLOOP      10$             ; SCOPE LOOP?
    00000000'9F   FF18  CF    FA 03C3                    CALLG   10$, a*DS$CKLOOP
                         6A    D6 03CC  1294         INCL    (R10)                   ; WRITE TO MAP REGISTER
```

```
                00000000'EF    01    88  03CE  1295          BISB2   #1,EXP_INTERRUPT          ; SET EXPECTED INTERRUPT FLAG
                     00000000'EF    94  03D5  1296          CLRB    INT_OCCURRED              ; CLEAR INTERRUPT OCCURRED FLAG
           0000'C2   00000000'8F    C8  03DB  1297          BISL    #INTERRUPT_EN,CR(R2)      ; SET INTERRUPT ENABLE
                      FC 59    0A    F3  03E4  1298 35$:     AOBLEQ  #10,R9,35$                ; WAIT, IN CASE OF INTERRUPT
             22 00000000'EF    00    E1  03E8  1299          BBC     #0,INT_OCCURRED,40$       ; INTERRUPT SHOULD NOT HAVE OCCURRED
                00000000'EF    07    D0  03F0  1300          MOVL    #7,IBC_MSG                ; GENERATE IBC MSG POINTER
                                         03F7  1301          $DS_ERRHARD_S   #3,LUN,-          ; REPORT ERROR
                                         03F7  1302                          ARRAY_MSG55,-
                                         03F7  1303                          PRINT_IBCERR
                     00000000'EF    DF  03F7                        PUSHAL  PRINT_IBCERR
                     00000000'EF    DF  03FD                        PUSHAL  ARRAY_MSG55
                     00000000'EF    DD  0403                        PUSHL   LUN
                                 03    DD  0409                        PUSHL   #3
                00000000'9F    04    FB  040B                        CALLS   #$$M, a#DS$ERRHARD
                                         0412  1304 40$:     $DS_CKLOOP      10$               ; SCOPE LOOP?
           00000000'9F    FEC9 CF    FA  0412                        CALLG   10$, a#DS$CKLOOP
                 22 0000'C2    13    E1  041B  1305          BBC     #19,SR(R2),50$            ; CHECK THAT PGE DID NOT OCCUR
                00000000'EF    08    D0  0421  1306          MOVL    #8,IBC_MSG                ; GENERATE IBC MSG POINTER
                                         0428  1307          $DS_ERRHARD_S   #4,LUN,-          ; REPORT ERROR
                                         0428  1308                          ARRAY_MSG5,-
                                         0428  1309                          PRINT_IBCERR
                     00000000'EF    DF  0428                        PUSHAL  PRINT_IBCERR
                     00000000'EF    DF  042E                        PUSHAL  ARRAY_MSG5
                     00000000'EF    DD  0434                        PUSHL   LUN
                                 04    DD  043A                        PUSHL   #4
                00000000'9F    04    FB  043C                        CALLS   #$$M, a#DS$ERRHARD
                                         0443  1310 50$:     $DS_CKLOOP      10$               ; SCOPE LOOP?
           00000000'9F    FE98 CF    FA  0443                        CALLG   10$, a#DS$CKLOOP
                      6A    58    D1  044C  1311          CMPL    R8,(R10)                  ; CHECK THAT MAP REGISTER WAS WRITTEN
                            26    13  044F  1312          BEQL    60$                       ; BRANCH IF MAP REG WAS WRITTEN
                00000000'EF    09    D0  0451  1313          MOVL    #9,IBC_MSG                ; GENERATE IBC MSG POINTER
                                         0458  1314          $DS_ERRHARD_S   #5,LUN,-          ; REPORT ERROR
                                         0458  1315                          ARRAY_MSG5,-
                                         0458  1316                          PRINT_IBCERR,-
                                         0458  1317                          R8,(R10)
                                      6A DD  0458                        PUSHL   (R10)
                                      58 DD  045A                        PUSHL   R8
                          00000000'EF    DF  045C                        PUSHAL  PRINT_IBCERR
                          00000000'EF    DF  0462                        PUSHAL  ARRAY_MSG5
                          00000000'EF    DD  0468                        PUSHL   LUN
                                      05 DD  046E                        PUSHL   #5
                00000000'9F    06    FB  0470                        CALLS   #$$M, a#DS$ERRHARD
                                         0477  1318 60$:     $DS_CKLOOP      10$               ; SCOPE LOOP?
           00000000'9F    FE64 CF    FA  0477                        CALLG   10$, a#DS$CKLOOP
                                         0480  1319 $DS_ENDSUB
                                         0480       T19_S4_X:
      00000000'9F    00000108'EF    FA  0480                        CALLG   $S$, a#DS$ENDSUB
                                         048B  1320 $DS_ENDTEST
                      50    01    D0  048B                        MOVL    #1, R0            ; NORMAL EXIT
                                         048E       TEST_019_X::
                00000000'9F    6E    FA  048E                        CALLG   (SP), a#DS$BREAK
                                      04 0495                        RET                       ; RETURN TO TEST SEQUENCER
                                         0496  1321 $DS_PAGE
```

```
                                  0496                .SBTTL  TEST 20:  RH11-TB TESTS
                             00000000                        .PSECT  TEST_020, PAGE, NOWRT
                                  0010
                                  0010  1324 $DS_BGNTEST      <DEFAULT,ALL,MBE>,,LONG
                                  0010       DATA_020:
                        00000000  0010                        .LONG   0                ; TEST ARGUMENT TABLE TERMINATOR
                                  0014       TEST_020::
                            0000  0014                        .WORD    M<>             ; ENTRY MASK
                                  0016  1325 ;++
                                  0016  1326 ; TEST DESCRIPTION:
                                  0016  1327 ;
                                  0016  1328 ;       IN THIS TEST THE PROGRAM WILL LOOK FOR A MASSBUS EXERCISER.
                                  0016  1329 ;       IF THERE IS AN MBE PRESENT ON THE MASSBUS OF THE RH750 UNDER
                                  0016  1330 ;       TEST, THEN THE PROGRAM WILL CHECK TO SEE IF THE MBE IS
                                  0016  1331 ;       INITIALIZED PROPERLY.  FOLLOWING THIS THE PROGRAM WILL CHECK
                                  0016  1332 ;       THE INTEGRITY OF ALL MASSBUS SIGNAL LINES (TO PROVIDE CABLE
                                  0016  1333 ;       TESTING), AND FINALLY THE PROGRAM WILL PERFORM A SERIES OF
                                  0016  1334 ;       HIGH SPEED DATA TRANSFERS.
                                  0016  1335 ;
                                  0016  1336 ; TEST ALGORITHM:
                                  0016  1337 ;
                                  0016  1338 ;       THE PROGRAM ASSERTS MASSBUS INITIALIZE. IF THERE IS AN MBE ON
                                  0016  1339 ;       THE MASSBUS, IT WILL BECOME DRIVE 7 ON THE TRAILING EDGE OF
                                  0016  1340 ;       MASS INIT. DRIVE 7'S DRIVE TYPE REGISTER IS CHECKED TO SEE IF
                                  0016  1341 ;       IT IS  HEX 20, IDENTIFYING IT AS AN MBE. AT THIS POINT, IF THERE
                                  0016  1342 ;       IS NO MBE ON THE MASSBUS OF THE RH750 UNDER TEST, THE PROGRAM
                                  0016  1343 ;       EXITS TO THE CEP SUPERVISOR.  IF THE MBE IS PRESENT, THE PROGRAM
                                  0016  1344 ;       PROCEEDS, CHECKING FOR THE PRESENCE OF DRIVE AVAILABLE, MEDIUM
                                  0016  1345 ;       ON LINE, DRIVE PRESENT, AND DRIVE READY.  IT FURTHER CHECKS THAT
                                  0016  1346 ;       ATTENTION, COMPOSITE ERROR, AND POSITIONING IN PROGRESS ARE NOT
                                  0016  1347 ;       ACTIVE.
                                  0016  1348 ;
                                  0016  1349 ;       THE ALGORTITHMS OF THE VARIOUS SUBTESTS ARE LISTED WITHIN THE
                                  0016  1350 ;       SUBTESTS.
                                  0016  1351 ;--
                                  0016  1352 $DS_BGNSUB
                                  0016       T20_S1::
00000000'9F   00000114'EF    FA   0016                        CALLG   $S$, a#DS$BGNSUB
         52   00000000'EF    D0   0021  1353          MOVL    RH_CUR_ADR,R2            ; GET ADDRESS OF MUT
    0000'C2   00000000'8F    C8   0028  1354 5$:      BISL    #PGM_INIT,CR(R2)         ; INIT MUT
         58   00000000'EF    D0   0031  1355          MOVL    MBE_CUR_ADR,R8           ; IS MBE SELECTED FOR THIS RH?
                        47    13   0038  1356          BEQL    7$                       ; BRANCH IF NO
         54   00000000'8F    D0   003A  1357          MOVL    #TYPE_MBE,R4             ; EXPECT MBE IDENTIFIER
              50   0000'C8    D0   0041  1358          MOVL    MBE_DTR(R8),R0           ; READ MBE DRIVE TYPE REGISTER
                   53   50    3C   0046  1359          MOVZWL  R0,R3                    ; EXTRACT CONTENTS OF REGISTER
                   54   53    D1   0049  1360          CMPL    R3,R4                    ; IS DRIVE 7 AN MBE?
                        5C    13   004C  1361          BEQL    10$                      ; SKIP IF IT IS
         0000FE54'EF    01    D1   004E  1362          CMPL    #1,DSA$GL_PASSNO         ; PASS 0 OR 1?
                        0D    18   0055  1363          BGEQ    6$                       ; BRANCH IF YES
0000FE00'EF   00000400 8F    D3   0057  1364          BITL    #DSA$M_TRACE,DSA$GL_FLAGS ; TRACE FLAG SET?
                        40    13   0062  1365          BEQL    8$                       ; BRANCH IF NO
                                  0064  1366 6$:      $DS_ERRHARD_S   UNIT=LUN,-                                  ;DPM001
                                  0064  1367                          MSGADR=NO_MBE_MSG,-                         ;DPM001
                                  0064  1368                          PRLINK=PRINT_NULL                           ;DPM001
              00000000'EF    DF   0064                        PUSHAL  PRINT_NULL
              00000000'EF    DF   006A                        PUSHAL  NO_MBE_MSG
              00000000'EF    DD   0070                        PUSHL   LUN
```

```
                        01   DD  0076                        PUSHL   #$ER
                                 0078        ;;; TEST 20, SUBTEST 1, ERROR 1
          00000000'9F   04   FB  0078                        CALLS   #$$M, @#DS$ERRHARD
                        23   11  007F  1369            BRB     8$                      ; EXIT TEST
          0000FE54'EF   01   D1  0081  1370 7$:       CMPL    #1,DSA$GL_PASSNO        ; PASS 0 OR 1?
                        0D   18  0088  1371            BGEQ    9$                      ; BRANCH IF YES
    0000FE00'EF   00000400 8F  D3  008A  1372            BITL    #DSA$M_TRACE,DSA$GL_FLAGS ; TRACE FLAG SET?
                        0D   13  0095  1373            BEQL    8$                      ; BRANCH IF NO
                                 0097  1374 9$:       $DS_PRINTF_S    FMT_NO_MBE_SEL  ; INFORM OPERATOR NO MBE SELECTED
              00000000'EF   9F  0097                        PUSHAB  FMT_NO_MBE_SEL
          00000000'9F   01   FB  009D                        CALLS   #$$N, @#DS$PRINTF
                                 00A4  1375 8$:       $DS_EXIT        TEST            ; ABORT TEST
                   50   01   D0  00A4                        MOVL    #1,R0           ; NOT AN ERROR
                      1EED  31  00A7                        BRW     TEST_020_X      ; EXIT TEST 20
                                 00AA  1376 ;
                                 00AA  1377 ; MBE IS SELECTED AND IS PHYSICALLY PRESENT. CONTINUE WITH TEST
                                 00AA  1378 ;
    00000000'EF   00000000'8F  C8  00AA  1379 10$:      BISL    #ATTENTION,RH11TB_MSK   ; SET EXPECTED BIT
        0000'C2   00000000'8F  C8  00B5  1380           BISL    #PGM_INIT,CR(R2)        ; INIT MBE
                                 00BE  1381           ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                 00BE  1382                   FMT_MBE_CR1,SA0_MSG
    00000000'EF   00000000'EF  DE  00BE                        MOVAL   MBE_REG_MSG,REG_NAME
              00000000'EF   0C  9A  00C9                        MOVZBL  #12,REG_NO
    00000000'EF   00000000'EF  DE  00D0                        MOVAL   FMT_MBE_CR1,REG_STRING
              54  00000000'8F  D0  00DB  1383           MOVL    #DVA,R4                 ; EXPECT DVA SET
                  50    0000'C8  D0  00E2  1384           MOVL    MBE_CR1(R8),R0          ; READ MBE CONTROL REGISTER 1
                        53   50  3C  00E7  1385           MOVZWL  R0,R3                   ; EXTRACT CONTENTS OF REGISTER
                        54   53  D1  00EA  1386           CMPL    R3,R4                   ; DVA SET?
                             21  13  00ED  1387           BEQL    20$                     ; SKIP IF IT IS
                        55   54  D0  00EF  1388           MOVL    R4,R5                   ; REPORT BIT FAILURE
             56   00000000'8F  D0  00F2  1389           MOVL    #MBE_CR1,R6             ; REPORT FAILING REGISTER
                                 00F9  1390           $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                 00F9  1391                   PRINT_SBE
                   00000000'EF   DF  00F9                        PUSHAL  PRINT_SBE
                              00   DD  00FF                        PUSHL   #0
                   00000000'EF   DD  0101                        PUSHL   LUN
                              01   DD  0107                        PUSHL   #1
              00000000'9F   04   FB  0109                        CALLS   #$$M, @#DS$ERRHARD
                                 0110  1392 20$:      $DS_CKLOOP      10$             ; SCOPE LOOP?
           00000000'9F   97 AF   FA  0110                        CALLG   10$, @#DS$CKLOOP
        0000'C2   00000000'8F  C8  0118  1393 25$:      BISL    #PGM_INIT,CR(R2)        ; INIT THE MBE
                                 0121  1394           ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                 0121  1395                   FMT_MBE_SR,SA1_MSG
    00000000'EF   00000000'EF  DE  0121                        MOVAL   MBE_REG_MSG,REG_NAME
              00000000'EF   0C  9A  012C                        MOVZBL  #12,REG_NO
    00000000'EF   00000000'EF  DE  0133                        MOVAL   FMT_MBE_SR,REG_STRING
              54  00000000'8F  D0  013E  1396           MOVL    #DRV_ERRMASK,R4         ; EXPECT NO ATA, ERR, OR PIP
                  50    0000'C8  D0  0145  1397           MOVL    MBE_SR(R8),R0           ; READ STATUS REGISTER
                        53   50  3C  014A  1398           MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
                        53   54  D3  014D  1399           BITL    R4,R3                   ; ANY ATA, ERR, OR PIP?
                             21  13  0150  1400           BEQL    30$                     ; SKIP IF NONE ARE SET
                        55   53  D0  0152  1401           MOVL    R3,R5                   ; REPORT BIT FAILURE
             56   00000000'8F  D0  0155  1402           MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                                 015C  1403           $DS_ERRHARD_S   #2,LUN,,-       ; PRINT ERROR
                                 015C  1404                   PRINT_SBE
                   00000000'EF   DF  015C                        PUSHAL  PRINT_SBE
                              00   DD  0162                        PUSHL   #0
```

```
                00000000'EF   DD  0164                      PUSHL   LUN
                          02  DD  016A                      PUSHL   #2
            00000000'9F   04  FB  016C                      CALLS   #$$M, a#DS$ERRHARD
                              0173  1405 30$:    $DS_CKLOOP      25$                     ; SCOPE LOOP?
          00000000'9F    A2 AF  FA  0173                      CALLG   25$, a#DS$CKLOOP
             53   00000000'8F  CA  017B  1406 35$:    BICL    #LOWBITS_MSK,R3         ; MASK LOWER BITS OF STATUS REGISTER
             54   00000000'8F  D0  0182  1407         MOVL    #DRV_INITMASK,R4        ; EXPECT MOL, DPR, AND DRY
                  54   53      D1  0189  1408         CMPL    R3,R4                   ; ALL SET?
                          1A   13  018C  1409         BEQL    40$                     ; SKIP IF ALL ARE SET
                  55   54      D0  018E  1410         MOVL    R4,R5                   ; REPORT BIT FAILURE
                               0191  1411         $DS_ERRHARD_S   #3,LUN,,-        ; PRINT ERROR
                               0191  1412                         PRINT_SBE
                00000000'EF   DF  0191                      PUSHAL  PRINT_SBE
                          00  DD  0197                      PUSHL   #0
                00000000'EF   DD  0199                      PUSHL   LUN
                          03  DD  019F                      PUSHL   #3
            00000000'9F   04  FB  01A1                      CALLS   #$$M, a#DS$ERRHARD
                              01A8  1413 40$:    $DS_CKLOOP      25$                     ; SCOPE LOOP?
        00000000'9F    FF6C CF  FA  01A8                      CALLG   25$, a#DS$CKLOOP
                              01B1  1414 $DS_ENDSUB
                              01B1       T20_S1_X:
    00000000'9F    00000114'EF  FA  01B1                      CALLG   $S$, a#DS$ENDSUB
```

```
                                01BC  1416 $DS_BGNSUB
                                01BC       T20_S2::
00000000'9F   00000120'EF   FA  01BC                CALLG   $S$, a#DS$BGNSUb
                                01C7  1417 ;++
                                01C7  1418 ; TEST MASSBUS DRIVE SELECT LINES
                                01C7  1419 ;
                                01C7  1420 ; TEST ALGORITHM:
                                01C7  1421 ;
                                01C7  1422 ;       IF OTHER DRIVES ARE PRESENT, DON'T PERFORM THIS SUBTEST. THIS
                                01C7  1423 ;       MUST BE DONE BECAUSE RH11TB DOES NOT PERFORM ACCORDING TO SPEC.
                                01C7  1424 ;
                                01C7  1425 ;       SELECT DRIVE N (WHERE N = 0 -> 7), EXAMINE RH750 STATUS REGISTER
                                01C7  1426 ;       FOR NON-EXISTANT DRIVE.  IF DRIVE DOES NOT EXIST, SELECT MBE
                                01C7  1427 ;       AS THAT DRIVE, CLEAR NED, WRITE REGISTER 0 OF THAT DRIVE, AND
                                01C7  1428 ;       CHECK MBE ATTENTION SUMMARY REGISTER TO SEE IF THE CORRECT
                                01C7  1429 ;       VALUE APPEARS IN THE DRIVE SELECT FIELD.
                                01C7  1430 ;--
            00000000'EF   16    01C7  1431          JSB     DRIVES_PRESENT          ; SEE IF ANY OTHER DRIVES PRESENT
             03 50    00   E0   01CD  1432          BBS     #0,R0,5$                ; EXIT SUB TEST IS OTHER DRIVES HERE
                   00B9   31    01D1  1433          BRW     50$                     ; SKIP SUB TEST
                                01D4  1434 5$:      ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO HANDLE ERROR
                                01D4  1435                  FMT_MBE_ASR
00000000'EF   00000000'EF   DE  01D4                MOVAL   MBE_REG_MSG,REG_NAME
         00000000'EF    0C  9A  01DF                MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  01E6                MOVAL   FMT_MBE_ASR,REG_STRING
         52   00000000'EF   D0  01F1  1436          MOVL    RH_CUR_ADR,R2           ; GET MUT
     0000'C2   00000000'8F   C8  01F8  1437 10$:     BISL    #PGM_INIT,CR(R2)        ; INIT MUT
         58   00000000'EF   D0  0201  1438          MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
                       5B   D4  0208  1439          CLRL    R11                     ; CLEAR N
      5A    00000000'EF   D0  020A  1440          MOVL    DRIVE0,R10              ; INTIALIZE DRIVE NUMBER
            50    6A   D0  0211  1441 20$:      MOVL    (R10),R0                ; READ DRIVE N'S REGISTER 0
          50    0000'C2   D0  0214  1442          MOVL    SR(R2),R0               ; CHECK NED IN RH750 STATUS REG
            0E 50    12   E1  0219  1443          BBC     #18,R0,30$              ; BRANCH IF DRIVE EXISTS
0000'C2    00000000'8F   C8  021D  1444          BISL    #PGM_INIT,CR(R2)        ; INIT MUT TO REASSIGN MBE AS 7
          0000'C8    5B   D0  0226  1445          MOVL    R11,MBE_CR2(R8)         ; SELECT MBE AS DRIVE N
            6A    00   D0  022B  1446 30$:      MOVL    #0,(R10)                ; WRITE REG 0 OF DRIVE N
  F8 50    00000064 8F   F3  022E  1447 31$:      AOBLEQ  #100,R0,31$             ; WAIT A BIT
          53    0000'CA   D0  0236  1448          MOVL    MBE_ASR(R10),R3         ; READ ATTENTION SUMMARY REG
      53    00000000'8F   CA  023B  1449          BICL    #DS_MSK,R3              ; MASK ALL BUT DRIVE SEL FIELD
            54    5B   D0  0242  1450          MOVL    R11,R4                  ; SET UP FOR EXPECTED DS VALUE
          54    54    08  9C  0245  1451          ROTL    #8,R4,R4                ; CALCULATE EXPECTED DS VALUE
            54    53   D1  0249  1452          CMPL    R3,R4                   ; EXPECTED = RECEIVED?
                  21   13  024C  1453          BEQL    40$                     ; BRANCH IF SO
            55    53   D0  024E  1454          MOVL    R3,R5                   ; REPORT FAILING BITS
      56    00000000'8F   D0  0251  1455          MOVL    #MBE_ASR,R6             ; REPORT FAILING REGISTER
                              0258  1456          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                              0258  1457                          PRINT_SBE
            00000000'EF   DF  0258                PUSHAL  PRINT_SBE
                       00   DD  025E                PUSHL   #0
            00000000'EF   DD  0260                PUSHL   LUN
                       01   DD  0266                PUSHL   #1
      00000000'9F    04   FB  0268                CALLS   #$$M, a#DS$ERRHARD
                              026F  1458 40$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
  00000000'9F    86 AF   FA  026F                CALLG   10$, a#DS$CKLOOP
0000'C2    00000000'8F   C8  0277  1459          BISL    #NON_XIST_DRIVE,SR(R2)  ; CLEAR NED
      5A   00000000'8F   C0  0280  1460          ADDL    #DRIVE_OFFSET,R10       ; SELECT NEXT DRIVE
     FF84 5B    01    07   F1  0287  1461          ACBL    #7,#1,R11,20$           ; DO FOR ALL DRIVE NUMBERS
```

```
                        028D  1462 50$:
                        028D  1463 $DS_ENDSUB
                        028D       T20_S2_X:
00000000'9F   00000120'EF   FA  028D                    CALLG   $S$, a#DS$ENDSUB
```

```
                                0298  1465 $DS_BGNSUB
                                0298       T20_S3::
00000000'9F   0000012C'EF   FA   0298                 CALLG   $S$, a#DS$BGNSUB
                                02A3  1466 ;++
                                02A3  1467 ; TEST MASSBUS REGISTER SELECT LINES
                                02A3  1468 ;
                                02A3  1469 ; TEST ALGORITHM:
                                02A3  1470 ;
                                02A3  1471 ;       SELECT ALL DRIVE REGISTERS STARTING WITH DRIVE REGISTER 0,
                                02A3  1472 ;       INCREMENTING THROUGH TO THE LAST.  EACH TIME THE REGISTER SELECT
                                02A3  1473 ;       FIELD OF THE ATTENTION SUMMARY REGISTER WILL BE CHECKED TO SEE
                                02A3  1474 ;       IF THE RS LINES WERE PROPERLY WRITTEN.
                                02A3  1475 ;--
        52   00000000'EF    D0   02A3  1476           MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
                                02AA  1477           ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                02AA  1478                   FMT_MBE_ASR
00000000'EF   00000000'EF   DE   02AA                        MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF   0C    9A   02B5                        MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   02BC                        MOVAL   FMT_MBE_ASR,REG_STRING
        58   00000000'EF    D0   02C7  1479           MOVL    MBE_CUR_ADR,R8          ; GET ADDRESS OF MBE
                    59     D4   02CE  1480           CLRL    R9                      ; INITIALIZE INDEX
    0000'C2   00000000'8F   C8   02D0  1481 20$:      BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
              54    59      D0   02D9  1482           MOVL    R9,R4                   ; SET UP EXPECTED RESULTS
            6849   6849     D0   02DC  1483           MOVL    (R8) R9 ,(R8) R9        ; SELECT REGISTER R9
      F8 50   00000064 8F    F3   02E1  1484 21$:      AOBLEQ  #100,R0,21$             ; WAIT A BIT
           50    0000'C8     D0   02E9  1485           MOVL    MBE_ASR(R8),R0          ; READ MBE ASR FOR RS LINES
                 53    50    3C   02EE  1486           MOVZWL  R0,R3                   ; EXTRACT CONTENTS OF REGISTER
           54    54    0B    9C   02F1  1487           ROTL    #11,R4,R4               ; ADJUST FOR RS FIELD
        53   00000000'8F    CA   02F5  1488           BICL    #RS_MSK,R3              ; ISOLATE REGISTER SELECT FIELD
                 54    53    D1   02FC  1489           CMPL    R3,R4                   ; IS THIS THE EXPECTED REGISTER?
                       20    13   02FF  1490           BEQL    30$                     ; BRANCH IF IT IS
                       55    D4   0301  1491           CLRL    R5
      56   00000000'8F    D0   0303  1492           MOVL    #MBE_ASR,R6             ; REPORT FAILING REGISTER
                                030A  1493           $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                030A  1494                   PRINT_SBE
                00000000'EF   DF   030A                      PUSHAL  PRINT_SBE
                              00   DD   0310                 PUSHL   #0
                00000000'EF   DD   0312                      PUSHL   LUN
                              01   DD   0318                 PUSHL   #1
      00000000'9F   04    FB   031A                        CALLS   #$$M, a#DS$ERRHARD
                                0321  1495 30$:      $DS_CKLOOP      20$             ; SCOPE LOOP?
   00000000'9F   AC AF    FA   0321                        CALLG   20$, a#DS$CKLOOP
            A3 59   1F    F3   0329  1496           AOBLEQ  #31,R9,20$              ; DONE FOR ALL REGISTERS?
                                032D  1497 $DS_ENDSUB
                                032D       T20_S3_X:
00000000'9F   0000012C'EF   FA   032D                 CALLG   $S$, a#DS$ENDSUB
```

```
                                0338  1499 $DS_BGNSUB
                                0338       T20_S4::
00000000'9F   00000138'EF   FA  0338                 CALLG   $S$, a#DS$BGNSUB
                                0343  1500 ;++
                                0343  1501 ; TEST MASSBUS ATTENTION LINE
                                0343  1502 ;
                                0343  1503 ; TEST ALGORITHM:
                                0343  1504 ;
                                0343  1505 ;       A SEARCH COMMAND IS ISSUED TO THE MBE, WHICH WILL RESULT IN
                                0343  1506 ;       THE MBE ASSERTING ATTENTION IN APPROXIMATELY 100 MICROSECONDS.
                                0343  1507 ;       ATTENTION IS CHECKED IN ITS ASSERTED STATE.  THE NEGATION OF
                                0343  1508 ;       ATTENTION HAS ALREADY BEEN CHECKED IN THE MBE INITIALIZATION.
                                0343  1509 ;--
        52   00000000'EF    D0  0343  1510          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
                                034A  1511          ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                034A  1512                  FMT_MBE_SR
00000000'EF   00000000'EF   DE  034A                        MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF   0C    9A  0355                        MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  035C                        MOVAL   FMT_MBE_SR,REG_STRING
        58   00000000'EF    D0  0367  1513          MOVL    MBE_CUR_ADR,R8          ; GET ADDRESS OF MBE
        54   00008000 8F    C8  036E  1514          BISL    #BIT15,R4               ; EXPECT ATTENTION BIT
    0000'C2   00000000'8F    C8  0375  1515 10$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
    0000'C8   00000000'8F    D0  037E  1516          MOVL    #SEARCH,MBE_CR1(R8)     ; INITIATE SEARCH COMMAND
                                0387  1517          $DS_WAITUS_S    #50             ; WAIT 500 MICROSECONDS
                     00     DD  0387                        PUSHL   #0
                     32     DD  0389                        PUSHL   #50
        00000000'9F    02    FB  038B                        CALLS   #2, a#DS$WAITUS
            50   0000'C8    D0  0392  1518          MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
                 53    50   3C  0397  1519          MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
                 54    53   D3  039A  1520          BITL    R3,R4                   ; ATA SET?
                       21   12  039D  1521          BNEQ    20$                     ; BRANCH IF IT IS
                 55    54   D0  039F  1522          MOVL    R4,R5                   ; REPORT BIT FAILURE
        56   00000000'8F    D0  03A2  1523          MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                                03A9  1524          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                03A9  1525                          PRINT_SBE
             00000000'EF    DF  03A9                        PUSHAL  PRINT_SBE
                     00     DD  03AF                        PUSHL   #0
             00000000'EF    DD  03B1                        PUSHL   LUN
                     01     DD  03B7                        PUSHL   #1
        00000000'9F    04    FB  03B9                        CALLS   #$$M, a#DS$ERRHARD
                                03C0  1526 20$:     $DS_CKLOOP      10$             ; SCOPE LOOP?
  00000000'9F    B2 AF    FA  03C0                        CALLG   10$, a#DS$CKLOOP
0000'C2   00000000'8F    C8  03C8  1527          BISL    #PGM_INIT,CR(R2)        ; CLEAR ATTENTION
                                03D1  1528 $DS_ENDSUB
                                03D1       T20_S4_X:
00000000'9F   00000138'EF   FA  03D1                 CALLG   $S$, a#DS$ENDSUB
```

```
                                    03DC  1530 $DS_BGNSUB
                                    03DC       T20_S5::
00000000'9F   00000144'EF   FA   03DC                    CALLG   $S$, a*DS$BGNSUB
                                    03E7  1531 ;++
                                    03E7  1532 ; TEST MASSBUS CONTROL LINES
                                    03E7  1533 ;
                                    03E7  1534 ; TEST ALGORITHM:
                                    03E7  1535 ;
                                    03E7  1536 ;       WRITE ALL ONES, ALL ZEROES, FLOATING ONE, FLOATING ZERO
                                    03E7  1537 ;       PATTERNS ON MASSBUS CONTROL LINES BY USING MBE DATA
                                    03E7  1538 ;       BUFFER REGISTER.
                                    03E7  1539 ;--
        52   00000000'EF    D0   03E7  1540          MOVL    RH_CUR_ADR,R2          ; GET ADDRESS OF MUT
                                    03EE  1541          ERRPREP MBE_REG_MSG,12,-       ; PREPARE TO PRINT ERROR
                                    03EE  1542                  FMT_ANY_REG
00000000'EF   00000000'EF    DE   03EE                    MOVAL   MBE_REG_MSG,REG_NAME
         00000000'EF    0C   9A   03F9                    MOVZBL  #12,REG_NO
00000000'EF   00000000'EF    DE   0400                    MOVAL   FMT_ANY_REG,REG_STRING
        58   00000000'EF    D0   040B  1543          MOVL    MBE_CUR_ADR,R8         ; GET MBE ADDRESS
        54   00000000'8F    D0   0412  1544          MOVL    #ALL_ONES,R4           ; EXPECT ALL ONES
    0000'C2   00000000'8F    C8   0419  1545 10$:     BISL    #PGM_INIT,CR(R2)       ; INITIALIZE MUT
            0000'C8   54    D0   0422  1546          MOVL    R4,MBE_DBR(R8)         ; WRITE ALL ONES ON C-LINES
            50   0000'C8    D0   0427  1547          MOVL    MBE_DBR(R8),R0         ; READ DATA BUFFER
                 53   50    3C   042C  1548          MOVZWL  R0,R3                  ; EXTRACT REGISTER CONTENTS
                 54   53    D1   042F  1549          CMPL    R3,R4                  ; ANY STUCK AT ZERO?
                      20    13   0432  1550          BEQL    20$                    ; BRANCH IF NONE
                      55    D4   0434  1551          CLRL    R5
        56   00000000'8F    D0   0436  1552          MOVL    #MBE_DBR,R6            ; REPORT FAILING REGISTER
                                    043D  1553          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                    043D  1554                  PRINT_SBE
             00000000'EF    DF   043D                    PUSHAL  PRINT_SBE
                      00    DD   0443                    PUSHL   #0
             00000000'EF    DD   0445                    PUSHL   LUN
                      01    DD   044B                    PUSHL   #1
        00000000'9F   04    FB   044D                    CALLS   #$$M, a*DS$ERRHARD
                                    0454  1555 20$:     $DS_CKLOOP      10$            ; SCOPE LOOP?
     00000000'9F   C2 AF    FA   0454                    CALLG   10$, a*DS$CKLOOP
    0000'C2   00000000'8F    C8   045C  1556 25$:     BISL    #PGM_INIT,CR(R2)       ; INITIALIZE MUT
                      54    D4   0465  1557          CLRL    R4                     ; EXPECT ZEROES
            0000'C8   54    D0   0467  1558          MOVL    R4,MBE_DBR(R8)         ; WRITE ZEROES ON C-LINES
            50   0000'C8    D0   046C  1559          MOVL    MBE_DBR(R8),R0         ; READ DATA BUFFER
                 53   50    3C   0471  1560          MOVZWL  R0,R3                  ; EXTRACT REGISTER CONTENTS
                 54   53    D1   0474  1561          CMPL    R3,R4                  ; ANY STUCK AT ONE?
                      21    13   0477  1562          BEQL    30$                    ; BRANCH IF NONE
                 55   53    D0   0479  1563          MOVL    R3,R5                  ; REPORT BIT FAILURE
        56   00000000'8F    D0   047C  1564          MOVL    #MBE_DBR,R6            ; REPORT FAILING REGISTER
                                    0483  1565          $DS_ERRHARD_S   #2,LUN,,-       ; PRINT ERROR
                                    0483  1566                  PRINT_SBE
             00000000'EF    DF   0483                    PUSHAL  PRINT_SBE
                      00    DD   0489                    PUSHL   #0
             00000000'EF    DD   048B                    PUSHL   LUN
                      02    DD   0491                    PUSHL   #2
        00000000'9F   04    FB   0493                    CALLS   #$$M, a*DS$ERRHARD
                                    049A  1567 30$:     $DS_CKLOOP      25$            ; SCOPE LOOP?
     00000000'9F   BF AF    FA   049A                    CALLG   25$, a*DS$CKLOOP
            54   01    D0   04A2  1568          MOVL    #1,R4                  ; INITIALIZE EXPECTED PATTERN
                 59    D4   04A5  1569          CLRL    R9                     ; CLEAR COUNTER
```

```
                00000000'8F   C8  04A7  1570 40$:    BISL    #PGM_INIT,CR(R2)          ; INITIALIZE MUT
              0000'C8   54    D0  04B0  1571         MOVL    R4,MBE_DBR(R8)            ; WRITE PATTERN TO DATA BUFFER
              50   0000'C8    D0  04B5  1572         MOVL    MBE_DBR(R8),R0            ; READ DATA BUFFER
                   53   50    3C  04BA  1573         MOVZWL  R0,R3                     ; EXTRACT REGISTER CONTENTS
                   54   53    D1  04BD  1574         CMPL    R3,R4                     ; EXPECTED PATTERN?
                        21    13  04C0  1575         BEQL    50$                       ; BRANCH IF IT IS
                   55   54    D0  04C2  1576         MOVL    R4,R5                     ; REPORT BIT FAILURE
           56   00000000'8F   D0  04C5  1577         MOVL    #MBE_DBR,R6               ; REPORT FAILING REGISTER
                              04CC  1578         $DS_ERRHARD_S   #3,LUN,,-          ; PRINT ERROR
                              04CC  1579                         PRINT_SBE
                00000000'EF   DF  04CC                     PUSHAL  PRINT_SBE
                        00    DD  04D2                     PUSHL   #0
                00000000'EF   DD  04D4                     PUSHL   LUN
                        03    DD  04DA                     PUSHL   #3
           00000000'9F    04    FB  04DC                     CALLS   #$$M, a#DS$ERRHARD
                              04E3  1580 50$:    $DS_CKLOOP      40$                ; SCOPE LOOP?
         00000000'9F    C1 AF    FA  04E3                     CALLG   40$, a#DS$CKLOOP
                54   54    01    9C  04EB  1581         ROTL    #1,R4,R4                  ; FLOAT PATTERN ONE BIT
                  B4 59    0F    F3  04EF  1582         AOBLEQ  #15,R9,40$                ; PATTERN FLOATED 16 BITS?
                        59    D4  04F3  1583         CLRL    R9                        ; CLEAR COUNTER
                   54   01    D2  04F5  1584         MCOML   #1,R4                     ; EXPECTED PATTERN
0000'C2    00000000'8F   C8  04F8  1585 60$:    BISL    #PGM_INIT,CR(R2)          ; INITIALIZE MUT
              0000'C8   54    D0  0501  1586         MOVL    R4,MBE_DBR(R8)            ; WRITE PATTERN TO DATA BUFFER
              50   0000'C8    D0  0506  1587         MOVL    MBE_DBR(R8),R0            ; READ DATA BUFFER
                   53   50    3C  050B  1588         MOVZWL  R0,R3                     ; EXTRACT REGISTER CONTENTS
                   54   54    3C  050E  1589         MOVZWL  R4,R4                     ; ADJUST EXPECTED VALUE
                   54   53    D1  0511  1590         CMPL    R3,R4                     ; EXPECTED PATTERN?
                        20    13  0514  1591         BEQL    70$                       ; BRANCH IF IT IS
                   55   D4    0516  1592         CLRL    R5
           56   00000000'8F   D0  0518  1593         MOVL    #MBE_DBR,R6               ; REPORT FAILING REGISTER
                              051F  1594         $DS_ERRHARD_S   #4,LUN,,-          ; PRINT ERROR
                              051F  1595                         PRINT_SBE
                00000000'EF   DF  051F                     PUSHAL  PRINT_SBE
                        00    DD  0525                     PUSHL   #0
                00000000'EF   DD  0527                     PUSHL   LUN
                        04    DD  052D                     PUSHL   #4
           00000000'9F    04    FB  052F                     CALLS   #$$M, a#DS$ERRHARD
                              0536  1596 70$:    $DS_CKLOOP      60$                ; SCOPE LOOP?
         00000000'9F    BF AF    FA  0536                     CALLG   60$, a#DS$CKLOOP
                54   54    01    9C  053E  1597         ROTL    #1,R4,R4                  ; FLOAT ZERO OVER ONE BIT
                  B2 59    0F    F3  0542  1598         AOBLEQ  #15,R9,60$                ; FLOATED THROUGH ALL 16 BITS?
                              0546  1599 80$:    ERRPREP MBE_REG_MSG,12,-          ; PREPARE TO PRINT ERROR
                              0546  1600                         FMT_MBE_ER
00000000'EF    00000000'EF   DE  0546                     MOVAL   MBE_REG_MSG,REG_NAME
          00000000'EF    0C    9A  0551                     MOVZBL  #12,REG_NO
00000000'EF    00000000'EF   DE  0558                     MOVAL   FMT_MBE_ER,REG_STRING
              50   0000'C8    D0  0563  1601         MOVL    MBE_ER(R8),R0             ; READ MBE ERROR REGISTER
                   53   50    3C  0568  1602         MOVZWL  R0,R3                     ; EXTRACT REGISTER CONTENTS
           54   00000000'8F   D0  056B  1603         MOVL    #CPE,R4                   ; EXPECT NO CONTROL PARITY ERROR
                   53   54    D3  0572  1604         BITL    R4,R3                     ; IS THERE A CONTROL PE?
                        21    13  0575  1605         BEQL    90$                       ; SKIP IF NOT
                   55   53    D0  0577  1606         MOVL    R3,R5                     ; REPORT FAILING BIT
           56   00000000 8F   D0  057A  1607         MOVL    #MBE_ER,R6                ; REPORT FAILING REGISTER
                              0581  1608         $DS_ERRHARD_S   #5,LUN,,-          ; PRINT ERROR
                              0581  1609                         PRINT_SBE
                00000000'EF   DF  0581                     PUSHAL  PRINT_SBE
                        00    DD  0587                     PUSHL   #0
```

```
                00000000'EF   DD  0589                          PUSHL   LUN
                         05   DD  058F                          PUSHL   #5
           00000000'9F   04   FB  0591                          CALLS   #$$M, a#DS$ERRHARD
                                  0598  1610 90$:    $DS_CKLOOP      60$                ; SCOPE LOOP?
      00000000'9F   FF5C CF   FA  0598                          CALLG   60$, a#DS$CKLOOP
                                  05A1  1611 $DS_ENDSUB
                                  05A1       T20_S5_X:
   00000000'9F   00000144'EF   FA  05A1                          CALLG   $S$, a#DS$ENDSUB
```

```
                                05AC  1613 $DS_BGNSUB
                                05AC       T20_S6::
    00000000'9F   00000:50'EF   FA 05AC                    CALLG   $S$, a#DS$BGNSUB
                                05B7  1614 ;++
                                05B7  1615 ; TEST MASSBUS OCCUPIED LINE
                                05B7  1616 ;
                                05B7  1617 ; TEST ALGORITHM:
                                05B7  1618 ;
                                05B7  1619 ;       THE ASSERTION OF OCCUPIED IS TESTED DURING ANY NORMAL DATA
                                05B7  1620 ;       TRANSFER OPERATION. THE NEGATION OF OCCUPIED WILL BE TESTED
                                05B7  1621 ;       HERE BY DISABLING OCCUPIED AND PERFORMING A DATA TRANSFER,
                                05B7  1622 ;       TESTING FOR MISSED TRANSFER IN THE RH750 STATUS REGISTER.
                                05B7  1623 ;--
                                05B7  1624             ERRPREP RHSR_MSG,2,-             ; PREPARE TO PRINT ERROR
                                05B7  1625                     FMT_STATUS_REG
    00000000'EF   00000000'EF   DE 05B7                        MOVAL   RHSR_MSG,REG_NAME
          00000000'EF   02      9A 05C2                        MOVZBL  #2,REG_NO
    00000000'EF   00000000'EF   DE 05C9                        MOVAL   FMT_STATUS_REG,REG_STRING
          52    00000000'EF     D0 05D4  1626             MOVL    RH_CUR_ADR,R2          ; GET ADDRESS OF MUT
          58    00000000'EF     D0 05DB  1627             MOVL    MBE_CUR_ADR,R8         ; GET ADDRESS OF MBE
       59 52    00000000'8F     C1 05E2  1628             ADDL3   #MAP_OFFSET,R2,R9      ; CALCULATE RH750 MAP REGISTER ADDRESS
       0000'C2  00000000'8F     C8 05EA  1629 10$:        BISL    #PGM_INIT,CR(R2)       ; INITIALIZE MUT
                00000000'EF     D4 05F3  1630             CLRL    DATA_OBUFFER           ; CLEAR FIRST FOUR BYTES OF DATA BUF
          5B    00000000'EF     DE 05F9  1631             MOVAL   DATA_OBUFFER,R11       ; GET ADDRESS OF WRITE BUFFER
           5A    5B    F7 8F    78 0600  1632             ASHL    #-9,R11,R10            ; PUT PAGE FRAME NUMBER INTO R10
          5A    00000000'8F     C8 0605  1633             BISL    #VALID_BIT,R10         ; SET V BIT
                69    5A        D0 060C  1634             MOVL    R10,(R9)               ; SET UP RH750 MAP REGISTER 0
          5B    00000000'8F     CA 060F  1635             BICL    #MAP_PTR_MSK,R11       ; MASK OUT PAGE FRAME NUMBER TO
                                0616  1636                                            ; OBTAIN BYTE OFFSET
              0000'C2   5B      D0 0616  1637             MOVL    R11,VAR(R2)            ; WRITE RH750 VIRTUAL ADDRESS REGISTER
                                061B  1638                                            ; SELECTING MAP 0
              0000'C2   02      CE 061B  1639             MNEGL   #2,BCR(R2)             ; SET UP RH750 BYTE COUNT
              0000'C8   07      D0 0620  1640             MOVL    #7,MBE_CR2(R8)         ; SET UP MBE PARAMETERS:
                                0625  1641                                            ;       BLOCK SIZE = 1 WORD
                                0625  1642                                            ;       GAP SIZE = 5 US
                                0625  1643                                            ;       DATA RATE = 1 WORD/US
0000'C8   00000000'8F     C8 0625  1644             BISL    #DOCC,MBE_MR(R8)       ; SET DISABLE OCCUPIED IN MBE
0000'C8   00000000'8F     C8 062E  1645             BISL    #DMD,MBE_MR(R8)        ; PUT MBE INTO DIAGNOSTIC MODE
0000'C8   00000000'8F     D0 0637  1646             MOVL    #WRITE,MBE_CR1(R8)     ; INITIATE WRITE COMMAND
                                0640  1647             $DS_WAITUS_S    #10            ; WAIT FOR SILO TO FILL
                     00   DD 0640                        PUSHL   #0
                     0A   DD 0642                        PUSHL   #10
          00000000'9F    02 FB 0644                        CALLS   #2, a#DS$WAITUS
0000'C8   00000000'8F     C8 064B  1648             BISL    #MCLK,MBE_MR(R8)       ; ASSERT MAINTENANCE CLOCK
0000'C8   00000000'8F     CA 0654  1649             BICL    #MCLK,MBE_MR(R8)       ; NEGATE MAINTENANCE CLOCK
                                065D  1650             $DS_WAITUS_S    #60            ; (M1.1) WAIT 500 US
                     00   DD 065D                        PUSHL   #0
                     3C   DD 065F                        PUSHL   #60
          00000000'9F    02 FB 0661                        CALLS   #2, a#DS$WAITUS
       53   0000'C2     D0 0668  1651             MOVL    SR(R2),R3              ; READ RH750 STATUS REGISTER
    54    00000000'8F     D0 066D  1652             MOVL    #MISSED_XFER,R4        ; EXPECT MISSED TRANSFER
             53   54      D3 0674  1653             BITL    R4,R3                  ; MISSED TRANSFER SET?
                  1A      12 0677  1654             BNEQ    20$                    ; BRANCH IF IT IS
             55   54      D0 0679  1655             MOVL    R4,R5                  ; REPORT FAILING BIT
                                067C  1656             $DS_ERRHARD_S   #1,LUN,,-      ; PRINT ERROR
                                067C  1657                             PRINT_SBE
          00000000'EF       DF 067C                        PUSHAL  PRINT_SBE
```

```
                        00   DD  0682                          PUSHL   #0
                00000000'EF  DD  0684                          PUSHL   LUN
                        01   DD  068A                          PUSHL   #1
          00000000'9F   04   FB  068C                          CALLS   #$$M, a#DS$ERRHARD
                                 0693  1658 20$:    $DS_CKLOOP      10$                ; SCOPE LOOP?
     00000000'9F   FF53 CF   FA  0693                          CALLG   10$, a#DS$CKLOOP
     0000'C2   00000000'8F   C8  069C  1659         BISL     #PGM_INIT,CR(R2)          ; CLEAR MXF, DOCC, DMD
                                 06A5  1660 $DS_ENDSUB
                                 06A5       T20_S6_X:
  00000000'9F   00000150'EF   FA  06A5                          CALLG   $S$, a#DS$ENDSUB
```

```
                                    06B0  1662 $DS_BGNSUB
                                    06B0       T20_S7::
00000000'9F   0000015C'EF   FA      06B0                CALLG   $S$, a#DS$BGNSUB
                                    06BB  1663 ;++
                                    06BB  1664 ; TEST MASSBUS FAIL
                                    06BB  1665 ;
                                    06BB  1666 ; TEST ALGORITHM:
                                    06BB  1667 ;
                                    06BB  1668 ;       ASSERT MASSBUS FAIL BY PLACING RH750 INTO MAINTENANCE MODE.
                                    06BB  1669 ;       CHECK THAT THERE IS NO CHANGE IN AN MBE REGISTER BY CHECKING
                                    06BB  1670 ;       RFAIL BIT, WHICH IS NORMALLY NOT SET.  ALTHOUGH IT BECOMES
                                    06BB  1671 ;       SET IN REALITY WHEN RH750 GOES INTO MAINTENANCE MODE, WITH
                                    06BB  1672 ;       MASSBUS FAIL ASSERTED ALL MBE REGISTERS SHOULD YIELD WHEN
                                    06BB  1673 ;       READ THEIR PREVIOUS VALUES.
                                    06BB  1674 ;--
                                    06BB  1675          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                    06BB  1676                  FMT_STATUS_REG
00000000'EF   00000000'EF   DE      06BB                        MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02    9A      06C6                        MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE      06CD                        MOVAL   FMT_STATUS_REG,REG_STRING
        52    00000000'EF   D0      06D8  1677          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
        58    00000000'EF   D0      06DF  1678          MOVL    MBE_CUR_ADR,R8          ; GET ADDRESS OF MBE
    0000'C2   00000000'8F   C8      06E6  1679 10$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
    0000'C2   00000000'8F   C8      06EF  1680          BISL    #MAINT_MODE,CR(R2)      ; PUT RH750 INTO MAINTENANCE MODE
                                    06F8  1681                                          ; ASSERTING MASS FAIL
            50    0000'C8   D0      06F8  1682          MOVL    MBE_ER(R8),R0           ; READ MBE ERROR REGISTER
                  53    50  3C      06FD  1683          MOVZWL  R0,R3                   ; ISOLATE MBE DATA
                        54  D4      0700  1684          CLRL    R4                      ; EXPECT NOTHING
                 29 53    06  E1    0702  1685          BBC     #6,R3,20$               ; BRANCH IF OKAY
                    55    53  D0    0706  1686          MOVL    R3,R5                   ; REPORT FAILING BIT
        56    00000000'8F   D0      0709  1687          MOVL    #MBE_ER,R6              ; REPORT FAILING REGISTER
                                    0710  1688          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                                    0710  1689                          PRINT_SBE
                00000000'EF   DF    0710                        PUSHAL  PRINT_SBE
                              00    DD  0716                    PUSHL   #0
                00000000'EF   DD    0718                        PUSHL   LUN
                              01    DD  071E                    PUSHL   #1
        00000000'9F   04    FB      0720                        CALLS   #$$M, a#DS$ERRHARD
                                    0727  1690          $DS_CKLOOP      10$             ; SCOPE LOOP?
  00000000'9F   BC AF       FA      0727                        CALLG   10$, a#DS$CKLOOP
0000'C2   00000000'8F   C8      072F  1691 20$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        50    0000'C8   D0      0738  1692          MOVL    MBE_ER(R8),R0           ; READ MBE ERROR REGISTER
              53    50  3C      073D  1693          MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
              53    54  D3      0740  1694          BITL    R4,R3                   ; IS RFAIL STILL SET?
                    21  13      0743  1695          BEQL    30$                     ; BRANCH IF IT IS NOT
              55    54  D0      0745  1696          MOVL    R4,R5                   ; REPORT FAILING BIT
        56    00000000'8F   D0      0748  1697          MOVL    #MBE_ER,R6              ; REPORT FAILING REGISTER
                                    074F  1698          $DS_ERRHARD_S   #2,LUN,,-       ; PRINT ERROR
                                    074F  1699                          PRINT_SBE
                00000000'EF   DF    074F                        PUSHAL  PRINT_SBE
                              00    DD  0755                    PUSHL   #0
                00000000'EF   DD    0757                        PUSHL   LUN
                              02    DD  075D                    PUSHL   #2
        00000000'9F   04    FB      075F                        CALLS   #$$M, a#DS$ERRHARD
                                    0766  1700 30$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
  00000000'9F   C6 AF       FA      0766                        CALLG   20$, a#DS$CKLOOP
                                    076E  1701 $DS_ENDSUB
```

```
                                  076E        T20_S7_X:
    00000000'9F   0000015C'EF   FA  076E                        CALLG   $S$, a#DS$ENDSUB
```

```
                                0779  1703 $DS_BGNSUB
                                0779       T20_S8::
00000000'9F   00000168'EF   FA  0779                 CALLG   $S$, a#DS$BGNSUB
                                0784  1704 ;++
                                0784  1705 ; DATA TRANSFER: ALL ONE'S PATTERN
                                0784  1706 ;       SINCE MBE ONLY RETAINS LAST WORD TRANSFERED, THE OUTPUT BUFFER
                                0784  1707 ;       IS INITIALIZED TO THE COMPLIMENT OF THE EXPECTED PATTERN
                                0784  1708 ;       EXCEPT FOR THE LAST WORD.
                                0784  1709 ;--
                                0784  1710          ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                0784  1711                  FMT_MBE_SR
00000000'EF   00000000'EF   DE  0784                 MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF   0C    9A  078F                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  0796                 MOVAL   FMT_MBE_SR,REG_STRING
        52    00000000'EF   D0  07A1  1712          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
    0000'C2   00000000'8F   C8  07A8  1713          BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        58    00000000'EF   D0  07B1  1714          MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
    59  52    00000000'8F   C1  07B8  1715          ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                    50      D4  07C0  1716          CLRL    R0                      ; CLEAR INDEX
              00000000'EF40 D4  07C2  1717 10$:     CLRL    DATA_OBUFFER R0         ; PUT ZERO'S PATTERN INTO BUFFER
              00000000'EF40 D4  07C9  1718          CLRL    DATA_IBUFFER R0         ; CLEAR OUT READ BUFFER
      EA 50     0000007F 8F   F3  07D0  1719          AOBLEQ  #127,R0,10$             ; FILL UP BOTH BUFFERS
        000001FE'EF   00    B2  07D8  1720          MCOMW   #0,DATA_OBUFFER+510     ; PUT ONE'S PATTERN IN LAST WORD OF BUFFER
    0000'C2   00000000'8F   C8  07DF  1721 20$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        5B    00000000'EF   DE  07E8  1722          MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
         5A    5B    F7 8F   78  07EF  1723          ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
        5A    00000000'8F   C8  07F4  1724          BISL    #VALID_BIT,R10          ; SET VALID BIT
                 69    5A   D0  07FB  1725          MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
        5B    00000000'8F   CA  07FE  1726          BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
            0000'C2   5B    D0  0805  1727          MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                              080A  1728                                          ; (SELECTING MAP 0)
    0000'C2   00000200 8F   CE  080A  1729          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
    0000'C8   00000000'8F   D0  0813  1730          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              081C  1731                                          ;       BLOCK SIZE = 256 W
                              081C  1732                                          ;       GAP SIZE = 5 US
                              081C  1733                                          ;       DATA RATE = 1 MW/S
    0000'C8   00000000'8F   D0  081C  1734          MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                              0825  1735          $DS_WAITUS_S    #100            ; WAIT 1 MILLISECONDS
                    00      DD  0825                        PUSHL   #0
            00000064 8F     DD  0827                        PUSHL   #100
        00000000'9F   02    FB  082D                        CALLS   #2, a#DS$WAITUS
            50    0000'C8   D0  0834  1736          MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
                  53    50  3C  0839  1737          MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
             6B 53    0E  E1  083C  1738          BBC     #14,R3,30$              ; ERROR BIT SET?
        56    00000000'8F   D0  0840  1739          MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                    54      D4  0847  1740          CLRL    R4                      ; EXPECTING NO ERRORS
                  55    53  D0  0849  1741          MOVL    R3,R5                   ; REPORT FAILING BIT
                              084C  1742          $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                              084C  1743                          PRINT_SBE
            00000000'EF   DF  084C                        PUSHAL  PRINT_SBE
                    00      DD  0852                        PUSHL   #0
            00000000'EF     DD  0854                        PUSHL   LUN
                    01      DD  085A                        PUSHL   #1
        00000000'9F   04    FB  085C                        CALLS   #$$M, a#DS$ERRHARD
                              0863  1744          ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO HANDLE ERROR
                              0863  1745                  FMT_MBE_ER
00000000'EF   00000000'EF   DE  0863                 MOVAL   MBE_REG_MSG,REG_NAME
```

```
                00000000'EF   0C   9A  086E                        MOVZBL  #12,REG_NO
        00000000'EF   00000000'EF   DE  0875                        MOVAL   FMT_MBE_ER,REG_STRING
                51   0000'C8   D0  0880  1746          MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                53   51   3C  0885  1747          MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
                     21   13  0888  1748          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                55   53   D0  088A  1749          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
           56   00000000'8F   D0  088D  1750          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                              0894  1751          $DS_ERRHARD_S   #2,LUN,,-       ; PRINT ERROR
                              0894  1752                          PRINT_SBE
                00000000'EF   DF  0894                        PUSHAL  PRINT_SBE
                         00   DD  089A                        PUSHL   #0
                00000000'EF   DD  089C                        PUSHL   LUN
                         02   DD  08A2                        PUSHL   #2
           00000000'9F   04   FB  08A4                        CALLS   #$$M, a#DS$ERRHARD
                              08AB  1753 30$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
        00000000'9F   FF30 CF   FA  08AB                        CALLG   20$, a#DS$CKLOOP
        0000'C2   00000000'8F   C8  08B4  1754 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
              5B   00000000'EF   DE  08BD  1755          MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
               5A   5B   F7 8F   78  08C4  1756          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
              5A   00000000'8F   C8  08C9  1757          BISL    #VALID_BIT,R10          ; SET V BIT
                        69   5A   D0  08D0  1758          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
              5B   00000000'8F   CA  08D3  1759          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
                   0000'C2   5B   D0  08DA  1760          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
        0000'C2   00000200 8F   CE  08DF  1761          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
        0000'C8   00000000'8F   D0  08E8  1762          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              08F1  1763                                              ; BLOCK SIZE = 256 W
                              08F1  1764                                              ; GAP SIZE = 5 US
                              08F1  1765                                              ; DATA RATE = 1 MW/S
     0000'C8   00000000'8F   D0  08F1  1766          MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                              08FA  1767          $DS_WAITUS_S    #100            ; WAIT 1 MILLISECONDS
                         00   DD  08FA                        PUSHL   #0
                   00000064 8F   DD  08FC                        PUSHL   #100
              00000000'9F   02   FB  0902                        CALLS   #2, a#DS$WAITUS
                              0909  1768          ERRPREP RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                              0909  1769                          FMT_STATUS_REG
00000000'EF   00000000'EF   DE  0909                        MOVAL   RHSR_MSG,REG_NAME
        00000000'EF   02   9A  0914                        MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  091B                        MOVAL   FMT_STATUS_REG,REG_STRING
           53   0000'C2   D0  0926  1770          MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
        53   00000000'8F   D3  092B  1771          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                         1C   13  0932  1772          BEQL    50$                     ; BRANCH IF NONE
                         54   D4  0934  1773          CLRL    R4                      ; EXPECT NO ERRORS
                    55   53   D0  0936  1774          MOVL    R3,R5                   ; REPORT FAILING BITS
                              0939  1775          $DS_ERRHARD_S   #3,LUN,,-       ; PRINT ERROR
                              0939  1776                          PRINT_SBE
                00000000'EF   DF  0939                        PUSHAL  PRINT_SBE
                         00   DD  093F                        PUSHL   #0
                00000000'EF   DD  0941                        PUSHL   LUN
                         03   DD  0947                        PUSHL   #3
           00000000'9F   04   FB  0949                        CALLS   #$$M, a#DS$ERRHARD
                              0950  1777 50$:     $DS_CKLOOP      35$             ; SCOPE LOOP
     00000000'9F   FF60 CF   FA  0950                        CALLG   35$, a#DS$CKLOOP
                         55   D4  0959  1778          CLRL    R5                      ; CLEAR INDEX
                         59   D4  095B  1779          CLRL    R9                      ; CLEAR INDEX
        54   000001FE'EF   B0  095D  1780 60$:     MOVW    DATA_OBUFFER+510,R4     ; EXPECTED DATA
     53   00000000'EF49   B0  0964  1781          MOVW    DATA_IBUFFER R9 ,R3     ; RECEIVED DATA
                    54   53   B1  096C  1782          CMPW    R3,R4                   ; EXPECTED=RECEIVED?
```

```
                    2D   13  096F  1783          BEQL    70$                          ; BRANCH IF THEY ARE EQUAL
                    55   D6  0971  1784          INCL    R5                           ; ADD ONE TO ERROR COUNT
                55  08   D1  0973  1785          CMPL    #8,R5                        ; PRINTED ERROR EIGHT TIMES?
                    26   19  0976  1786          BLSS    70$                          ; BRANCH IF SO
        56  000001FE'EF   DE  0978  1787          MOVAL   DATA_OBUFFER+510,R6          ; GET ADDRESS OF EXP DATA BUFFER
     57  00000000'EF49   3E  097F  1788          MOVAW   DATA_IBUFFER R9 ,R7          ; GET ADDRESS OF REC DATA BUFFER
                              0987  1789          $DS_ERRHARD_S   #4,LUN,,-            ; PRINT ERROR
                              0987  1790                          PRINT_DTERR
            00000000'EF   DF  0987                          PUSHAL  PRINT_DTERR
                    00   DD  098D                          PUSHL   #0
            00000000'EF   DD  098F                          PUSHL   LUN
                    04   DD  0995                          PUSHL   #4
        00000000'9F  04   FB  0997                          CALLS   #$$M, @#DS$ERRHARD
                              099E  1791 70$:     $DS_CKLOOP      20$                  ; SCOPE LOOP?
    00000000'9F   FE3D CF   FA  099E                          CALLG   20$, @#DS$CKLOOP
      AE 59  000000FF 8F   F3  09A7  1792          AOBLEQ  #255,R9,60$                  ; CHECK ENTIRE BUFFER
                    55   D5  09AF  1793          TSTL    R5                           ; ANY ERRORS REPORTED?
                    0F   13  09B1  1794          BEQL    80$                          ; BRANCH IF NONE
                              09B3  1795          $DS_PRINTB_S    FMT_ERRSUM,R5        ; REPORT NUMBER OF ERRORS
                    55   DD  09B3                          PUSHL   R5
            00000000'EF   9F  09B5                          PUSHAB  FMT_ERRSUM
        00000000'9F  02   FB  09BB                          CALLS   #$$N, @#DS$PRINTB
                              09C2  1796 80$:
                              09C2  1797 $DS_ENDSUB
                              09C2       T20_S8_X:
  00000000'9F  00000168'EF   FA  09C2                          CALLG   $S$, @#DS$ENDSUB
```

```
                                09CD  1799 $DS_BGNSUB
                                09CD       T20_S9::
00000000'9F   00000174'EF   FA  09CD                CALLG   $S$, a#DS$BGNSUB
                                09D8  1800 ;++
                                09D8  1801 ; DATA TRANSFER: ALL ZERO'S
                                09D8  1802 ;       SINCE THE MBE ANLY RETAINS THE LAST WORD TRANSFERED,
                                09D8  1803 ;       THE OUTPUT BUFFER IS INITIALIZED TO THE COMPLIMENT
                                09D8  1804 ;       OF THE EXPECTED DATA EXCEPT FOR THE LAST WORD.
                                09D8  1805 ;--
                                09D8  1806         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                09D8  1807                 FMT_MBE_SR
00000000'EF   00000000'EF   DE  09D8                MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  09E3                MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  09EA                MOVAL   FMT_MBE_SR,REG_STRING
         52   00000000'EF   D0  09F5  1808          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
         58   00000000'EF   D0  09FC  1809          MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
    0000'C2   00000000'8F   C8  0A03  1810          BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
    59   52   00000000'8F   C1  0A0C  1811          ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                        50  D4  0A14  1812          CLRL    R0                      ; CLEAR INDEX
      00000000'EF40   00    D2  0A16  1813 10$:     MCOML   #0,DATA_OBUFFER R0      ; FILL BUFFER WITH ONE'S
      00000000'EF40   00    D2  0A1E  1814          MCOML   #0,DATA_IBUFFER R0      ; AND INPUT BUFFER
     E8 50   0000007F 8F    F3  0A26  1815          AOBLEQ  #127,R0,10$             ; FILL UP BOTH BUFFERS
             000001FE'EF    B4  0A2E  1816          CLRW    DATA_OBUFFER+510         ; PUT EXPECTED PATTERN IN LAST WORD OF BUFFE
    0000'C2   00000000'8F   C8  0A34  1817 20$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
         5B   00000000'EF   DE  0A3D  1818          MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
          5A    5B   F7 8F   78  0A44  1819          ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
         5A   00000000'8F   C8  0A49  1820          BISL    #VALID_BIT,R10          ; SET VALID BIT
              69   5A   D0  0A50  1821          MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
         5B   00000000'8F   CA  0A53  1822          BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
            0000'C2   5B   D0  0A5A  1823          MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                           0A5F  1824                                              ; (SELECTING MAP 0)
    0000'C2   00000200 8F   CE  0A5F  1825          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
    0000'C8   00000000'8F   D0  0A68  1826          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                            0A71  1827                                              ;       BLOCK SIZE = 256 W
                            0A71  1828                                              ;       GAP SIZE = 5 US
                            0A71  1829                                              ;       DATA RATE = 1 MW/S
    0000'C8   00000000'8F   D0  0A71  1830          MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                            0A7A  1831          $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                        00  DD  0A7A                    PUSHL   #0
              00000064  8F  DD  0A7C                    PUSHL   #100
         00000000'9F    02  FB  0A82                    CALLS   #2, a#DS$WAITUS
             50   0000'C8   D0  0A89  1832          MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
                  53   50   3C  0A8E  1833          MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
               6B 53   0E   E1  0A91  1834          BBC     #14,R3,30$              ; ERROR BIT SET?
         56   00000000'8F   D0  0A95  1835          MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                        54  D4  0A9C  1836          CLRL    R4                      ; EXPECTING NO ERRORS
                  55   53   D0  0A9E  1837          MOVL    R3,R5                   ; REPORT FAILING BIT
                            0AA1  1838          $DS_ERRHARD_S   #1,LUN,,-           ; PRINT ERROR
                            0AA1  1839                          PRINT_SBE
              00000000'EF   DF  0AA1                    PUSHAL  PRINT_SBE
                        00  DD  0AA7                    PUSHL   #0
              00000000'EF   DD  0AA9                    PUSHL   LUN
                        01  DD  0AAF                    PUSHL   #1
         00000000'9F    04  FB  0AB1                    CALLS   #$$M, a#DS$ERRHARD
                            0AB8  1840          ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                            0AB8  1841                  FMT_MBE_ER
00000000'EF   00000000'EF   DE  0AB8                    MOVAL   MBE_REG_MSG,REG_NAME
```

```
                00000000'EF   0C   9A  0AC3                        MOVZBL  #12,REG_NO
      00000000'EF   00000000'EF   DE  0ACA                        MOVAL   FMT_MBE_ER,REG_STRING
                51    0000'C8   D0  0AD5  1842          MOVL      MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                      53    51  3C  0ADA  1843          MOVZWL    R1,R3                   ; EXTRACT REGISTER CONTENTS
                            21  13  0ADD  1844          BEQL      30$                     ; IF NO ERROR BITS SET, SKIP
                      55    53  D0  0ADF  1845          MOVL      R3,R5                   ; REPORT FAILING BIT(S)
             56   00000000'8F   D0  0AE2  1846          MOVL      #MBE_ER,R6              ; FAILING REGISTER
                                  0AE9  1847          $DS_ERRHARD_S   #2,LUN,,-         ; PRINT ERROR
                                  0AE9  1848                        PRINT_SBE
                00000000'EF   DF  0AE9                        PUSHAL  PRINT_SBE
                            00  DD  0AEF                        PUSHL   #0
                00000000'EF   DD  0AF1                        PUSHL   LUN
                            02  DD  0AF7                        PUSHL   #2
             00000000'9F    04  FB  0AF9                        CALLS   #$$M, a#DS$ERRHARD
                                  0B00  1849 30$:     $DS_CKLOOP      20$               ; SCOPE LOOP?
      00000000'9F   FF30 CF   FA  0B00                        CALLG   20$, a#DS$CKLOOP
      0000'C2   00000000'8F   C8  0B09  1850 35$:     BISL      #PGM_INIT,CR(R2)        ; INITIALIZE MUT
            5B  00000000'EF   DE  0B12  1851          MOVAL     DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
             5A   5B   F7 8F   78  0B19  1852          ASHL      #-9,R11,R10             ; PUT PFN INTO R10
            5A  00000000'8F   C8  0B1E  1853          BISL      #VALID_BIT,R10          ; SET V BIT
                      69    5A  D0  0B25  1854          MOVL      R10,(R9)                ; PUT PFN INTO MAP 0
            5B  00000000'8F   CA  0B28  1855          BICL      #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
                0000'C2     5B  D0  0B2F  1856          MOVL      R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
      0000'C2   00000200 8F   CE  0B34  1857          MNEGL     #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
      0000'C8   00000000'8F   D0  0B3D  1858          MOVL      #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                                  0B46  1859                                            ;       BLOCK SIZE=256 WORDS
                                  0B46  1860                                            ;       GAP SIZE=5 US
                                  0B46  1861                                            ;       DATA RATE=1 MW/SEC
      0000'C8   00000000'8F   D0  0B46  1862          MOVL      #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                                  0B4F  1863          $DS_WAITUS_S    #100              ; WAIT 1 MILLISECONDS
                            00  DD  0B4F                        PUSHL   #0
                  00000064 8F   DD  0B51                        PUSHL   #100
             00000000'9F    02  FB  0B57                        CALLS   #2, a#DS$WAITUS
                                  0B5E  1864          ERRPREP   RHSR_MSG,2,-            ; PREPARE TO PRINT ERROR
                                  0B5E  1865                    FMT_STATUS_REG
00000000'EF   00000000'EF   DE  0B5E                        MOVAL   RHSR_MSG,REG_NAME
      00000000'EF   02  9A  0B69                        MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  0B70                        MOVAL   FMT_STATUS_REG,REG_STRING
            53    0000'C2   D0  0B7B  1866          MOVL      SR(R2),R3               ; READ RH750 STATUS REGISTER
         53   00000000'8F   D3  0B80  1867          BITL      #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                      1C  13  0B87  1868          BEQL      50$                     ; BRANCH IF NONE
                      54  D4  0B89  1869          CLRL      R4                      ; EXPECT NO ERRORS
                55    53  D0  0B8B  1870          MOVL      R3,R5                   ; REPORT FAILING BITS
                                  0B8E  1871          $DS_ERRHARD_S   #3,LUN,,-         ; PRINT ERROR
                                  0B8E  1872                        PRINT_SBE
                00000000'EF   DF  0B8E                        PUSHAL  PRINT_SBE
                            00  DD  0B94                        PUSHL   #0
                00000000'EF   DD  0B96                        PUSHL   LUN
                            03  DD  0B9C                        PUSHL   #3
             00000000'9F    04  FB  0B9E                        CALLS   #$$M, a#DS$ERRHARD
                                  0BA5  1873 50$:     $DS_CKLOOP      35$               ; SCOPE LOOP
      00000000'9F   FF60 CF   FA  0BA5                        CALLG   35$, a#DS$CKLOOP
                            55  D4  0BAE  1874          CLRL      R5                      ; CLEAR INDEX
                            59  D4  0BB0  1875          CLRL      R9                      ; CLEAR INDEX
         54   000001FE'EF    B0  0BB2  1876 60$:     MOVW      DATA_OBUFFER+510,R4     ; EXPECTED DATA
      53   00000000'EF49     B0  0BB9  1877          MOVW      DATA_IBUFFER R9 ,R3     ; RECEIVED DATA
                54    53  B1  0BC1  1878          CMPW      R3,R4                   ; EXPECTED=RECEIVED?
```

```
                    2D   13   0BC4  1879          BEQL    70$                     ; BRANCH IF THEY ARE EQUAL
                    55   D6   0BC6  1880          INCL    R5                      ; ADD ONE TO ERROR COUNT
                55  08   D1   0BC8  1881          CMPL    #8,R5                   ; PRINTED ERROR EIGHT TIMES?
                    26   19   0BCB  1882          BLSS    70$                     ; BRANCH IF SO
          56  000001FE'EF   DE   0BCD  1883          MOVAL   DATA_OBUFFER+510,R6     ; GET ADDRESS OF EXP DATA BUFFER
        57  00000000'EF49   3E   0BD4  1884          MOVAW   DATA_IBUFFER R9 ,R7     ; GET ADDRESS OF REC DATA BUFFER
                         0BDC  1885          $DS_ERRHARD_S   #4,LUN,,-               ; PRINT ERROR
                         0BDC  1886                          PRINT_DTERR
            00000000'EF   DF   0BDC                  PUSHAL  PRINT_DTERR
                    00   DD   0BE2                  PUSHL   #0
            00000000'EF   DD   0BE4                  PUSHL   LUN
                    04   DD   0BEA                  PUSHL   #4
        00000000'9F   04   FB   0BEC                  CALLS   #$$M, a#DS$ERRHARD
                         0BF3  1887 70$:     $DS_CKLOOP      20$                     ; SCOPE LOOP?
      00000000'9F   FE3D CF   FA   0BF3                  CALLG   20$, a#DS$CKLOOP
        AE 59   000000FF 8F   F3   0BFC  1888          AOBLEQ  #255,R9,60$             ; CHECK ENTIRE BUFFER
                    55   D5   0C04  1889          TSTL    R5                      ; ANY ERRORS REPORTED?
                    0F   13   0C06  1890          BEQL    80$                     ; BRANCH IF NONE
                         0C08  1891          $DS_PRINTB_S    FMT_ERRSUM,R5           ; REPORT NUMBER OF ERRORS
                    55   DD   0C08                  PUSHL   R5
            00000000'EF   9F   0C0A                  PUSHAB  FMT_ERRSUM
        00000000'9F   02   FB   0C10                  CALLS   #$$N, a#DS$PRINTB
                         0C17  1892 80$:
                         0C17  1893 $DS_ENDSUB
                         0C17       T20_S9_X:
  00000000'9F   00000174'EF   FA   0C17                  CALLG   $S$, a#DS$ENDSUB
```

```
                                0C22  1895 $DS_BGNSUB
                                0C22       T20_S10::
00000000'9F   00000180'EF   FA  0C22                    CALLG   $S$, a#DS$BGNSUB
                                0C2D  1896 ;++
                                0C2D  1897 ; DATA TRANSFER: F00F
                                0C2D  1898 ;       SEE NOTE ON PREVIOUS SUBTEST
                                0C2D  1899 ;--
                                0C2D  1900         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                0C2D  1901                 FMT_MBE_SR
00000000'EF   00000000'EF   DE  0C2D                    MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  0C38                    MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  0C3F                    MOVAL   FMT_MBE_SR,REG_STRING
      52      00000000'EF   D0  0C4A  1902         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
  0000'C2     00000000'8F   C8  0C51  1903         BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      58      00000000'EF   D0  0C5A  1904         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
   59   52     00000000'8F   C1  0C61  1905         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                      50    D4  0C69  1906         CLRL    R0                      ; CLEAR INDEX
 00000000'EF40    0FF0 8F   B0  0C6B  1907 10$:    MOVW    # X0FF0,DATA_OBUFFER R0 ; PUT COMPLIMENT PATTERN INTO BUFFER
          00000000'EF40      B4  0C75  1908         CLRW    DATA_IBUFFER R0         ; CLEAR OUT READ BUFFER
    E7 50     000000FF 8F   F3  0C7C  1909         AOBLEQ  #255,R0,10$             ; FILL UP BOTH BUFFERS
  000001FE'EF    F00F 8F   B0  0C84  1910         MOVW    # XF00F,DATA_OBUFFER+510; SET LAST WORD OF BUFFER
  0000'C2     00000000'8F   C8  0C8D  1911 20$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      5B      00000000'EF   DE  0C96  1912         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
       5A   5B    F7 8F   78  0C9D  1913         ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
      5A      00000000'8F   C8  0CA2  1914         BISL    #VALID_BIT,R10          ; SET VALID BIT
                 69   5A    D0  0CA9  1915         MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
      5B      00000000'8F   CA  0CAC  1916         BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
           0000'C2   5B     D0  0CB3  1917         MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                             0CB8  1918                                         ; (SELECTING MAP 0)
  0000'C2     00000200 8F   CE  0CB8  1919         MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
  0000'C8     00000000'8F   D0  0CC1  1920         MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                             0CCA  1921                                         ;       BLOCK SIZE = 256 W
                             0CCA  1922                                         ;       GAP SIZE = 5 US
                             0CCA  1923                                         ;       DATA RATE = 1 MW/S
  0000'C8     00000000'8F   D0  0CCA  1924         MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                             0CD3  1925         $DS_WAITUS_S    #100            ; WAIT 1 MILLISECONDS
                      00     DD  0CD3                    PUSHL   #0
                00000064 8F   DD  0CD5                    PUSHL   #100
          00000000'9F   02    FB  0CDB                    CALLS   #2, a#DS$WAITUS
              50    0000'C8   D0  0CE2  1926         MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
                  53    50    3C  0CE7  1927         MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
                6B 53     0E  E1  0CEA  1928         BBC     #14,R3,30$              ; ERROR BIT SET?
          56    00000000'8F   D0  0CEE  1929         MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                      54    D4  0CF5  1930         CLRL    R4                      ; EXPECTING NO ERRORS
                55    53    D0  0CF7  1931         MOVL    R3,R5                   ; REPORT FAILING BIT
                             0CFA  1932         $DS_ERRHARD_S   #1,LUN,,-        ; PRINT ERROR
                             0CFA  1933                 PRINT_SBE
                00000000'EF   DF  0CFA                    PUSHAL  PRINT_SBE
                      00     DD  0D00                    PUSHL   #0
                00000000'EF   DD  0D02                    PUSHL   LUN
                      01     DD  0D08                    PUSHL   #1
          00000000'9F   04    FB  0D0A                    CALLS   #$$M, a#DS$ERRHARD
                             0D11  1934         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                             0D11  1935                 FMT_MBE_ER
00000000'EF   00000000'EF   DE  0D11                    MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  0D1C                    MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  0D23                    MOVAL   FMT_MBE_ER,REG_STRING
```

```
                51   0000'C8   D0   0D2E  1936          MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                     53   51   3C   0D33  1937          MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
                          21   13   0D36  1938          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                     55   53   D0   0D38  1939          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
           56   00000000'8F   D0   0D3B  1940          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                              0D42  1941          $DS_ERRHARD_S   #2,LUN,,-           ; PRINT ERROR
                              0D42  1942                          PRINT_SBE
                00000000'EF   DF   0D42                  PUSHAL  PRINT_SBE
                          00   DD   0D48                  PUSHL   #0
                00000000'EF   DD   0D4A                  PUSHL   LUN
                          02   DD   0D50                  PUSHL   #2
           00000000'9F    04   FB   0D52                  CALLS   #$$M, a#DS$ERRHARD
                              0D59  1943 30$:     $DS_CKLOOP      20$                 ; SCOPE LOOP?
      00000000'9F    FF30 CF   FA   0D59                  CALLG   20$, a#DS$CKLOOP
      0000'C2   00000000'8F   C8   0D62  1944 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
           5B   00000000'EF   DE   0D6B  1945          MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
            5A    5B    F7 8F   78   0D72  1946          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
           5A   00000000'8F   C8   0D77  1947          BISL    #VALID_BIT,R10          ; SET V BIT
                     69   5A   D0   0D7E  1948          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
           5B   00000000'8F   CA   0D81  1949          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
                0000'C2   5B   D0   0D88  1950          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
      0000'C2    00000200 8F   CE   0D8D  1951          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
      0000'C8   00000000'8F   D0   0D96  1952          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              0D9F  1953                                              ;       BLOCK SIZE = 256 W
                              0D9F  1954                                              ;       GAP SIZE = 5 US
                              0D9F  1955                                              ;       DATA RATE = 1 MW/S
      0000'C8   00000000'8F   D0   0D9F  1956          MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                              0DA8  1957          $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                          00   DD   0DA8                  PUSHL   #0
                00000064 8F   DD   0DAA                  PUSHL   #100
           00000000'9F    02   FB   0DB0                  CALLS   #2, a#DS$WAITUS
                              0DB7  1958          ERRPREP RHSR_MSG,2,-                ; PREPARE TO PRINT ERROR
                              0DB7  1959                  FMT_STATUS_REG
00000000'EF   00000000'EF   DE   0DB7                  MOVAL   RHSR_MSG,REG_NAME
           00000000'EF   02   9A   0DC2                  MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE   0DC9                  MOVAL   FMT_STATUS_REG,REG_STRING
                53   0000'C2   D0   0DD4  1960          MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
           53   00000000'8F   D3   0DD9  1961          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                          1C   13   0DE0  1962          BEQL    50$                     ; BRANCH IF NONE
                          54   D4   0DE2  1963          CLRL    R4                      ; EXPECT NO ERRORS
                     55   53   D0   0DE4  1964          MOVL    R3,R5                   ; REPORT FAILING BITS
                              0DE7  1965          $DS_ERRHARD_S   #3,LUN,,-           ; PRINT ERROR
                              0DE7  1966                          PRINT_SBE
                00000000'EF   DF   0DE7                  PUSHAL  PRINT_SBE
                          00   DD   0DED                  PUSHL   #0
                00000000'EF   DD   0DEF                  PUSHL   LUN
                          03   DD   0DF5                  PUSHL   #3
           00000000'9F    04   FB   0DF7                  CALLS   #$$M, a#DS$ERRHARD
                              0DFE  1967 50$:     $DS_CKLOOP      35$                 ; SCOPE LOOP
      00000000'9F    FF60 CF   FA   0DFE                  CALLG   35$, a#DS$CKLOOP
                              0E07  1968          ERRPREP MBE_REG_MSG,12,-            ; PREPARE TO PRINT ERROR
                              0E07  1969                  FMT_ANY_REG
00000000'EF   00000000'EF   DE   0E07                  MOVAL   MBE_REG_MSG,REG_NAME
           00000000'EF   0C   9A   0E12                  MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   0E19                  MOVAL   FMT_ANY_REG,REG_STRING
                          55   D4   0E24  1970          CLRL    R5                      ; CLEAR INDEX
                          59   D4   0E26  1971          CLRL    R9                      ; CLEAR INDEX
```

```
                54   000001FE'EF   B0   0E28  1972 60$:     MOVW    DATA_OBUFFER+510,R4      ; EXPECTED DATA
             53   00000000'EF49   B0   0E2F  1973           MOVW    DATA_IBUFFER R9 ,R3      ; RECEIVED DATA
                       54   53    B1   0E37  1974           CMPW    R3,R4                    ; EXPECTED=RECEIVED?
                            31    13   0E3A  1975           BEQL    70$                      ; BRANCH IF THEY ARE EQUAL
                            55    D6   0E3C  1976           INCL    R5                       ; ADD ONE TO ERROR COUNT
                       55   08    D1   0E3E  1977           CMPL    #8,R5                    ; PRINTED ERROR EIGHT TIMES?
                            2A    19   0E41  1978           BLSS    70$                      ; BRANCH IF SO
                56   000001FE'EF   DE   0E43  1979           MOVAL   DATA_OBUFFER+510,R6      ; GET ADDRESS OF EXP DATA BUFFER
             57   00000000'EF49   3E   0E4A  1980           MOVAW   DATA_IBUFFER R9 ,R7      ; GET ADDRESS OF REC DATA BUFFER
                                  0E52  1981           $DS_ERRHARD_S   #4,LUN,-          ; PRINT ERROR
                                  0E52  1982                           ARRAY_MSG15,-
                                  0E52  1983                           PRINT_DTERR
                      00000000'EF   DF   0E52                PUSHAL  PRINT_DTERR
                      00000000'EF   DF   0E58                PUSHAL  ARRAY_MSG15
                      00000000'EF   DD   0E5E                PUSHL   LUN
                            04      DD   0E64                PUSHL   #4
                00000000'9F   04    FB   0E66                CALLS   #$$M, a#DS$ERRHARD
                                  0E6D  1984 70$:     $DS_CKLOOP      20$              ; SCOPE LOOP?
             00000000'9F   FE1C CF   FA   0E6D                CALLG   20$, a#DS$CKLOOP
               AA 59   000000FF 8F   F3   0E76  1985           AOBLEQ  #255,R9,60$              ; CHECK ENTIRE BUFFER
                            55    D5   0E7E  1986           TSTL    R5                       ; ANY ERRORS REPORTED?
                            0F    13   0E80  1987           BEQL    80$                      ; BRANCH IF NONE
                                  0E82  1988           $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                            55    DD   0E82                PUSHL   R5
                      00000000'EF   9F   0E84                PUSHAB  FMT_ERRSUM
                00000000'9F   02    FB   0E8A                CALLS   #$$N, a#DS$PRINTB
                                  0E91  1989 80$:
                                  0E91  1990 $DS_ENDSUB
                                  0E91       T20_S10_X:
00000000'9F   00000180'EF   FA   0E91                CALLG   $S$, a#DS$ENDSUB
```

```
                                0E9C  1992 $DS_BGNSUB
                                0E9C       T20_S11::
00000000'9F   0000018C'EF   FA   0E9C                 CALLG   $S$, a#DS$BGNSUB
                                0EA7  1993 ;++
                                0EA7  1994 ; DATA TRANSFER: A55A
                                0EA7  1995 ;       SEE NOTE ON PREVIOUS SUBTEST
                                0EA7  1996 ;--
                                0EA7  1997         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                0EA7  1998                 FMT_MBE_SR
00000000'EF   00000000'EF   DE   0EA7                 MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A   0EB2                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   0EB9                 MOVAL   FMT_MBE_SR,REG_STRING
      52   00000000'EF      D0   0EC4  1999           MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
  0000'C2   00000090'8F      C8   0ECB  2000           BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      58   00000000'EF      D0   0ED4  2001           MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
   59 52   00000000'8F      C1   0EDB  2002           ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                  50        D4   0EE3  2003           CLRL    R0                      ; CLEAR INDEX
00000000'EF40   5AA5 8F      B0   0EE5  2004 10$:      MOVW    # X5AA5,DATA_OBUFFER R0 ; PUT COMPLIMENT PATTERN INTO BUFFER
        00000000'EF40        B4   0EEF  2005           CLRW    DATA_IBUFFER R0         ; CLEAR OUT READ BUFFER
   E7 50   000000FF 8F      F3   0EF6  2006           AOBLEQ  #255,R0,10$             ; FILL UP BOTH BUFFERS
000001FE'EF   A55A 8F      B0   0EFE  2007           MOVW    # XA55A,DATA_OBUFFER+510; PUT TEST PATTERN IN LAST LOCATION
  0000'C2   00000000'8F      C8   0F07  2008 20$:      BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      5B   00000000'EF      DE   0F10  2009           MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
       5A   5B   F7 8F      78   0F17  2010           ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
      5A   00000000'8F      C8   0F1C  2011           BISL    #VALID_BIT,R10          ; SET VALID BIT
                69   5A      D0   0F23  2012           MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
      5B   00000000'8F      CA   0F26  2013           BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
         0000'C2   5B      D0   0F2D  2014           MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                           0F32  2015                                                 ; (SELECTING MAP 0)
  0000'C2   00000200 8F      CE   0F32  2016           MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
  0000'C8   00000000'8F      D0   0F3B  2017           MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                           0F44  2018                                                 ;       BLOCK SIZE = 256 W
                           0F44  2019                                                 ;       GAP SIZE = 5 US
                           0F44  2020                                                 ;       DATA RATE = 1 MW/S
  0000'C8   00000000'8F      D0   0F44  2021           MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                           0F4D  2022           $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                   00      DD   0F4D                       PUSHL   #0
            00000064 8F      DD   0F4F                       PUSHL   #100
      00000000'9F   02      FB   0F55                       CALLS   #2, a#DS$WAITUS
          50   0000'C8      D0   0F5C  2023           MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
               53   50      3C   0F61  2024           MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
            6B 53   0E      E1   0F64  2025           BBC     #14,R3,30$              ; ERROR BIT SET?
      56   00000000'8F      D0   0F68  2026           MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                   54      D4   0F6F  2027           CLRL    R4                      ; EXPECTING NO ERRORS
               55   53      D0   0F71  2028           MOVL    R3,R5                   ; REPORT FAILING BIT
                           0F74  2029           $DS_ERRHARD_S   #1,LUN,,-           ; PRINT ERRROR
                           0F74  2030                           PRINT_SBE
            00000000'EF    DF   0F74                       PUSHAL  PRINT_SBE
                   00      DD   0F7A                       PUSHL   #0
            00000000'EF    DD   0F7C                       PUSHL   LUN
                   01      DD   0F82                       PUSHL   #1
      00000000'9F   04      FB   0F84                       CALLS   #$$M, a#DS$ERRHARD
                           0F8B  2031           ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                           0F8B  2032                   FMT_MBE_ER
00000000'EF   00000000'EF   DE   0F8B                       MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A   0F96                       MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   0F9D                       MOVAL   FMT_MBE_ER,REG_STRING
```

```
                51   0000'C8   D0   0FA8  2033          MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                     53   51   3C   0FAD  2034          MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
                          21   13   0FB0  2035          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                     55   53   D0   0FB2  2036          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
           56   00000000'8F   D0   0FB5  2037          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                              0FBC  2038          $DS_ERRHARD_S   #2,LUN,,-           ; PRINT ERROR
                              0FBC  2039                          PRINT_SBE
                00000000'EF   DF   0FBC                      PUSHAL  PRINT_SBE
                         00   DD   0FC2                      PUSHL   #0
                00000000'EF   DD   0FC4                      PUSHL   LUN
                         02   DD   0FCA                      PUSHL   #2
           00000000'9F   04   FB   0FCC                      CALLS   #$$M, a#DS$ERRHARD
                              0FD3  2040 30$:     $DS_CKLOOP      20$                 ; SCOPE LOOP?
      00000000'9F   FF30 CF   FA   0FD3                      CALLG   20$, a#DS$CKLOOP
      0000'C2   00000000'8F   C8   0FDC  2041 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
           5B   00000000'EF   DE   0FE5  2042          MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
            5A    5B   F7 8F   78   0FEC  2043          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
           5A   00000000'8F   C8   0FF1  2044          BISL    #VALID_BIT,R10          ; SET V BIT
                     69   5A   D0   0FF8  2045          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
           5B   00000000'8F   CA   0FFB  2046          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
                0000'C2   5B   D0   1002  2047          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
      0000'C2   00000200 8F   CE   1007  2048          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
      0000'C8   00000000'8F   D0   1010  2049          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              1019  2050                                              ;       BLOCK SIZE = 256 W
                              1019  2051                                              ;       GAP SIZE = 5 US
                              1019  2052                                              ;       DATA RATE = 1 MW/S
      0000'C8   00000000'8F   D0   1019  2053          MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                              1022  2054          $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                         00   DD   1022                      PUSHL   #0
                00000064 8F   DD   1024                      PUSHL   #100
           00000000'9F   02   FB   102A                      CALLS   #2, a#DS$WAITUS
                              1031  2055          ERRPREP RHSR_MSG,2,-                ; PREPARE TO PRINT ERROR
                              1031  2056                  FMT_STATUS_REG
      00000000'EF   00000000'EF   DE   1031                      MOVAL   RHSR_MSG,REG_NAME
           00000000'EF   02   9A   103C                      MOVZBL  #2,REG_NO
      00000000'EF   00000000'EF   DE   1043                      MOVAL   FMT_STATUS_REG,REG_STRING
                53   0000'C2   D0   104E  2057          MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
           53   00000000'8F   D3   1053  2058          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                         1C   13   105A  2059          BEQL    50$                     ; BRANCH IF NONE
                         54   D4   105C  2060          CLRL    R4                      ; EXPECT NO ERRORS
                     55   53   D0   105E  2061          MOVL    R3,R5                   ; REPORT FAILING BITS
                              1061  2062          $DS_ERRHARD_S   #3,LUN,,-           ; PRINT ERROR
                              1061  2063                          PRINT_SBE
                00000000'EF   DF   1061                      PUSHAL  PRINT_SBE
                         00   DD   1067                      PUSHL   #0
                00000000'EF   DD   1069                      PUSHL   LUN
                         03   DD   106F                      PUSHL   #3
           00000000'9F   04   FB   1071                      CALLS   #$$M, a#DS$ERRHARD
                              1078  2064 50$:     $DS_CKLOOP      35$                 ; SCOPE LOOP
      00000000'9F   FF60 CF   FA   1078                      CALLG   35$, a#DS$CKLOOP
                              1081  2065          ERRPREP MBE_REG_MSG,12,-            ; PREPARE TO PRINT ERROR
                              1081  2066                  FMT_ANY_REG
      00000000'EF   00000000'EF   DE   1081                      MOVAL   MBE_REG_MSG,REG_NAME
           00000000'EF   0C   9A   108C                      MOVZBL  #12,REG_NO
      00000000'EF   00000000'EF   DE   1093                      MOVAL   FMT_ANY_REG,REG_STRING
                         55   D4   109E  2067          CLRL    R5                      ; CLEAR INDEX
                         59   D4   10A0  2068          CLRL    R9                      ; CLEAR INDEX
```

```
                54   000001FE'EF   B0   10A2  2069 60$:    MOVW    DATA_OBUFFER+510,R4      ; EXPECTED DATA
              53   00000000'EF49   B0   10A9  2070         MOVW    DATA_IBUFFER R9 ,R3      ; RECEIVED DATA
                        54   53    B1   10B1  2071         CMPW    R3,R4                    ; EXPECTED=RECEIVED?
                             31    13   10B4  2072         BEQL    70$                      ; BRANCH IF THEY ARE EQUAL
                             55    D6   10B6  2073         INCL    R5                       ; ADD ONE TO ERROR COUNT
                        55   08    D1   10B8  2074         CMPL    #8,R5                    ; PRINTED ERROR EIGHT TIMES?
                             2A    19   10BB  2075         BLSS    70$                      ; BRANCH IF SO
                56   000001FE'EF   DE   10BD  2076         MOVAL   DATA_OBUFFER+510,R6      ; GET ADDRESS OF EXP DATA BUFFER
              57   00000000'EF49   3E   10C4  2077         MOVAW   DATA_IBUFFER R9 ,R7      ; GET ADDRESS OF REC DATA BUFFER
                                   10CC  2078         $DS_ERRHARD_S   #4,LUN,-              ; PRINT ERROR
                                   10CC  2079                         ARRAY_MSG15,-
                                   10CC  2080                         PRINT_DTERR
                        00000000'EF   DF   10CC                PUSHAL  PRINT_DTERR
                        00000000'EF   DF   10D2                PUSHAL  ARRAY_MSG15
                        00000000'EF   DD   10D8                PUSHL   LUN
                             04       DD   10DE                PUSHL   #4
                   00000000'9F   04   FB   10E0                CALLS   #$$M, a#DS$ERRHARD
                                   10E7  2081 70$:    $DS_CKLOOP      20$                  ; SCOPE LOOP?
          00000000'9F   FE1C CF   FA   10E7                     CALLG   20$, a#DS$CKLOOP
            AA 59   000000FF 8F   F3   10F0  2082         AOBLEQ  #255,R9,60$              ; CHECK ENTIRE BUFFER
                             55   D5   10F8  2083         TSTL    R5                       ; ANY ERRORS REPORTED?
                             0F   13   10FA  2084         BEQL    80$                      ; BRANCH IF NONE
                                   10FC  2085         $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                             55   DD   10FC                PUSHL   R5
                        00000000'EF   9F   10FE                PUSHAB  FMT_ERRSUM
                   00000000'9F   02   FB   1104                CALLS   #$$N, a#DS$PRINTB
                                   110B  2086 80$:
                                   110B  2087 $DS_ENDSUB
                                   110B       T20_S11_X:
          00000000'9F   0000018C'EF   FA   110B                CALLG   $S$, a#DS$ENDSUB
```

```
                                1116  2089 $DS_BGNSUB
                                1116       T20_S12::
00000000'9F   00000198'EF   FA  1116                 CALLG   $S$, a#DS$BGNSUB
                                1121  2090 ;++
                                1121  2091 ; DATA TRANSFER: AA55 PATTERN
                                1121  2092 ;--
                                1121  2093         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                1121  2094                 FMT_MBE_SR
00000000'EF   00000000'EF   DE  1121                 MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF   0C    9A  112C                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  1133                 MOVAL   FMT_MBE_SR,REG_STRING
        52    00000000'EF   D0  113E  2095         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
    0000'C2   00000000'8F   C8  1145  2096         BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        58    00000000'EF   D0  114E  2097         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
    59  52    00000000'8F   C1  1155  2098         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                    50    D4  115D  2099         CLRL    R0                      ; CLEAR INDEX
  00000000'EF40   55AA 8F   B0  115F  2100 10$:    MOVW    # X55AA,DATA_OBUFFER R0 ; PUT COMPLIMENT PATTERN INTO BUFFER
          00000000'EF40   B4  1169  2101         CLRW    DATA_IBUFFER R0         ; CLEAR OUT READ BUFFER
      E7 50   000000FF 8F   F3  1170  2102         AOBLEQ  #255,R0,10$             ; FILL UP BOTH BUFFERS
    000001FE'EF   AA55 8F   B0  1178  2103         MOVW    # XAA55,DATA_OBUFFER+510; PUT EXPECTED PATTERN IN BUFFER
    0000'C2   00000000'8F   C8  1181  2104 20$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        5B    00000000'EF   DE  118A  2105         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
         5A   5B    F7 8F   78  1191  2106         ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
        5A    00000000'8F   C8  1196  2107         BISL    #VALID_BIT,R10          ; SET VALID BIT
                  69    5A   D0  119D  2108         MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
        5B    00000000'8F   CA  11A0  2109         BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
            0000'C2   5B    D0  11A7  2110         MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                              11AC  2111                                         ; (SELECTING MAP 0)
    0000'C2   00000200 8F   CE  11AC  2112         MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
    0000'C8   00000000'8F   D0  11B5  2113         MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              11BE  2114                                         ;       BLOCK SIZE = 256 W
                              11BE  2115                                         ;       GAP SIZE = 5 US
                              11BE  2116                                         ;       DATA RATE = 1 MW/S
    0000'C8   00000000'8F   D0  11BE  2117         MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                              11C7  2118         $DS_WAITUS_S    #100            ; WAIT 1 MILLISECONDS
                    00    DD  11C7                         PUSHL   #0
            00000064 8F   DD  11C9                         PUSHL   #100
      00000000'9F   02    FB  11CF                         CALLS   #2, a#DS$WAITUS
          50    0000'C8   D0  11D6  2119         MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
              53    50    3C  11DB  2120         MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
           6B 53    0E    E1  11DE  2121         BBC     #14,R3,30$              ; ERROR BIT SET?
      56   00000000'8F   D0  11E2  2122         MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                    54    D4  11E9  2123         CLRL    R4                      ; EXPECTING NO ERRORS
                55  53    D0  11EB  2124         MOVL    R3,R5                   ; REPORT FAILING BIT
                              11EE  2125         $DS_ERRHARD_S   #1,LUN,,-       ; PRINT ERROR
                              11EE  2126                         PRINT_SBE
            00000000'EF   DF  11EE                         PUSHAL  PRINT_SBE
                    00    DD  11F4                         PUSHL   #0
            00000000'EF   DD  11F6                         PUSHL   LUN
                    01    DD  11FC                         PUSHL   #1
      00000000'9F   04    FB  11FE                         CALLS   #$$M, a#DS$ERRHARD
                              1205  2127         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                              1205  2128                 FMT_MBE_ER
00000000'EF   00000000'EF   DE  1205                 MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF   0C    9A  1210                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  1217                 MOVAL   FMT_MBE_ER,REG_STRING
          51    0000'C8   D0  1222  2129         MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
```

```
                  53   51   3C  1227  2130          MOVZWL  R1,R3                      ; EXTRACT REGISTER CONTENTS
                       21   13  122A  2131          BEQL    30$                        ; IF NO ERROR BITS SET, SKIP
                  55   53   D0  122C  2132          MOVL    R3,R5                      ; REPORT FAILING BIT(S)
        56   00000000'8F   D0  122F  2133          MOVL    #MBE_ER,R6                 ; FAILING REGISTER
                                 1236  2134          $DS_ERRHARD_S   #2,LUN,,-          ; PRINT ERROR
                                 1236  2135                          PRINT_SBE
             00000000'EF   DF  1236                          PUSHAL  PRINT_SBE
                       00   DD  123C                          PUSHL   #0
             00000000'EF   DD  123E                          PUSHL   LUN
                       02   DD  1244                          PUSHL   #2
        00000000'9F    04   FB  1246                          CALLS   #$$M, a#DS$ERRHARD
                                 124D  2136 30$:     $DS_CKLOOP      20$                ; SCOPE LOOP?
   00000000'9F   FF30 CF   FA  124D                          CALLG   20$, a#DS$CKLOOP
   0000'C2   00000000'8F   C8  1256  2137 35$:     BISL    #PGM_INIT,CR(R2)           ; INITIALIZE MUT
        5B   00000000'EF   DE  125F  2138          MOVAL   DATA_IBUFFER,R11           ; GET ADDRESS OF READ BUFFER
         5A   5B   F7 8F   78  1266  2139          ASHL    #-9,R11,R10                ; PUT PFN INTO R10
        5A   00000000'8F   C8  126B  2140          BISL    #VALID_BIT,R10             ; SET V BIT
                  69   5A   D0  1272  2141          MOVL    R10,(R9)                   ; PUT PFN INTO MAP 0
        5B   00000000'8F   CA  1275  2142          BICL    #MAP_PTR_MSK,R11           ; ISOLATE BYTE OFFSET
             0000'C2   5B   D0  127C  2143          MOVL    R11,VAR(R2)                ; WRITE VAR,SELECTING MAP 0
   0000'C2   00000200 8F   CE  1281  2144          MNEGL   #512,BCR(R2)               ; SET UP RH750 BYTE COUNT
   0000'C8   00000000'8F   D0  128A  2145          MOVL    #MBE_PARAM,MBE_CR2(R8)     ; SET MBE PARAMETERS:
                                 1293  2146                                             ;       BLOCK SIZE = 256 W
                                 1293  2147                                             ;       GAP SIZE = 5 US
                                 1293  2148                                             ;       DATA RATE = 1 MW/S
   0000'C8   00000000'8F   D0  1293  2149          MOVL    #READ,MBE_CR1(R8)          ; ISSUE READ DATA COMMAND
                                 129C  2150          $DS_WAITUS_S    #100               ; WAIT 1 MILLISECONDS
                       00   DD  129C                          PUSHL   #0
             00000064 8F   DD  129E                          PUSHL   #100
        00000000'9F    02   FB  12A4                          CALLS   #2, a#DS$WAITUS
                                 12AB  2151          ERRPREP RHSR_MSG,2,-               ; PREPARE TO PRINT ERROR
                                 12AB  2152                  FMT_STATUS_REG
00000000'EF   00000000'EF   DE  12AB                          MOVAL   RHSR_MSG,REG_NAME
        00000000'EF    02   9A  12B6                          MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  12BD                          MOVAL   FMT_STATUS_REG,REG_STRING
             53   0000'C2   D0  12C8  2153          MOVL    SR(R2),R3                  ; READ RH750 STATUS REGISTER
        53   00000000'8F   D3  12CD  2154          BITL    #RHSR_CHECK,R3             ; ANY STATUS ERRORS?
                       1C   13  12D4  2155          BEQL    50$                        ; BRANCH IF NONE
                       54   D4  12D6  2156          CLRL    R4                         ; EXPECT NO ERRORS
                  55   53   D0  12D8  2157          MOVL    R3,R5                      ; REPORT FAILING BITS
                                 12DB  2158          $DS_ERRHARD_S   #3,LUN,,-          ; PRINT ERROR
                                 12DB  2159                          PRINT_SBE
             00000000'EF   DF  12DB                          PUSHAL  PRINT_SBE
                       00   DD  12E1                          PUSHL   #0
             00000000'EF   DD  12E3                          PUSHL   LUN
                       03   DD  12E9                          PUSHL   #3
        00000000'9F    04   FB  12EB                          CALLS   #$$M, a#DS$ERRHARD
                                 12F2  2160 50$:     $DS_CKLOOP      35$                ; SCOPE LOOP
   00000000'9F   FF60 CF   FA  12F2                          CALLG   35$, a#DS$CKLOOP
                                 12FB  2161          ERRPREP MBE_REG_MSG,12,-           ; PREPARE TO PRINT ERROR
                                 12FB  2162                  FMT_ANY_REG
00000000'EF   00000000'EF   DE  12FB                          MOVAL   MBE_REG_MSG,REG_NAME
        00000000'EF    0C   9A  1306                          MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  130D                          MOVAL   FMT_ANY_REG,REG_STRING
                       55   D4  1318  2163          CLRL    R5                         ; CLEAR INDEX
                       59   D4  131A  2164          CLRL    R9                         ; CLEAR INDEX
        54   000001FE'EF   B0  131C  2165 60$:     MOVW    DATA_OBUFFER+510,R4        ; EXPECTED DATA
```

```
                53   00000000'EF49   B0  1323  2166          MOVW    DATA_IBUFFER R9 ,R3     ; RECEIVED DATA
                          54   53    B1  132B  2167          CMPW    R3,R4                   ; EXPECTED=RECEIVED?
                               31    13  132E  2168          BEQL    70$                     ; BRANCH IF THEY ARE EQUAL
                               55    D6  1330  2169          INCL    R5                      ; ADD ONE TO ERROR COUNT
                          55   08    D1  1332  2170          CMPL    #8,R5                   ; PRINTED ERROR EIGHT TIMES?
                               2A    19  1335  2171          BLSS    70$                     ; BRANCH IF SO
                  56   000001FE'EF   DE  1337  2172          MOVAL   DATA_OBUFFER+510,R6     ; GET ADDRESS OF EXP DATA BUFFER
                57   00000000'EF49   3E  133E  2173          MOVAW   DATA_IBUFFER R9 ,R7     ; GET ADDRESS OF REC DATA BUFFER
                                         1346  2174          $DS_ERRHARD_S   #4,LUN,-        ; PRINT ERROR
                                         1346  2175                          ARRAY_MSG15,-
                                         1346  2176                          PRINT_DTERR
                     00000000'EF    DF  1346                      PUSHAL  PRINT_DTERR
                     00000000'EF    DF  134C                      PUSHAL  ARRAY_MSG15
                     00000000'EF    DD  1352                      PUSHL   LUN
                               04   DD  1358                      PUSHL   #4
                00000000'9F    04    FB  135A                      CALLS   #$$M, a#DS$ERRHARD
                                         1361  2177 70$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
              00000000'9F   FE1C CF    FA  1361                      CALLG   20$, a#DS$CKLOOP
                AA 59   000000FF 8F    F3  136A  2178          AOBLEQ  #255,R9,60$             ; CHECK ENTIRE BUFFER
                               55    D5  1372  2179          TSTL    R5                      ; ANY ERRORS REPORTED?
                               0F    13  1374  2180          BEQL    80$                     ; BRANCH IF NONE
                                         1376  2181          $DS_PRINTB_S    FMT_ERRSUM,R5   ; REPORT NUMBER OF ERRORS
                               55    DD  1376                      PUSHL   R5
                     00000000'EF    9F  1378                      PUSHAB  FMT_ERRSUM
                00000000'9F    02    FB  137E                      CALLS   #$$N, a#DS$PRINTB
                                         1385  2182 80$:
                                         1385  2183 $DS_ENDSUB
                                         1385       T20_S12_X:
00000000'9F   00000198'EF    FA  1385                      CALLG   $S$, a#DS$ENDSUB
```

```
                                1390  2185 $DS_BGNSUB
                                1390       T20_S13::
00000000'9F   000001A4'EF   FA  1390                 CALLG   $S$, a#DS$BGNSUB
                                139B  2186 ;++
                                139B  2187 ; DATA TRANSFER: FLOATING ONE PATTERN
                                139B  2188 ;--
                                139B  2189       ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                139B  2190               FMT_MBE_SR
00000000'EF   00000000'EF   DE  139B                     MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  13A6                     MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  13AD                     MOVAL   FMT_MBE_SR,REG_STRING
      52   00000000'EF      D0  13B8  2191       MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
  0000'C2   00000000'8F      C8  13BF  2192       BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      58   00000000'EF      D0  13C8  2193       MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
  59  52   00000000'8F      C1  13CF  2194       ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                   50       D4  13D7  2195       CLRL    R0                      ; CLEAR INDEX
      51   00010001 8F      D0  13D9  2196       MOVL    # X10001,R1             ; INITIALIZE PATTERN
   00000000'EF40   51      B0  13E0  2197 10$:   MOVW    R1,DATA_OBUFFER R0      ; PUT PATTERN IN OUTPUT BUFFER
   00000000'EF40   51      B2  13E8  2198       MCOMW   R1,DATA_IBUFFER R0      ; PUT COMPLIMENT PATTERN IN READ BUFFER
         51   51   01      9C  13F0  2199       ROTL    #1,R1,R1                ; FLOAT PATTERN LEFT ONCE
    E4 50   000000FF 8F      F3  13F4  2200       AOBLEQ  #255,R0,10$             ; FILL UP BOTH BUFFERS
  0000'C2   00000000'8F      C8  13FC  2201 20$:   BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      5B   00000000'EF      DE  1405  2202       MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
       5A   5B   F7 8F      78  140C  2203       ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
      5A   00000000'8F      C8  1411  2204       BISL    #VALID_BIT,R10          ; SET VALID BIT
               69   5A      D0  1418  2205       MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
      5B   00000000'8F      CA  141B  2206       BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
         0000'C2   5B      D0  1422  2207       MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                           1427  2208                                           ; (SELECTING MAP 0)
  0000'C2   00000200 8F      CE  1427  2209       MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
  0000'C8   00000000'8F      D0  1430  2210       MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                           1439  2211                                           ;       BLOCK SIZE = 256 W
                           1439  2212                                           ;       GAP SIZE = 5 US
                           1439  2213                                           ;       DATA RATE = 1 MW/S
  0000'C8   00000000'8F      D0  1439  2214       MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                           1442  2215       $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                      00   DD  1442                 PUSHL   #0
               00000064 8F   DD  1444                 PUSHL   #100
      00000000'9F    02   FB  144A                 CALLS   #2, a#DS$WAITUS
          50   0000'C8      D0  1451  2216       MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
               53   50      3C  1456  2217       MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
            6B 53   0E      E1  1459  2218       BBC     #14,R3,30$              ; ERROR BIT SET?
      56   00000000'8F      D0  145D  2219       MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                   54      D4  1464  2220       CLRL    R4                      ; EXPECTING NO ERRORS
               55   53      D0  1466  2221       MOVL    R3,R5                   ; REPORT FAILING BIT
                           1469  2222       $DS_ERRHARD_S   #1,LUN,,-           ; PRINT ERROR
                           1469  2223                       PRINT_SBE
               00000000'EF   DF  1469                 PUSHAL  PRINT_SBE
                      00   DD  146F                 PUSHL   #0
               00000000'EF   DD  1471                 PUSHL   LUN
                      01   DD  1477                 PUSHL   #1
      00000000'9F    04   FB  1479                 CALLS   #$$M, a#DS$ERRHARD
                           1480  2224       ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                           1480  2225               FMT_MBE_ER
00000000'EF   00000000'EF   DE  1480                 MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  148B                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  1492                 MOVAL   FMT_MBE_ER,REG_STRING
```

```
                51   0000'C8   D0  149D  2226          MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                     53   51   3C  14A2  2227          MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
                          21   13  14A5  2228          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                     55   53   D0  14A7  2229          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
           56   00000000'8F   D0  14AA  2230          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                              14B1  2231          $DS_ERRHARD_S   #2,LUN,,-           ; PRINT ERROR
                              14B1  2232                          PRINT_SBE
                00000000'EF   DF  14B1                          PUSHAL  PRINT_SBE
                         00   DD  14B7                          PUSHL   #0
                00000000'EF   DD  14B9                          PUSHL   LUN
                         02   DD  14BF                          PUSHL   #2
           00000000'9F   04   FB  14C1                          CALLS   #$$M, a#DS$ERRHARD
                              14C8  2233 30$:     $DS_CKLOOP      20$                 ; SCOPE LOOP?
      00000000'9F   FF30 CF   FA  14C8                          CALLG   20$, a#DS$CKLOOP
      0000'C2   00000000'8F   C8  14D1  2234 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
           5B   00000000'EF   DE  14DA  2235          MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
            5A    5B   F7 8F   78  14E1  2236          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
           5A   00000000'8F   C8  14E6  2237          BISL    #VALID_BIT,R10          ; SET V BIT
                     69   5A   D0  14ED  2238          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
           5B   00000000'8F   CA  14F0  2239          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
                0000'C2   5B   D0  14F7  2240          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
      0000'C2   00000200 8F   CE  14FC  2241          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
                0000'C8   01   D0  1505  2242          MOVL    #1,MBE_DBR(R8)          ; PUT PATTERN IN DATA BUFFER
      0000'C8   00000000'8F   D0  150A  2243          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                              1513  2244                                              ; BLOCK SIZE = 256 W
                              1513  2245                                              ; GAP SIZE = 5 US
                              1513  2246                                              ; DATA RATE = 1 MW/S
      0000'C8   00000000'8F   D0  1513  2247          MOVL    #ENABLE_PS,MBE_MR(R8)   ; ENABLE PATTERN SHIFT
      0000'C8   00000000'8F   D0  151C  2248          MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                              1525  2249          $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                         00   DD  1525                          PUSHL   #0
                00000064 8F   DD  1527                          PUSHL   #100
           00000000'9F   02   FB  152D                          CALLS   #2, a#DS$WAITUS
                              1534  2250          ERRPREP RHSR_MSG,2,-                ; PREPARE TO PRINT ERROR
                              1534  2251                  FMT_STATUS_REG
00000000'EF   00000000'EF   DE  1534                          MOVAL   RHSR_MSG,REG_NAME
      00000000'EF   02   9A  153F                          MOVZBL  #2,REG_NO
00000000'EF   00000000'EF   DE  1546                          MOVAL   FMT_STATUS_REG,REG_STRING
           53   0000'C2   D0  1551  2252          MOVL    SR(R2),R3                   ; READ RH750 STATUS REGISTER
           53   00000000'8F   D3  1556  2253          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                         1C   13  155D  2254          BEQL    50$                     ; BRANCH IF NONE
                         54   D4  155F  2255          CLRL    R4                      ; EXPECT NO ERRORS
                     55   53   D0  1561  2256          MOVL    R3,R5                   ; REPORT FAILING BITS
                              1564  2257          $DS_ERRHARD_S   #3,LUN,,-           ; PRINT ERROR
                              1564  2258                          PRINT_SBE
                00000000'EF   DF  1564                          PUSHAL  PRINT_SBE
                         00   DD  156A                          PUSHL   #0
                00000000'EF   DD  156C                          PUSHL   LUN
                         03   DD  1572                          PUSHL   #3
           00000000'9F   04   FB  1574                          CALLS   #$$M, a#DS$ERRHARD
                              157B  2259 50$:     $DS_CKLOOP      35$                 ; SCOPE LOOP
      00000000'9F   FF52 CF   FA  157B                          CALLG   35$, a#DS$CKLOOP
                         55   D4  1584  2260          CLRL    R5                      ; CLEAR INDEX
                         59   D4  1586  2261          CLRL    R9                      ; CLEAR INDEX
        54   00000000'EF49   90  1588  2262 60$:     MOVB    DATA_OBUFFER R9 ,R4     ; EXPECTED DATA
        53   00000000'EF49   90  1590  2263          MOVB    DATA_IBUFFER R9 ,R3     ; RECEIVED DATA
                     54   53   91  1598  2264          CMPB    R3,R4                   ; EXPECTED=RECEIVED?
```

```
                    32   13  159B  2265          BEQL    70$                      ; BRANCH IF THEY ARE EQUAL
                    55   D6  159D  2266          INCL    R5                       ; ADD ONE TO ERROR COUNT
                55  08   D1  159F  2267          CMPL    #8,R5                    ; PRINTED ERROR EIGHT TIMES?
                    2B   19  15A2  2268          BLSS    70$                      ; BRANCH IF SO
        56  00000000'EF49   9E  15A4  2269          MOVAB   DATA_OBUFFER R9 ,R6      ; GET ADDRESS OF EXP DATA BUFFER
        57  00000000'EF49   9E  15AC  2270          MOVAB   DATA_IBUFFER R9 ,R7      ; GET ADDRESS OF REC DATA BUFFER
                                15B4  2271          $DS_ERRHARD_S   #4,LUN,-         ; PRINT ERROR
                                15B4  2272                          ARRAY_MSG15,-
                                15B4  2273                          PRINT_DTERR
            00000000'EF   DF  15B4                          PUSHAL  PRINT_DTERR
            00000000'EF   DF  15BA                          PUSHAL  ARRAY_MSG15
            00000000'EF   DD  15C0                          PUSHL   LUN
                    04   DD  15C6                          PUSHL   #4
        00000000'9F   04   FB  15C8                          CALLS   #$$M, a#DS$ERRHARD
                                15CF  2274 70$:     $DS_CKLOOP      20$              ; SCOPE LOOP?
    00000000'9F   FE29 CF   FA  15CF                          CALLG   20$, a#DS$CKLOOP
      A8 59   000001FF 8F   F3  15D8  2275          AOBLEQ  #511,R9,60$              ; CHECK ENTIRE BUFFER
                    55   D5  15E0  2276          TSTL    R5                       ; ANY ERRORS REPORTED?
                    0F   13  15E2  2277          BEQL    80$                      ; BRANCH IF NONE
                                15E4  2278          $DS_PRINTB_S    FMT_ERRSUM,R5    ; REPORT NUMBER OF ERRORS
                    55   DD  15E4                          PUSHL   R5
            00000000'EF   9F  15E6                          PUSHAB  FMT_ERRSUM
        00000000'9F   02   FB  15EC                          CALLS   #$$N, a#DS$PRINTB
                                15F3  2279 80$:
                                15F3  2280 $DS_ENDSUB
                                15F3       T20_S13_X:
  00000000'9F   000001A4'EF   FA  15F3                          CALLG   $S$, a#DS$ENDSUB
```

```
                                15FE  2282 $DS_BGNSUB
                                15FE       T20_S14::
00000000'9F   000001B0'EF   FA   15FE                    CALLG   $S$, a#DS$BGNSUB
                                1609  2283 ;++
                                1609  2284 ; DATA TRANSFER: FLOATING ZERO PATTERN
                                1609  2285 ;--
                                1609  2286         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                1609  2287                 FMT_MBE_SR
00000000'EF   00000000'EF   DE   1609                    MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A   1614                    MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   161B                    MOVAL   FMT_MBE_SR,REG_STRING
      52      00000000'EF   D0   1626  2288         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
  0000'C2     00000000'8F   C8   162D  2289         BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      58      00000000'EF   D0   1636  2290         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
   59 52      00000000'8F   C1   163D  2291         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                        50   D4   1645  2292         CLRL    R0                      ; CLEAR INDEX
      51      FFFEFFFE 8F   D0   1647  2293         MOVL    # XFFFEFFFE,R1         ; INITIALIZE PATTERN
   00000000'EF40      51   B0   164E  2294 10$:     MOVW    R1,DATA_OBUFFER R0      ; PUT PATTERN INTO BUFFER
   00000000'EF40      51   B2   1656  2295         MCOMW   R1,DATA_IBUFFER R0      ; PUT COMPLIMENT PATTERN IN READ BUFFER
         51   51      01   9C   165E  2296         ROTL    #1,R1,R1                ; FLOAT PATTERN LEFT ONCE
   E4 50      000000FF 8F   F3   1662  2297         AOBLEQ  #255,R0,10$             ; FILL UP BOTH BUFFERS
  0000'C2     00000000'8F   C8   166A  2298 20$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      5B      00000000'EF   DE   1673  2299         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
       5A     5B    F7 8F   78   167A  2300         ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
      5A      00000000'8F   C8   167F  2301         BISL    #VALID_BIT,R10          ; SET VALID BIT
                 69   5A   D0   1686  2302         MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
      5B      00000000'8F   CA   1689  2303         BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
           0000'C2   5B   D0   1690  2304         MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                             1695  2305                                             ; (SELECTING MAP 0)
  0000'C2     00000200 8F   CE   1695  2306         MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
  0000'C8     00000000'8F   D0   169E  2307         MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                             16A7  2308                                             ;       BLOCK SIZE = 256 W
                             16A7  2309                                             ;       GAP SIZE = 5 US
                             16A7  2310                                             ;       DATA RATE = 1 MW/S
  0000'C8     00000000'8F   D0   16A7  2311         MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                             16B0  2312         $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                        00   DD   16B0                    PUSHL   #0
                00000064 8F   DD   16B2                    PUSHL   #100
      00000000'9F       02   FB   16B8                    CALLS   #2, a#DS$WAITUS
           50   0000'C8   D0   16BF  2313         MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
                53   50   3C   16C4  2314         MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
             6B 53      0E   E1   16C7  2315         BBC     #14,R3,30$              ; ERROR BIT SET?
      56     00000000'8F   D0   16CB  2316         MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                        54   D4   16D2  2317         CLRL    R4                      ; EXPECTING NO ERRORS
                55      53   D0   16D4  2318         MOVL    R3,R5                   ; REPORT FAILING BIT
                             16D7  2319         $DS_ERRHARD_S   #1,LUN,,-           ; PRINT ERROR
                             16D7  2320                         PRINT_SBE
                00000000'EF   DF   16D7                    PUSHAL  PRINT_SBE
                        00   DD   16DD                    PUSHL   #0
                00000000'EF   DD   16DF                    PUSHL   LUN
                        01   DD   16E5                    PUSHL   #1
      00000000'9F       04   FB   16E7                    CALLS   #$$M, a#DS$ERRHARD
                             16EE  2321         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                             16EE  2322                 FMT_MBE_ER
00000000'EF   00000000'EF   DE   16EE                    MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A   16F9                    MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE   1700                    MOVAL   FMT_MBE_ER,REG_STRING
```

```
                51   0000'C8    D0  170B  2323          MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
                53     51       3C  1710  2324          MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
                       21       13  1713  2325          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                55     53       D0  1715  2326          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
           56   00000000'8F     D0  1718  2327          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                               171F  2328          $DS_ERRHARD_S   #2,LUN,,-           ; PRINT ERROR
                               171F  2329                          PRINT_SBE
                00000000'EF    DF  171F                        PUSHAL  PRINT_SBE
                         00    DD  1725                        PUSHL   #0
                00000000'EF    DD  1727                        PUSHL   LUN
                         02    DD  172D                        PUSHL   #2
           00000000'9F   04    FB  172F                        CALLS   #$$M, a#DS$ERRHARD
                               1736  2330 30$:     $DS_CKLOOP      20$                 ; SCOPE LOOP?
      00000000'9F   FF30 CF    FA  1736                        CALLG   20$, a#DS$CKLOOP
      0000'C2   00000000'8F    C8  173F  2331 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
           5B   00000000'EF    DE  1748  2332          MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF READ BUFFER
            5A    5B    F7 8F  78  174F  2333          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
           5A   00000000'8F    C8  1754  2334          BISL    #VALID_BIT,R10          ; SET V BIT
                     69   5A    D0  175B  2335          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
           5B   00000000'8F    CA  175E  2336          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
               0000'C2   5B    D0  1765  2337          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
      0000'C2   00000200 8F    CE  176A  2338          MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT
      0000'C8   0000FFFE 8F    D0  1773  2339          MOVL    # XFFFE,MBE_DBR(R8)    ; PUT PATTERN IN DATA BUFFER
      0000'C8   00000000'8F    D0  177C  2340          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                               1785  2341                                              ; BLOCK SIZE = 256 W
                               1785  2342                                              ; GAP SIZE = 5 US
                               1785  2343                                              ; DATA RATE = 1 MW/S
      0000'C8   00000000'8F    D0  1785  2344          MOVL    #ENABLE_PS,MBE_MR(R8)   ; ENABLE PATTERN SHIFT
      0000'C8   00000000'8F    D0  178E  2345          MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ DATA COMMAND
                               1797  2346          $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                         00    DD  1797                        PUSHL   #0
                00000064 8F    DD  1799                        PUSHL   #100
           00000000'9F   02    FB  179F                        CALLS   #2, a#DS$WAITUS
                               17A6  2347          ERRPREP RHSR_MSG,2,FMT_STATUS_REG       ; PREPARE TO HANDLE ERROR
00000000'EF    00000000'EF     DE  17A6                        MOVAL   RHSR_MSG,REG_NAME
      00000000'EF    02       9A  17B1                        MOVZBL  #2,REG_NO
00000000'EF    00000000'EF     DE  17B8                        MOVAL   FMT_STATUS_REG,REG_STRING
           53   0000'C2        D0  17C3  2348          MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
           53   00000000'8F    D3  17C8  2349          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                         1C    13  17CF  2350          BEQL    50$                     ; BRANCH IF NONE
                         54    D4  17D1  2351          CLRL    R4                      ; EXPECT NO ERRORS
                55     53      D0  17D3  2352          MOVL    R3,R5                   ; REPORT FAILING BITS
                               17D6  2353          $DS_ERRHARD_S   #3,LUN,,-           ; PRINT ERROR
                               17D6  2354                          PRINT_SBE
                00000000'EF    DF  17D6                        PUSHAL  PRINT_SBE
                         00    DD  17DC                        PUSHL   #0
                00000000'EF    DD  17DE                        PUSHL   LUN
                         03    DD  17E4                        PUSHL   #3
           00000000'9F   04    FB  17E6                        CALLS   #$$M, a#DS$ERRHARD
                               17ED  2355 50$:     $DS_CKLOOP      35$                 ; SCOPE LOOP
      00000000'9F   FF4E CF    FA  17ED                        CALLG   35$, a#DS$CKLOOP
                         55    D4  17F6  2356          CLRL    R5                      ; CLEAR INDEX
                         59    D4  17F8  2357          CLRL    R9                      ; CLEAR INDEX
        54   00000000'EF49     90  17FA  2358 60$:     MOVB    DATA_OBUFFER R9 ,R4     ; EXPECTED DATA
        53   00000000'EF49     90  1802  2359          MOVB    DATA_IBUFFER R9 ,R3     ; RECEIVED DATA
                55     54      91  180A  2360          CMPB    R3,R4                   ; EXPECTED=RECEIVED?
                         32    13  180D  2361          BEQL    70$                     ; BRANCH IF THEY ARE EQUAL
```

```
                      55   D6  180F  2362          INCL    R5                      ; ADD ONE TO ERROR COUNT
                   55  08   D1  1811  2363          CMPL    #8,R5                   ; PRINTED ERROR EIGHT TIMES?
                       2B   19  1814  2364          BLSS    70$                     ; BRANCH IF SO
          56   00000000'EF49   9E  1816  2365          MOVAB   DATA_OBUFFER R9 ,R6     ; GET ADDRESS OF EXP DATA BUFFER
          57   00000000'EF49   9E  181E  2366          MOVAB   DATA_IBUFFER R9 ,R7     ; GET ADDRESS OF REC DATA BUFFER
                               1826  2367          $DS_ERRHARD_S   #4,LUN,-        ; PRINT ERROR
                               1826  2368                          ARRAY_MSG15,-
                               1826  2369                          PRINT_DTERR
                00000000'EF    DF  1826                      PUSHAL  PRINT_DTERR
                00000000'EF    DF  182C                      PUSHAL  ARRAY_MSG15
                00000000'EF    DD  1832                      PUSHL   LUN
                        04     DD  1838                      PUSHL   #4
             00000000'9F   04   FB  183A                      CALLS   #$$M, a#DS$ERRHARD
                               1841  2370 70$:       $DS_CKLOOP      20$             ; SCOPE LOOP?
          00000000'9F   FE25 CF   FA  1841                      CALLG   20$, a#DS$CKLOOP
            A8 59   000001FF 8F   F3  184A  2371          AOBLEQ  #511,R9,60$             ; CHECK ENTIRE BUFFER
                        55   D5  1852  2372          TSTL    R5                      ; ANY ERRORS REPORTED?
                        0F   13  1854  2373          BEQL    80$                     ; BRANCH IF NONE
                               1856  2374          $DS_PRINTB_S    FMT_ERRSUM,R5   ; REPORT NUMBER OF ERRORS
                        55   DD  1856                      PUSHL   R5
                00000000'EF    9F  1858                      PUSHAB  FMT_ERRSUM
             00000000'9F   02   FB  185E                      CALLS   #$$N, a#DS$PRINTB
                               1865  2375 80$:
                               1865  2376 $DS_ENDSUB
                               1865       T20_S14_X:
    00000000'9F   000001B0'EF   FA  1865                      CALLG   $S$, a#DS$ENDSUB
```

```
                                1870  2378 $DS_BGNSUB
                                1870       T20_S15::
00000000'9F   000001BC'EF   FA  1870                 CALLG   $S$, a#DS$BGNSUB
                                187B  2379 ;++
                                187B  2380 ; DATA TRANSFER: ALL ONE'S PATTERN WITH WRITE CHECK REVERSE
                                187B  2381 ;                AND ODD BYTE COUNT
                                187B  2382 ;--
                                187B  2383         ERRPREP MBE_REG_MSG,12,-        ; PREPARE TO PRINT ERROR
                                187B  2384                 FMT_MBE_SR
00000000'EF   00000000'EF   DE  187B                 MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  1886                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  188D                 MOVAL   FMT_MBE_SR,REG_STRING
      52   00000000'EF      D0  1898  2385         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF MUT
 0000'C2   00000000'8F      C8  189F  2386         BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      58   00000000'EF      D0  18A8  2387         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
   59 52   00000000'8F      C1  18AF  2388         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REGISTER ADDRESS
                  50        D4  18B7  2389         CLRL    R0                      ; CLEAR INDEX
   00000000'EF40   00        D2  18B9  2390 10$:    MCOML   #0,DATA_OBUFFER R0      ; PUT PATTERN IN BUFFER
 F0 50   0000007F  8F        F3  18C1  2391         AOBLEQ  #127,R0,10$             ; FILL UP BOTH BUFFERS
 0000'C2   00000000'8F      C8  18C9  2392 20$:    BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
      5B   00000000'EF      DE  18D2  2393         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE BUFFER
       5A   5B    F7      8F  78  18D9  2394         ASHL    #-9,R11,R10             ; PUT PAGE FRAME NUMBER INTO R10
      5A   00000000'8F      C8  18DE  2395         BISL    #VALID_BIT,R10          ; SET VALID BIT
              69   5A       D0  18E5  2396         MOVL    R10,(R9)                ; WRITE PFN INTO MAP 0
      5B   00000000'8F      CA  18E8  2397         BICL    #MAP_PTR_MSK,R11        ; MASK OUT PFN TO OBTAIN BYTE ADDRESS
          0000'C2   5B      D0  16EF  2398         MOVL    R11,VAR(R2)             ; WRITE VIRTUAL ADDRESS REGISTER
                            18F4  2399                                             ; (SELECTING MAP 0)
 0000'C2   00000200  8F     CE  18F4  2400         MNEGL   #512,BCR(R2)            ; SET UP RH750 BYTE COUNT REGISTER
 0000'C8   00000000'8F      D0  18FD  2401         MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                            1906  2402                                             ;       BLOCK SIZE = 256 W
                            1906  2403                                             ;       GAP SIZE = 5 US
                            1906  2404                                             ;       DATA RATE = 1 MW/S
 0000'C8   00000000'8F      D0  1906  2405         MOVL    #WRITE,MBE_CR1(R8)      ; INITIATE WRITE COMMAND
                            190F  2406         $DS_WAITUS_S    #100                ; WAIT 1 MILLISECONDS
                      00    DD  190F                 PUSHL   #0
              00000064  8F    DD  1911                 PUSHL   #100
      00000000'9F       02    FB  1917                 CALLS   #2, a#DS$WAITUS
          50   0000'C8      D0  191E  2407         MOVL    MBE_SR(R8),R0           ; READ MBE STATUS REGISTER
               53   50      3C  1923  2408         MOVZWL  R0,R3                   ; EXTRACT REGISTER CONTENTS
            6B 53   0E      E1  1926  2409         BBC     #14,R3,30$              ; ERROR BIT SET?
      56   00000000'8F      D0  192A  2410         MOVL    #MBE_SR,R6              ; REPORT FAILING REGISTER
                      54    D4  1931  2411         CLRL    R4                      ; EXPECTING NO ERRORS
               55   53      D0  1933  2412         MOVL    R3,R5                   ; REPORT FAILING BIT
                            1936  2413         $DS_ERRHARD_S   #1,LUN,,-           ; PRINT ERROR
                            1936  2414                         PRINT_SBE
              00000000'EF   DF  1936                 PUSHAL  PRINT_SBE
                      00    DD  193C                 PUSHL   #0
              00000000'EF   DD  193E                 PUSHL   LUN
                      01    DD  1944                 PUSHL   #1
      00000000'9F       04    FB  1946                 CALLS   #$$M, a#DS$ERRHARD
                            194D  2415         ERRPREP MBE_REG_MSG,12,-            ; PRINT ERROR
                            194D  2416                 FMT_MBE_ER
00000000'EF   00000000'EF   DE  194D                 MOVAL   MBE_REG_MSG,REG_NAME
      00000000'EF   0C      9A  1958                 MOVZBL  #12,REG_NO
00000000'EF   00000000'EF   DE  195F                 MOVAL   FMT_MBE_ER,REG_STRING
          51   0000'C8      D0  196A  2417         MOVL    MBE_ER(R8),R1           ; READ MBE ERROR REGISTER
               53   51      3C  196F  2418         MOVZWL  R1,R3                   ; EXTRACT REGISTER CONTENTS
```

```
                    21   13  1972  2419          BEQL    30$                     ; IF NO ERROR BITS SET, SKIP
                55  53   D0  1974  2420          MOVL    R3,R5                   ; REPORT FAILING BIT(S)
        56  00000000'8F  D0  1977  2421          MOVL    #MBE_ER,R6              ; FAILING REGISTER
                             197E  2422          $DS_ERRHARD_S   #2,LUN,,-       ; PRINT ERROR
                             197E  2423                          PRINT_SBE
            00000000'EF  DF  197E                          PUSHAL  PRINT_SBE
                    00   DD  1984                          PUSHL   #0
            00000000'EF  DD  1986                          PUSHL   LUN
                    02   DD  198C                          PUSHL   #2
        00000000'9F  04  FB  198E                          CALLS   #$$M, a#DS$ERRHARD
                             1995  2424 30$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
    00000000'9F  FF30 CF  FA  1995                          CALLG   20$, a#DS$CKLOOP
    0000'C2  00000000'8F  C8  199E  2425 35$:     BISL    #PGM_INIT,CR(R2)        ; INITIALIZE MUT
        5B  00000000'EF  DE  19A7  2426          MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF READ BUFFER
         5A   5B   F7 8F  78  19AE  2427          ASHL    #-9,R11,R10             ; PUT PFN INTO R10
        5A  00000000'8F  C8  19B3  2428          BISL    #VALID_BIT,R10          ; SET V BIT
                69  5A   D0  19BA  2429          MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
        5B  000001FF 8F  D0  19BD  2430          MOVL    # X1FF,R11              ; POINT TO BYTE 511
            0000'C2  5B  D0  19C4  2431          MOVL    R11,VAR(R2)             ; WRITE VAR,SELECTING MAP 0
    0000'C2  000001FF 8F  CE  19C9  2432          MNEGL   #511,BCR(R2)            ; SET UP RH750 BYTE COUNT
    0000'C8  0000FFFF 8F  D0  19D2  2433          MOVL    # XFFFF,MBE_DBR(R8)     ; PUT PATTERN IN DATA BUFFER
    0000'C8  00000000'8F  D0  19DB  2434          MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                             19E4  2435                                          ; BLOCK SIZE = 256 W
                             19E4  2436                                          ; GAP SIZE = 5 US
                             19E4  2437                                          ; DATA RATE = 1 MW/S
            00000000'EF  94  19E4  2438          CLRB    DATA_OBUFFER            ; GARBAGE BYTE ZERO
    0000'C8  00000000'8F  D0  19EA  2439          MOVL    #WRITE_CHECKREV,MBE_CR1(R8)     ; ISSUE WRITE CHECK
                             19F3  2440                                                  ; REVERSE COMMAND
                             19F3  2441          $DS_WAITUS_S    #100            ; WAIT 1 MILLISECONDS
                    00   DD  19F3                          PUSHL   #0
            00000064 8F  DD  19F5                          PUSHL   #100
        00000000'9F  02  FB  19FB                          CALLS   #2, a#DS$WAITUS
                             1A02  2442          ERRPREP RHSR_MSG,2,FMT_STATUS_REG       ; PREPARE TO HANDLE ERROR
00000000'EF  00000000'EF  DE  1A02                          MOVAL   RHSR_MSG,REG_NAME
        00000000'EF  02  9A  1A0D                          MOVZBL  #2,REG_NO
00000000'EF  00000000'EF  DE  1A14                          MOVAL   FMT_STATUS_REG,REG_STRING
            53   0000'C2  D0  1A1F  2443          MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
        53  00000000'8F  D3  1A24  2444          BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                    20   13  1A2B  2445          BEQL    50$                     ; BRANCH IF NONE
                    54   D4  1A2D  2446          CLRL    R4                      ; EXPECT NO ERRORS
                55  53   D0  1A2F  2447          MOVL    R3,R5                   ; REPORT FAILING BITS
                             1A32  2448          $DS_ERRHARD_S   #3,LUN,-        ; PRINT ERROR
                             1A32  2449                          ARRAY_MSG24,-
                             1A32  2450                          PRINT_SBE
            00000000'EF  DF  1A32                          PUSHAL  PRINT_SBE
            00000000'EF  DF  1A38                          PUSHAL  ARRAY_MSG24
            00000000'EF  DD  1A3E                          PUSHL   LUN
                    03   DD  1A44                          PUSHL   #3
        00000000'9F  04  FB  1A46                          CALLS   #$$M, a#DS$ERRHARD
                             1A4D  2451 50$:     $DS_CKLOOP      35$             ; SCOPE LOOP
    00000000'9F  FF4D CF  FA  1A4D                          CALLG   35$, a#DS$CKLOOP
                             1A56  2452
                             1A56  2453 $DS_ENDSUB
                             1A56       T20_S15_X:
00000000'9F  000001BC'EF  FA  1A56                          CALLG   $S$, a#DS$ENDSUB
```

```
                                1A61  2455 $DS_BGNSUB
                                1A61       T20_S16::
    00000000'9F   000001C8'EF   FA  1A61                    CALLG   $S$, a#DS$BGNSUB
                                1A6C  2456 ;++
                                1A6C  2457 ; DATA TRANSFER:  ALL ONES'S PATTERN WITH WRITE CHECK FORWARD
                                1A6C  2458 ;                 AND ODD BYTE COUNT
                                1A6C  2459 ;--
        0000'C2   00000000'8F   C8  1A6C  2460 10$:    BISL    #PGM_INIT,CR(R2)        ; INIT MUT
                          50    D4  1A75  2461         CLRL    R0                      ; CLEAR INDEX
           00000000'EF40    00   D2  1A77  2462 15$:    MCOML   #0,DATA_OBUFFER R0      ; PUT PATTERN IN BUFFER
         F0 50    0000007F 8F   F3  1A7F  2463         AOBLEQ  #127,R0,15$             ; FILL UP BOTH BUFFERS
            5B    00000000'EF   DE  1A87  2464         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF READ BUFFER
                      01FF CB   94  1A8E  2465         CLRB    511(R11)                ; CLEAR HIGH BYTE FOR LAST XFER
             5A    5B    F7 8F   78  1A92  2466         ASHL    #-9,R11,R10             ; PUT PFN INTO R10
            5A    00000000'8F   C8  1A97  2467         BISL    #VALID_BIT,R10          ; SET V-BIT
                      69    5A   D0  1A9E  2468         MOVL    R10,(R9)                ; PUT PFN INTO MAP 0
                      0000'C2   D4  1AA1  2469         CLRL    VAR(R2)                 ; START WITH BYTE 0 MAP 0
        0000'C2   000001FF 8F   CE  1AA5  2470         MNEGL   #511,BCR(R2)            ; SET UP RH750 BYTE COUNT
        0000'C8   0000FFFF 8F   D0  1AAE  2471         MOVL    # XFFFF,MBE_DBR(R8)    ; PUT PATTERN IN DATA BUFFER
        0000'C8   00000000'8F   D0  1AB7  2472         MOVL    #MBE_PARAM,MBE_CR2(R8)  ; SET MBE PARAMETERS:
                                1AC0  2473                                             ; BLOCK SIZE = 256 W
                                1AC0  2474                                             ; GAP SIZE = 5 US
                                1AC0  2475                                             ; DATA RATE = 1 MW/S
        0000'C8   00000000'8F   D0  1AC0  2476         MOVL    #WRITE_CHECK,MBE_CR1(R8); ISSUE WRITE CHECK FOWARD CMD
                                1AC9  2477         $DS_WAITUS_S    #100                ; WAIT ONE MILLISECOND
                          00    DD  1AC9                    PUSHL   #0
                      00000064 8F   DD  1ACB                    PUSHL   #100
              00000000'9F   02    FB  1AD1                    CALLS   #2, a#DS$WAITUS
                                1AD8  2478         ERRPREP RHSR_MSG,2,FMT_STATUS_REG; PREPARE TO HANDLE ERROR
    00000000'EF   00000000'EF   DE  1AD8                    MOVAL   RHSR_MSG,REG_NAME
              00000000'EF   02  9A  1AE3                    MOVZBL  #2,REG_NO
    00000000'EF   00000000'EF   DE  1AEA                    MOVAL   FMT_STATUS_REG,REG_STRING
                 53   0000'C2   D0  1AF5  2479         MOVL    SR(R2),R3               ; READ RH750 STATUS REGISTER
              53   00000000'8F   D3  1AFA  2480         BITL    #RHSR_CHECK,R3          ; ANY STATUS ERRORS?
                          20    13  1B01  2481         BEQL    20$                     ; BRANCH IF NONE
                          54    D4  1B03  2482         CLRL    R4                      ; EXPECT NO ERRORS
                      55    53  D0  1B05  2483         MOVL    R3,R5                   ; REPORT FAILING BITS
                                1B08  2484         $DS_ERRHARD_S   #1,LUN,-            ; PRINT ERROR
                                1B08  2485                         ARRAY_MSG24,-
                                1B08  2486                         PRINT_SBE
                    00000000'EF   DF  1B08                    PUSHAL  PRINT_SBE
                    00000000'EF   DF  1B0E                    PUSHAL  ARRAY_MSG24
                    00000000'EF   DD  1B14                    PUSHL   LUN
                          01    DD  1B1A                    PUSHL   #1
              00000000'9F   04    FB  1B1C                    CALLS   #$$M, a#DS$ERRHARD
                                1B23  2487 20$:    $DS_CKLOOP      10$                 ; SCOPE LOOP?
        00000000'9F   FF45 CF   FA  1B23                    CALLG   10$, a#DS$CKLOOP
                                1B2C  2488 $DS_ENDSUB
                                1B2C       T20_S16_X:
    00000000'9F   000001C8'EF   FA  1B2C                    CALLG   $S$, a#DS$ENDSUB
```

```
                                1B37  2490 $DS_BGNSUB
                                1B37       T20_S17::
00000000'9F   000001D4'EF   FA  1B37                   CALLG   $S$, a#DS$BGNSUB
                                1B42  2491 ;++
                                1B42  2492 ; THIS SUBTEST CHECKS THAT MASSBUS DATA IS LOADED INTO THE CORRECT
                                1B42  2493 ; POSITION IN THE SCDB FOR THE FOUR POSSIBLE BYTE BOUNDARIES IN THE
                                1B42  2494 ; FORWARD DIRECTION.
                                1B42  2495 ;--
                                1B42  2496
      52   00000000'EF      D0  1B42  2497         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF RH750
0000'C2   00000000'8F      C8  1B49  2498         BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                                1B52  2499
                                1B52  2500 ;
                                1B52  2501 ; The first section of code loads an expected data pattern of
                                1B52  2502 ; 32 bytes into memory at DATA_OBUFFER.
                                1B52  2503 ;
                                1B52  2504
                     50     D4  1B52  2505         CLRL    R0                      ; CLEAR BYTE COUNT INDEX
      55   00000000'EF      DE  1B54  2506         MOVAL   DATA_OBUFFER,R5         ; GET STARTING ADDRESS OF BUFFER
      58   00000000'EF      D0  1B5B  2507         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
      51   33553355 8F      D0  1B62  2508         MOVL    # X33553355,R1          ; INITIALIZE PATTERN
                85     51   B0  1B69  2509 10$:    MOVW    R1,(R5)+                ; WRITE PATTERN INTO EXP DATA BUFFER
             51   51     01   9C  1B6C  2510         ROTL    #1,R1,R1                ; GENERATE NEXT PATTERN
               F5 50     11   F3  1B70  2511         AOBLEQ  #17,R0,10$              ; WRITE 36 BYTES OF DATA
                                1B74  2512
                                1B74  2513 ;
                                1B74  2514 ; The next section of code clears out the rcvd data buffer and does
                                1B74  2515 ; a Read-From-Drive command in the forward direction with the
                                1B74  2516 ; Enable Pattern Shift Bit asserted in the MBE. The RFD command
                                1B74  2517 ; is repeated for the four byte boundaries, and the exp. and rcvd
                                1B74  2518 ; data is compared each time.
                                1B74  2519 ;
                                1B74  2520
                     55     D4  1B74  2521         CLRL    R5                      ; CLEAR THE BYTE INDEX COUNTER
0000'C2   00000000'8F      C8  1B76  2522 20$:    BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
      5B   00000000'EF      DE  1B7F  2523         MOVAL   DATA_IBUFFER,R11        ; GET ADDRESS OF RCV DATA BUFFER
                     50     D4  1B86  2524         CLRL    R0                      ; CLEAR BUFFER INDEX
                     8B     D4  1B88  2525 30$:    CLRL    (R11)+                  ; CLEAR RCV DATA BUFFER
               FA 50    08   F3  1B8A  2526         AOBLEQ  #8,R0,30$               ; CLEAR OUT BUFFER
      5B   00000000'EF      DE  1B8E  2527         MOVAL   DATA_IBUFFER,R11        ; RESTORE ADDRESS OF RCV DATA BUFFER
       5A    5B    F7 8F      78  1B95  2528         ASHL    #-9,R11,R10             ; EXTRACT PFN
      5A   00000000'8F      C8  1B9A  2529         BISL    #VALID_BIT,R10          ; SET V-BIT
      58   00000000'EF      D0  1BA1  2530         MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
59    52   00000000'8F      C1  1BA8  2531         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REG ADDRESS
                69     5A   D0  1BB0  2532         MOVL    R10,(R9)                ; PUT PFN INTO MAP REG 0
      5B   00000000'8F      CA  1BB3  2533         BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
    0000'C2    5B     55   C1  1BBA  2534         ADDL3   R5,R11,VAR(R2)          ; WRITE VAR, SEL MAP REG 0
         0000'C2    20     CE  1BC0  2535         MNEGL   #32,BCR(R2)             ; SET UP RH750 BYTE COUNT
0000'C8   00003355 8F      D0  1BC5  2536         MOVL    # X3355,MBE_DBR(R8)     ; LOAD PATTERN INTO DATA BUFFER
         0000'C8    27     D0  1BCE  2537         MOVL    # X27,MBE_CR2(R8)       ; SET MBE PARAMETERS:
                                1BD3  2538                                         ; BLOCK SIZE = 64 W
                                1BD3  2539                                         ; GAP SIZE = 5 US
                                1BD3  2540                                         ; DATA RATE = 1 MW/S
0000'C8   00000000'8F      D0  1BD3  2541         MOVL    #ENABLE_PS,MBE_MR(R8)   ; ENABLE PATTERN SHIFT
0000'C8   00000000'8F      D0  1BDC  2542         MOVL    #READ,MBE_CR1(R8)       ; ISSUE READ FWD COMMAND
                                1BE5  2543         $DS_WAITUS_S    #100            ; WAIT 1 MILLISECOND
                     00     DD  1BE5                   PUSHL   #0
```

```
                00000064 8F   DD  1BE7                          PUSHL   #100
            00000000'9F   02   FB  1BED                          CALLS   #2, a#DS$WAITUS
                          59   D4  1BF4  2544          CLRL     R9                          ; CLEAR INDEX
        54   00000000'EF49   90  1BF6  2545 40$:     MOVB     DATA_OBUFFER R9 ,R4         ; GET EXPECTED DATA
           53   0000'C549   90  1BFE  2546          MOVB     DATA_IBUFFER(R5) R9 ,R3     ; GET RECEIVED DATA
                     54   53   91  1C04  2547          CMPB     R3,R4                       ; EXP=RCV ?
                          2B   13  1C07  2548          BEQL     50$                         ; BRANCH IF DATA EQUAL
        56   00000000'EF49   9E  1C09  2549          MOVAB    DATA_OBUFFER R9 ,R6         ; GET ADDRESS OF EXP DATA BUFFER
        57   00000000'EF49   9E  1C11  2550          MOVAB    DATA_IBUFFER R9 ,R7         ; GET ADRESS OF RCVD DATA BUFFER
                                  1C19  2551          $DS_ERRHARD_S    #1,LUN,-           ; REPORT ERROR
                                  1C19  2552                           ARRAY_MSG15,-
                                  1C19  2553                           PRINT_DTERR
                00000000'EF   DF  1C19                          PUSHAL  PRINT_DTERR
                00000000'EF   DF  1C1F                          PUSHAL  ARRAY_MSG15
                00000000'EF   DD  1C25                          PUSHL   LUN
                          01   DD  1C2B                          PUSHL   #1
            00000000'9F   04   FB  1C2D                          CALLS   #$$M, a#DS$ERRHARD
                                  1C34  2554 50$:     $DS_CKLOOP       20$                 ; SCOPE LOOP?
      00000000'9F   FF3E CF   FA  1C34                          CALLG   20$, a#DS$CKLOOP
                 B5 59   1F   F3  1C3D  2555          AOBLEQ   #31,R9,40$                  ; CHECK 32 BYTES OF DATA
          FF2F 55   01   04   3D  1C41  2556          ACBW     #4,#1,R5,20$                ; REPEAT FOR FOUR BYTE BOUNDARIES
                                  1C47  2557 $DS_ENDSUB
                                  1C47       T20_S17_X:
   00000000'9F   000001D4'EF   FA  1C47                          CALLG   $S$, a#DS$ENDSUB
```

```
                                1C52  2559 $DS_BGNSUB
                                1C52       T20_S18::
00000000'9F   000001E0'EF   FA  1C52                 CALLG   $S$, a#DS$BGNSUB
                                1C5D  2560 ;++
                                1C5D  2561 ; THIS SUBTEST CHECKS THAT MASSBUS DATA IS LOADED INTO THE CORRECT
                                1C5D  2562 ; POSITION IN THE SCDB FOR THE FOUR POSSIBLE BYTE BOUNDARIES IN THE
                                1C5D  2563 ; REVERSE DIRECTION.
                                1C5D  2564 ;--
                                1C5D  2565
      52   00000000'EF   D0     1C5D  2566         MOVL    RH_CUR_ADR,R2          ; GET ADDRESS OF RH750
0000'C2   00000000'8F   C8     1C64  2567         BISL    #PGM_INIT,CR(R2)       ; INIT THE MUT
                                1C6D  2568
                                1C6D  2569 ;
                                1C6D  2570 ; The first section of code loads an expected data pattern in
                                1C6D  2571 ; reverse direction into memory at DATA_OBUFFER +36.
                                1C6D  2572 ;
                                1C6D  2573
            50   D4             1C6D  2574         CLRL    R0                     ; CLEAR BYTE COUNT INDEX
      55   00000000'EF   DE     1C6F  2575         MOVAL   DATA_OBUFFER,R5        ; GET STARTING ADDRESS OF BUFFER
         55   24             C0  1C76  2576         ADDL2   #36,R5                 ; POINT TO END OF EXP DATA BUFFER
      51   33553355 8F   D0     1C79  2577         MOVL    # X33553355,R1        ; INITIALIZE PATTERN
         75   51             B0  1C80  2578 10$:    MOVW    R1,-(R5)               ; WRITE PATTERN INTO EXP DATA BUFFER
      51   51   01         9C  1C83  2579         ROTL    #1,R1,R1               ; GENERATE NEXT PATTERN
        F5 50   11         F3  1C87  2580         AOBLEQ  #17,R0,10$             ; WRITE 36 BYTES OF DATA
                                1C8B  2581
                                1C8B  2582 ;
                                1C8B  2583 ; The next section of code clears out the rcvd data buffer and does
                                1C8B  2584 ; a Read-From-Drive command in the reverse direction with the
                                1C8B  2585 ; Enable Pattern Shift Bit asserted in the MBE. The RFD command
                                1C8B  2586 ; is repeated for the four byte boundaries, and the exp. and rcvd
                                1C8B  2587 ; data is compared each time.
                                1C8B  2588 ;
                                1C8B  2589
   00000000'EF   D4             1C8B  2590         CLRL    TEMP1                  ; ZERO COUNTER
      55   03   D0             1C91  2591         MOVL    #3,R5                  ; INITIALIZE THE BYTE INDEX COUNTER
      56   55   D0             1C94  2592         MOVL    R5,R6                  ; GET COPY OF BYTE INDEX
0000'C2   00000000'8F   C8     1C97  2593 20$:    BISL    #PGM_INIT,CR(R2)       ; INIT THE MUT
   5B   00000000'EF   DE     1CA0  2594         MOVAL   DATA_IBUFFER,R11       ; GET ADDRESS OF RCV DATA BUFFER
            50   D4             1CA7  2595         CLRL    R0                     ; CLEAR BUFFER INDEX
            8B   D4             1CA9  2596 30$:    CLRL    (R11)+                 ; CLEAR RCV DATA BUFFER
     FA 50   08   F3             1CAB  2597         AOBLEQ  #8,R0,30$              ; CLEAR OUT BUFFER
   5B   00000000'EF   DE     1CAF  2598         MOVAL   DATA_IBUFFER,R11       ; RESTORE ADDRESS OF RCV DATA BUFFER
    5A   5B   F7 8F   78     1CB6  2599         ASHL    #-9,R11,R10            ; EXTRACT PFN
   5A   00000000'8F   C8     1CBB  2600         BISL    #VALID_BIT,R10         ; SET V-BIT
   58   00000000'EF   DE     1CC2  2601         MOVL    MBE_CUR_ADR,R8         ; BET MBE ADDRESS
59 52   00000000'8F   C1     1CC9  2602         ADDL3   #MAP_OFFSET,R2,R9      ; CALCULATE MAP REG ADDRESS
         69   5A   D0         1CD1  2603         MOVL    R10,(R9)               ; PUT PFN INTO MAP REG 0
   5B   00000000'8F   CA     1CD4  2604         BICL    #MAP_PTR_MSK,R11       ; ISOLATE BYTE OFFSET
         5B   20   C0         1CDB  2605         ADDL2   #32,R11                ; POINT TO BYTE 32
    0000'C2   5B   55   C1     1CDE  2606         ADDL3   R5,R11,VAR(R2)         ; WRITE VAR, SEL MAP REG 0
         0000'C2   20   CE     1CE4  2607         MNEGL   #32,BCR(R2)            ; SET UP RH750 BYTE COUNT
0000'C8   00003355 8F   D0     1CE9  2608         MOVL    # X3355,MBE_DBR(R8)   ; LOAD PATTERN INTO DATA BUFFER
        0000'C8   27   D0     1CF2  2609         MOVL    # X27,MBE_CR2(R8)     ; SET MBE PARAMETERS:
                                1CF7  2610                                        ; BLOCK SIZE = 64 W
                                1CF7  2611                                        ; GAP SIZE = 5 US
                                1CF7  2612                                        ; DATA RATE = 1 MW/S
0000'C8   00000000'8F   D0     1CF7  2613         MOVL    #ENABLE_PS,MBE_MR(R8)  ; ENABLE PATTERN SHIFT
```

```
                0000'C8   00000000'8F   D0  1D00  2614          MOVL    #READ_REV,MBE_CR1(R8)   ; ISSUE READ REVERSE COMMAND
                                       1D09  2615          $DS_WAITUS_S    #100            ; WAIT 1 MILLISECOND
                                 00    DD  1D09                    PUSHL   #0
                          00000064 8F    DD  1D0B                    PUSHL   #100
                      00000000'9F   02    FB  1D11                    CALLS   #2, a#DS$WAITUS
                             59   20    D0  1D18  2616          MOVL    #32,R9                  ; INITIALIZE RCV DATA BUF POINTER
                       54   0000'C649    90  1D1B  2617 40$:    MOVB    DATA_OBUFFER(R6) R9 ,R4 ; GET EXPECTED DATA
                       53   0000'C549    90  1D21  2618          MOVB    DATA_IBUFFER(R5) R9 ,R3 ; GET RECEIVED DATA
                             54   53    91  1D27  2619          CMPB    R3,R4                   ; EXP=RCV ?
                                  2B    13  1D2A  2620          BEQL    50$                     ; BRANCH IF DATA EQUAL
                  56   00000000'EF49    9E  1D2C  2621          MOVAB   DATA_OBUFFER R9 ,R6     ; GET ADDRESS OF EXP DATA BUFFER
                  57   00000000'EF49    9E  1D34  2622          MOVAB   DATA_IBUFFER R9 ,R7     ; GET ADDRESS OF RCVD DATA BUFFER
                                       1D3C  2623          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                       1D3C  2624                      ARRAY_MSG15,-
                                       1D3C  2625                      PRINT_DTERR
                          00000000'EF    DF  1D3C                    PUSHAL  PRINT_DTERR
                          00000000'EF    DF  1D42                    PUSHAL  ARRAY_MSG15
                          00000000'EF    DD  1D48                    PUSHL   LUN
                                 01    DD  1D4E                    PUSHL   #1
                      00000000'9F   04    FB  1D50                    CALLS   #$$M, a#DS$ERRHARD
                                       1D57  2626 50$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
                00000000'9F   FF3C CF    FA  1D57                    CALLG   20$, a#DS$CKLOOP
                                B8 59    F5  1D60  2627          SOBGTR  R9,40$                  ; CHECK 32 BYTES OF DATA
                                   55    D7  1D63  2628          DECL    R5                      ; POINT TO NEXT BYTE ADDRESS IN RCV BUF
            FF28 00000000'EF   01   04    3D  1D65  2629          ACBW    #4,#1,TEMP1,20$         ; REPEAT FOR FOUR BYTE BOUNDARIES
                                       1D6F  2630 $DS_ENDSUB
                                       1D6F       T20_S18_X:
             00000000'9F   000001E0'EF    FA  1D6F                    CALLG   $S$, a#DS$ENDSUB
```

```
                                1D7A  2632 $DS_BGNSUB
                                1D7A       T20_S19::
00000000'9F   000001EC'EF   FA  1D7A                   CALLG   $S$, a#DS$BGNSUB
                                1D85  2633 ;++
                                1D85  2634 ; THIS SUBTEST CHECKS THAT MASSBUS DATA IS LOADED INTO THE CORRECT
                                1D85  2635 ; POSITION IN THE SCDB FOR THE FOUR POSSIBLE BYTE BOUNDARIES IN THE
                                1D85  2636 ; WRITE CHECK FORWARD DIRECTION.
                                1D85  2637 ;--
                                1D85  2638
      52   00000000'EF   D0   1D85  2639          MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF RH750
0000'C2   00000000'8F   C8   1D8C  2640          BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                                1D95  2641
                                1D95  2642 ;
                                1D95  2643 ; The first section of code loads an expected data pattern of
                                1D95  2644 ; 32 bytes into memory at DATA_OBUFFER.
                                1D95  2645 ;
                                1D95  2646
                     50   D4   1D95  2647          CLRL    R0                      ; CLEAR BYTE COUNT INDEX
      55   00000000'EF   DE   1D97  2648          MOVAL   DATA_OBUFFER,R5         ; GET STARTING ADDRESS OF BUFFER
      58   00000000'EF   D0   1D9E  2649          MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
      51   33553355 8F   D0   1DA5  2650          MOVL    # X33553355,R1          ; INITIALIZE PATTERN
             6540   51   B0   1DAC  2651 10$:     MOVW    R1,(R5) R0              ; WRITE PATTERN INTO EXP DATA BUFFER
          51   51   01   9C   1DB0  2652          ROTL    #1,R1,R1                ; GENERATE NEXT PATTERN
            F4 50   11   F3   1DB4  2653          AOBLEQ  #17,R0,10$              ; WRITE 36 BYTES OF DATA
                                1DB8  2654
                                1DB8  2655 ;
                                1DB8  2656 ; The next section of code clears out the rcvd data buffer and does
                                1DB8  2657 ; a Write-Check-Foreward command in the forward direction with the
                                1DB8  2658 ; Enable Pattern Shift Bit asserted in the MBE. The WCF command
                                1DB8  2659 ; is repeated for the four byte boundaries, and the exp. and rcvd
                                1DB8  2660 ; data is compared each time.
                                1DB8  2661 ;
                                1DB8  2662
                     55   D4   1DB8  2663          CLRL    R5                      ; CLEAR THE BYTE INDEX COUNTER
0000'C2   00000000'8F   C8   1DBA  2664 20$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
      5B   00000000'EF   DE   1DC3  2665          MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE CHECK BUFFER
       5A   5B   F7 8F   78   1DCA  2666          ASHL    #-9,R11,R10             ; EXTRACT PFN
      5A   00000000'8F   C8   1DCF  2667          BISL    #VALID_BIT,R10          ; SET V-BIT
      58   00000000'EF   D0   1DD6  2668          MOVL    MBE_CUR_ADR,R8          ; GET MBE ADDRESS
59    52   00000000'8F   C1   1DDD  2669          ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REG ADDRESS
             69   5A   D0   1DE5  2670          MOVL    R10,(R9)                ; PUT PFN INTO MAP REG 0
      5B   00000000'8F   CA   1DE8  2671          BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
    0000'C2   5B   55   C1   1DEF  2672          ADDL3   R5,R11,VAR(R2)          ; WRITE VAR, SEL MAP REG 0
      0000'C2   20   CE   1DF5  2673          MNEGL   #32,BCR(R2)             ; SET UP RH750 BYTE COUNT
0000'C8   00003355 8F   D0   1DFA  2674          MOVL    # X3355,MBE_DBR(R8)     ; LOAD PATTERN INTO DATA BUFFER
      0000'C8   27   D0   1E03  2675          MOVL    # X27,MBE_CR2(R8)       ; SET MBE PARAMETERS:
                                1E08  2676                                          ; BLOCK SIZE = 64 W
                                1E08  2677                                          ; GAP SIZE = 5 US
                                1E08  2678                                          ; DATA RATE = 1 MW/S
0000'C8   00000000'8F   D0   1E08  2679          MOVL    #ENABLE_PS,MBE_MR(R8)   ; ENABLE PATTERN SHIFT
0000'C8   00000000'8F   D0   1E11  2680          MOVL    #WRITE_CHECK,MBE_CR1(R8); ISSUE WRITE CHECK FWD COMMAND
                                1E1A  2681          $DS_WAITUS_S    #100            ; WAIT 1 MILLISECOND
                     00   DD   1E1A                         PUSHL   #0
            00000064 8F   DD   1E1C                         PUSHL   #100
      00000000'9F   02   FB   1E22                         CALLS   #2, a#DS$WAITUS
                                1E29  2682          ERRPREP RHSR_MSG,2,FMT_STATUS_REG
00000000'EF   00000000'EF   DE   1E29                         MOVAL   RHSR_MSG,REG_NAME
```

```
                00000000'EF    02    9A  1E34                          MOVZBL  #2,REG_NO
     00000000'EF    00000000'EF    DE  1E3B                          MOVAL   FMT_STATUS_REG,REG_STRING
    55    0000'C2    FFFFFFFF'8F    CB  1E46  2683          BICL3   # C<RHSR_CHECK>,SR(R2),R5; GET STATUS REGISTER
                                 1D    13  1E50  2684          BEQL    50$                     ; BRANCH IF NO
                                 54    D4  1E52  2685          CLRL    R4                      ; EXPECTED DATA
                                       1E54  2686          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                                       1E54  2687                          ARRAY_MSG24,-
                                       1E54  2688                          PRINT_SBE
                     00000000'EF    DF  1E54                          PUSHAL  PRINT_SBE
                     00000000'EF    DF  1E5A                          PUSHAL  ARRAY_MSG24
                     00000000'EF    DD  1E60                          PUSHL   LUN
                                 01    DD  1E66                          PUSHL   #1
                00000000'9F    04    FB  1E68                          CALLS   #$$M, a#DS$ERRHARD
                                       1E6F  2689
                                       1E6F  2690 50$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
          00000000'9F    FF47 CF    FA  1E6F                          CALLG   20$, a#DS$CKLOOP
                                       1E78  2691 $DS_ENDSUB
                                       1E78       T20_S19_X:
     00000000'9F    000001EC'EF    FA  1E78                          CALLG   $S$, a#DS$ENDSUB
```

```
                                1E83  2693 $DS_BGNSUB
                                1E83       T20_S20::
00000000'9F   000001F8'EF   FA   1E83                 CALLG   $S$, a#DS$BGNSUB
                                1E8E  2694 ;++
                                1E8E  2695 ; THIS SUBTEST CHECKS THAT MASSBUS DATA IS LOADED INTO THE CORRECT
                                1E8E  2696 ; POSITION IN THE SCDB FOR THE FOUR POSSIBLE BYTE BOUNDARIES IN THE
                                1E8E  2697 ; WRITE CHECK REVERSE DIRECTION.
                                1E8E  2698 ;--
                                1E8E  2699
      52   00000000'EF   D0   1E8E  2700         MOVL    RH_CUR_ADR,R2           ; GET ADDRESS OF RH750
 0000'C2   00000000'8F   C8   1E95  2701         BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
                                1E9E  2702
                                1E9E  2703 ;
                                1E9E  2704 ; The first section of code loads an expected data pattern in
                                1E9E  2705 ; reverse direction into memory at DATA_OBUFFER +36.
                                1E9E  2706 ;
                                1E9E  2707
            50   D4   1E9E  2708         CLRL    R0                      ; CLEAR BYTE COUNT INDEX
      55   00000000'EF   DE   1EA0  2709         MOVAL   DATA_OBUFFER,R5         ; GET STARTING ADDRESS OF BUFFER
           55   24   C0   1EA7  2710         ADDL2   #36,R5                  ; POINT TO END OF EXP DATA BUFFER
      51   33553355 8F   D0   1EAA  2711         MOVL    # X33553355,R1         ; INITIALIZE PATTERN
           75   51   B0   1EB1  2712 10$:     MOVW    R1,-(R5)                ; WRITE PATTERN INTO EXP DATA BUFFER
        51   51   01   9C   1EB4  2713         ROTL    #1,R1,R1                ; GENERATE NEXT PATTERN
          F5 50   11   F3   1EB8  2714         AOBLEQ  #17,R0,10$              ; WRITE 36 BYTES OF DATA
                                1EBC  2715
                                1EBC  2716 ;
                                1EBC  2717 ; The next section of code clears out the rcvd data buffer and does
                                1EBC  2718 ; a Write-Check-Reverse command in the reverse direction with the
                                1EBC  2719 ; Enable Pattern Shift Bit asserted in the MBE. The WCR command
                                1EBC  2720 ; is repeated for the four byte boundaries, and the exp. and rcvd
                                1EBC  2721 ; data is compared each time.
                                1EBC  2722 ;
                                1EBC  2723
   00000000'EF   D4   1EBC  2724         CLRL    TEMP1                   ; ZERO COUNTER
       55   03   D0   1EC2  2725         MOVL    #3,R5                   ; INITIALIZE THE BYTE INDEX COUNTER
       56   55   D0   1EC5  2726         MOVL    R5,R6                   ; GET COPY OF BYTE INDEX
0000'C2   00000000'8F   C8   1EC8  2727 20$:     BISL    #PGM_INIT,CR(R2)        ; INIT THE MUT
   5B   00000000'EF   DE   1ED1  2728         MOVAL   DATA_OBUFFER,R11        ; GET ADDRESS OF WRITE CHECK BUFFER
    5A   5B   F7 8F   78   1ED8  2729         ASHL    #-9,R11,R10             ; EXTRACT PFN
   5A   00000000'8F   C8   1EDD  2730         BISL    #VALID_BIT,R10          ; SET V-BIT
   58   00000000'EF   D0   1EE4  2731         MOVL    MBE_CUR_ADR,R8          ; BET MBE ADDRESS
59 52   00000000'8F   C1   1EEB  2732         ADDL3   #MAP_OFFSET,R2,R9       ; CALCULATE MAP REG ADDRESS
       69   5A   D0   1EF3  2733         MOVL    R10,(R9)                ; PUT PFN INTO MAP REG 0
   5B   00000000'8F   CA   1EF6  2734         BICL    #MAP_PTR_MSK,R11        ; ISOLATE BYTE OFFSET
       5B   20   C0   1EFD  2735         ADDL2   #32,R11                 ; POINT TO BYTE 32
 0000'C2   5B   55   C1   1F00  2736         ADDL3   R5,R11,VAR(R2)          ; WRITE VAR, SEL MAP REG 0
      0000'C2   20   CE   1F06  2737         MNEGL   #32,BCR(R2)             ; SET UP RH750 BYTE COUNT
0000'C8    00003355 8F   D0   1F0B  2738         MOVL    # X3355,MBE_DBR(R8)     ; LOAD PATTERN INTO DATA BUFFER
      0000'C8   27   D0   1F14  2739         MOVL    # X27,MBE_CR2(R8)       ; SET MBE PARAMETERS:
                                1F19  2740                                 ; BLOCK SIZE = 64 W
                                1F19  2741                                 ; GAP SIZE = 5 US
                                1F19  2742                                 ; DATA RATE = 1 MW/S
0000'C8   00000000'8F   D0   1F19  2743         MOVL    #ENABLE_PS,MBE_MR(R8)   ; ENABLE PATTERN SHIFT
0000'C8   00000000'8F   D0   1F22  2744         MOVL    #WRITE_CHECKREV,MBE_CR1(R8)
                                1F2B  2745                                 ; ISSUE WRITE CHECK REVERSE COMMAND
                                1F2B  2746         $DS_WAITUS_S    #100    ; WAIT 1 MILLISECOND
                   00   DD   1F2B                         PUSHL   #0
```

```
                00000064 8F    DD  1F2D                        PUSHL   #100
            00000000'9F  02    FB  1F33                        CALLS   #2, a#DS$WAITUS
                               1F3A  2747          ERRPREP     RHSR_MSG,2,FMT_STATUS_REG
    00000000'EF  00000000'EF   DE  1F3A                        MOVAL   RHSR_MSG,REG_NAME
            00000000'EF  02    9A  1F45                        MOVZBL  #2,REG_NO
    00000000'EF  00000000'EF   DE  1F4C                        MOVAL   FMT_STATUS_REG,REG_STRING
 55   0000'C2   FFFFFFFF'8F    CB  1F57  2748          BICL3   # C<RHSR_CHECK>,SR(R2),R5
                               1F61  2749                                  ; READ THE STATUS REGISTER
                         1D    13  1F61  2750          BEQL    50$         ; BRANCH IF NO ERROR
                         54    D4  1F63  2751          CLRL    R4          ; EXPECTED DATA
                               1F65  2752          $DS_ERRHARD_S   #1,LUN,-        ; REPORT ERROR
                               1F65  2753                          ARRAY_MSG24,-
                               1F65  2754                          PRINT_SBE
                00000000'EF    DF  1F65                        PUSHAL  PRINT_SBE
                00000000'EF    DF  1F6B                        PUSHAL  ARRAY_MSG24
                00000000'EF    DD  1F71                        PUSHL   LUN
                         01    DD  1F77                        PUSHL   #1
            00000000'9F  04    FB  1F79                        CALLS   #$$M, a#DS$ERRHARD
                               1F80  2755 50$:     $DS_CKLOOP      20$             ; SCOPE LOOP?
        00000000'9F  FF44 CF   FA  1F80                        CALLG   20$, a#DS$CKLOOP
                               1F89  2756 $DS_ENDSUB
                               1F89       T20_S20_X:
    00000000'9F  000001F8'EF   FA  1F89                        CALLG   $S$, a#DS$ENDSUB
                               1F94  2757 $DS_ENDTEST
                      50   01  D0  1F94                        MOVL    #1, R0          ; NORMAL EXIT
                               1F97       TEST_020_X::
            00000000'9F  6E    FA  1F97                        CALLG   (SP), a#DS$BREAK
                         04    1F9E                        RET                         ; RETURN TO TEST SEQUENCER
                               1F9F  2758 $DS_PAGE
                               1F9F  2759 ;***** END OF TEST SECTION
                         00000000                          .PSECT  $TSTCNT, NOEXE, NOWRT, OVR, LONG
                     00000014  0000                        .LONG   $TN
                               0004  2761 .END
```

```
$$$                     = 000001F8 R     04
$$E                     = 00000001
$$M                     = 00000004
$$N                     = 00000000
$$S                     = FFFFFFFF
$ENV                    = 00000001  G
$ER                     = FFFFFFFF
$MO                     = 00000001  G
$S$                     = 000001F8 R     04
$ST                     = 00000000
$TN                     = 00000014
ALL                     = 00000001  G
ALL_ONES                  ********    X   08
ARRAY_MSG1                ********    X   05
ARRAY_MSG15               ********    X   08
ARRAY_MSG24               ********    X   08
ARRAY_MSG3                ********    X   05
ARRAY_MSG30               ********    X   05
ARRAY_MSG4                ********    X   02
ARRAY_MSG5                ********    X   05
ARRAY_MSG54               ********    X   02
ARRAY_MSG55               ********    X   07
ARRAY_MSG57               ********    X   05
ARRAY_MSG58               ********    X   05
ARRAY_MSG59               ********    X   05
ARRAY_MSG6                ********    X   05
ARRAY_MSG60               ********    X   05
ARRAY_MSG61               ********    X   02
ARRAY_MSG62               ********    X   05
ATTENTION                 ********    X   05
BASE_016                  00000000 R      02
BASE_017                  00000000 R      05
BASE_018                  00000000 R      06
BASE_019                  00000000 R      07
BASE_020                  00000000 R      08
BCR                       ********    X   02
BIT0                    = 00000001
BIT1                    = 00000002
BIT10                   = 00000400
BIT11                   = 00000800
BIT12                   = 00001000
BIT13                   = 00002000
BIT14                   = 00004000
BIT15                   = 00008000
BIT16                   = 00010000
BIT17                   = 00020000
BIT18                   = 00040000
BIT19                   = 00080000
BIT2                    = 00000004
BIT20                   = 00100000
BIT21                   = 00200000
BIT22                   = 00400000
BIT23                   = 00800000
BIT24                   = 01000000
BIT25                   = 02000000
BIT26                   = 04000000
BIT27                   = 08000000
BIT28                   = 10000000
BIT29                   = 20000000
BIT3                    = 00000008
BIT30                   = 40000000
BIT31                   = 80000000
BIT4                    = 00000010
BIT5                    = 00000020
BIT6                    = 00000040
BIT7                    = 00000080
BIT8                    = 00000100
BIT9                    = 00000200
BLKSND_COMD               ********    X   05
BYTE2                     ********    X   07
BYTE3                     ********    X   07
CEP_FUNCTIONAL          = 00000000
CEP_REPAIR              = 00000001
CPE                       ********    X   08
CR                        ********    X   02
DATA_016                  0000000C R      02
DATA_017                  00000014 R      05
DATA_018                  00000024 R      06
DATA_019                  00000020 R      07
DATA_020                  00000010 R      08
DATA_IBUFFER              ********    X   02
DATA_OBUFFER              ********    X   08
DATA_XFER_ABRT            ********    X   02
DATA_XFER_DONE            ********    X   02
DATA_XFER_LATE            ********    X   05
DEFAULT                 = 00000000  G
DMD                       ********    X   08
DOCC                      ********    X   08
DR                        ********    X   02
DRIVE0                    ********    X   02
DRIVES_PRESENT            ********    X   08
DRIVE_OFFSET              ********    X   05
DRV_ERRMASK               ********    X   08
DRV_INITMASK              ********    X   08
DS$BGNSUB                 ********    X   02
DS$BREAK                  ********    X   02
DS$CKLOOP                 ********    X   02
DS$CLRVEC                 ********    X   05
DS$ENDSUB                 ********    X   02
DS$ERRHARD                ********    X   02
DS$PRINTB                 ********    X   08
DS$PRINTF                 ********    X   08
DS$PRINTX                 ********    X   02
DS$SETVEC                 ********    X   05
DS$WAITUS                 ********    X   02
DSA$AL_APTMAIL            0000FE00
DSA$AT_APTTXT             0000FA00
DSA$GL_APTCOM             0000FE04
DSA$GL_DEVLEN             0000FE58
DSA$GL_ERRNO              0000FE44
DSA$GL_EVENT              0000FE48
DSA$GL_FLAGS              0000FE00
DSA$GL_MSGTYP             0000FE40
DSA$GL_PASSES             0000FE08
```

DSA$GL_PASSNO		0000FE54			INTERRUPT_EN		********	X	05
DSA$GL_SECTNO		0000FE10			INT_OCCURRED		********	X	05
DSA$GL_SID		0000FE14			INVRT_MAP_PAR		********	X	05
DSA$GL_SUBTNO		0000FE4C			INVRT_MB_CPAR		********	X	05
DSA$GL_TESTNO		0000FE50			INVRT_MB_DPAR		********	X	02
DSA$GL_UNITS		0000FE0C			IPLR		********	X	05
DSA$GQ_MSGPTR		0000FE68			ISR_TBL		********	X	05
DSA$GT_DEVNAM		0000FE5C			LOWBITS_MSK		********	X	08
DSA$M_TRACE	=	00000400			LUN		********	X	02
DS_MSK		********	X	08	MAINT_MODE		********	X	02
DT_ABORT		********	X	02	MAP_INVALID		********	X	02
DT_BUSY		********	X	05	MAP_OFFSET		********	X	02
DVA		********	X	08	MAP_PE		********	X	05
ENABLE_PS		********	X	08	MAP_PTR_MSK		********	X	05
ENV$M_DOMAIN	=	00C00002			MASS_CNTRL_PE		********	X	05
ENV$M_LEVEL	=	00000001			MASS_DATA_PE		********	X	05
ENV$M_SUPER	=	000003FC			MBE	=	00000002	G	
ENV$S_DOMAIN	=	00000001			MBE_ASR		********	X	08
ENV$S_LEVEL	=	00000001			MBE_CR1		********	X	08
ENV$S_SUPER	=	00000008			MBE_CR2		********	X	08
ENV$V_DOMAIN	=	00000001			MBE_CUR_ADR		********	X	08
ENV$V_LEVEL	=	00000000			MBE_DBR		********	X	08
ENV$V_SUPER	=	00000002			MBE_DTR		********	X	08
ENV$_CPU	=	00000000			MBE_ER		********	X	08
ENV$_FUNCTIONAL	=	00000000			MBE_MR		********	X	08
ENV$_REPAIR	=	00000001			MBE_PARAM		********	X	08
ENV$_SUPER	=	00000001			MBE_REG_MSG		********	X	08
ENV$_SYSTEM	=	00000001			MBE_SR		********	X	08
ERR_STAT		********	X	05	MCLK		********	X	08
EXP_INTERRUPT		********	X	05	MIOBUFFER		********	X	02
FMT_ALLDRVPRES		********	X	06	MISSED_XFER		********	X	05
FMT_ANY_REG		********	X	08	MM_XFER		********	X	05
FMT_ERRSUM		********	X	08	MSG_016		00000002	R	02
FMT_EXCP_ERR		********	X	02	MSG_017		00000002	R	05
FMT_MBE_ASR		********	X	08	MSG_018		00000002	R	06
FMT_MBE_CR1		********	X	08	MSG_019		00000002	R	07
FMT_MBE_ER		********	X	08	MSG_020		00000002	R	08
FMT_MBE_SR		********	X	08	NON_XIST_DRIVE		********	X	06
FMT_NOINTEST		********	X	05	NO_MBE_MSG		********	X	08
FMT_NO_MBE_SEL		********	X	08	NO_RESPONSE		********	X	05
FMT_SILOERROR		********	X	02	PF_FIELD		********	X	02
FMT_STATUS_REG		********	X	02	PGM_INIT		********	X	02
HP$A_DEPENDENT		00000032			PRINT_DTERR		********	X	08
HP$A_DEVICE		00000018			PRINT_IBCERR		********	X	07
HP$A_DVA		0000001C			PRINT_INTERR		********	X	05
HP$A_LINK		00000020			PRINT_NULL		********	X	08
HP$B_DRIVE		0000000B			PRINT_SBE		********	X	02
HP$B_FLAGS		0000000A			PRINT_SILOER		********	X	02
HP$B_RH750_BR		00000032			PRINT_VECTERR		********	X	05
HP$K_RH750_LEN		00000033			PROGRAM_ERROR		********	X	05
HP$Q_DEVICE		00000000			READ		********	X	02
HP$T_DEVICE		0000000C			READ_REV		********	X	08
HP$T_TYPE		00000026			REG_NAME		********	X	02
HP$W_SIZE		00000008			REG_NO		********	X	02
HP$W_VECTOR		00000024			REG_STRING		********	X	02
IBC_MSG		********	X	07	RH11TB_MSK		********	X	02
INH_BYTE_CNT		********	X	07	RH11Tb_PRES		********	X	02

```
RH750$B_BR                      00000032
RH750$K_LEN                     00000033
RHSR_CHECK                      ********  X   08
RHSR_MSG                        ********  X   02
RH_CUR_ADR                      ********  X   02
RH_ISR                          ********  X   05
RH_VECS                         ********  X   05
RH_VECTOR                       ********  X   05
RS_MSK                          ********  X   08
SEARCH                          ********  X   08
SEP_FUNCTIONAL                = 00000002
SEP_REPAIR                    = 00000003
SIM_ATTN                        ********  X   05
SIM_EBL                         ********  X   02
SIM_OCC                         ********  X   02
SIM_SCLK                        ********  X   02
SIZ...                        = 00000001
SR                              ********  X   02
T16_S1                          00000012 RG     02
T16_S1_X                        0000020F R      02
T16_S2                          0000021A RG     02
T16_S2_X                        00000401 R      02
T17_S1                          0000005F RG     05
T17_S10                         00000B5A RG     05
T17_S10_X                       00000C46 R      05
T17_S11                         00000C51 RG     05
T17_S11_X                       00000D70 R      05
T17_S12                         00000D7B RG     05
T17_S12_X                       00000EA1 R      05
T17_S13                         00000EAC RG     05
T17_S13_X                       00000FF1 R      05
T17_S14                         00000FFC RG     05
T17_S14_X                       00001126 R      05
T17_S15                         00001131 RG     05
T17_S15_X                       00001255 R      05
T17_S1_X                        00000232 R      05
T17_S2                          0000023D RG     05
T17_S2_X                        00000331 R      05
T17_S3                          0000033C RG     05
T17_S3_X                        000003D6 R      05
T17_S4                          000003E1 RG     05
T17_S4_X                        000004F0 R      05
T17_S5                          000004FB RG     05
T17_S5_X                        00000657 R      05
T17_S6                          00000662 RG     05
T17_S6_X                        0000078A R      05
T17_S7                          00000795 RG     05
T17_S7_X                        000008B4 R      05
T17_S8                          000008BF RG     05
T17_S8_X                        000009F2 R      05
T17_S9                          000009FD RG     05
T17_S9_X                        00000B4F R      05
T18_S1                          0000002A RG     06
T18_S1_X                        00000092 R      06
T18_S2                          0000009D RG     06
T18_S2_X                        00000121 R      06
T19_S1                          00000026 RG     07
T19_S1_X                        0000012A R      07
T19_S2                          00000135 RG     07
T19_S2_X                        000001EE R      07
T19_S3                          000001F9 RG     07
T19_S3_X                        000002C9 R      07
T19_S4                          000002D4 RG     07
T19_S4_X                        00000480 R      07
T20_S1                          00000016 RG     08
T20_S10                         00000C22 RG     08
T20_S10_X                       00000E91 R      08
T20_S11                         00000E9C RG     08
T20_S11_X                       0000110B R      08
T20_S12                         00001116 RG     08
T20_S12_X                       00001385 R      08
T20_S13                         00001390 RG     08
T20_S13_X                       000015F3 R      08
T20_S14                         000015FE RG     08
T20_S14_X                       00001865 R      08
T20_S15                         00001870 RG     08
T20_S15_X                       00001A56 R      08
T20_S16                         00001A61 RG     08
T20_S16_X                       00001B2C R      08
T20_S17                         00001B37 RG     08
T20_S17_X                       00001C47 R      08
T20_S18                         00001C52 RG     08
T20_S18_X                       00001D6F R      08
T20_S19                         00001D7A RG     08
T20_S19_X                       00001E78 R      08
T20_S1_X                        000001B1 R      08
T20_S2                          000001BC RG     08
T20_S20                         00001E83 RG     08
T20_S20_X                       00001F89 R      08
T20_S2_X                        0000028D R      08
T20_S3                          00000298 RG     08
T20_S3_X                        0000032D R      08
T20_S4                          00000338 RG     08
T20_S4_X                        000003D1 R      08
T20_S5                          000003DC RG     08
T20_S5_X                        000005A1 R      08
T20_S6                          000005AC RG     08
T20_S6_X                        000006A5 R      08
T20_S7                          000006B0 RG     08
T20_S7_X                        0000076E R      08
T20_S8                          00000779 RG     08
T20_S8_X                        000009C2 R      08
T20_S9                          000009CD RG     08
T20_S9_X                        00000C17 R      08
TEMP1                           ********    X   08
TEST_016                        00000010 RG     02
TEST_016_X                      0000040F RG     02
TEST_017                        00000018 RG     05
TEST_017_X                      00001263 RG     05
TEST_018                        00000028 RG     06
TEST_018_X                      0000012F RG     06
TEST_019                        00000024 RG     07
TEST_019_X                      0000048E RG     07
TEST_020                        00000014 RG     08
```

```
TEST_020_X                      00001F97 RG    08
TEST_VEC                        ********   X   05
TYPE_MBE                        ********   X   08
VALID_BIT                       ********   X   02
VAR                             ********   X   02
WRITE                           ********   X   02
WRITE_CHECK                     ********   X   05
WRITE_CHECKREV                  ********   X   08
WRITE_CHK_HIGH                  ********   X   05
WRITE_CHK_LOW                   ********   X   05
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	0000FE70 (65136.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
TEST_016	00000417 (1047.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
DISPATCH	00000078 (120.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
ARGLIST	00000204 (516.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG
TEST_017	0000126B (4715.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_018	00000137 (311.)	06 (6.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_019	00000496 (1174.)	07 (7.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
TEST_020	00001F9F (8095.)	08 (8.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE
$TSTCNT	00000004 (4.)	09 (9.)	NOPIC USR OVR REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG

Symbol Cross Reference

SYMBOL	VALUE	DEFINITION	REFERENCES...
$$$	=000001F8-R	2693 (44)	1011 (18) 1078 (19) 1098 (20) 1137 (21) 1182 (22) 1215 (23) 1252 (24) 1352 (25) 1416 (26) 1465 (27) 1499 (28) 1530 (29) 1613 (30) 1662 (31) 1703 (32) 171 (2) 1799 (33) 1895 (34) 1992 (35) 2089 (36) 2185 (37) 2282 (38) 2378 (39) 2455 (40) 246 (3) 2490 (41) 2559 (42) 2632 (43) 2693 (44) 358 (4) 431 (5) 465 (6) 501 (7) 538 (8) 591 (9) 637 (10) 680 (11) 724 (12) 780 (13) 817 (14) 859 (15) 904 (16) 961 (17)
$$E	=00000001	2693 (44)	1007 (18) 1053 (18) 1096 (19) 1121 (20) 1180 (21) 1213 (22) 1250 (23) 1319 (24) 1414 (25) 1463 (26) 1497 (27) 1528 (28) 1611 (29) 1660 (30) 1701 (31) 1797 (32) 1893 (33) 1990 (34) 2087 (35) 2183 (36) 2280 (37) 2376 (38) 244 (2) 2453 (39) 2488 (40) 2557 (41) 2630 (42) 2691 (43) 2756 (44) 327 (3) 426 (4) 460 (5) 496 (6) 533 (7) 586 (8) 632 (9) 675 (10) 719 (11) 775 (12) 812 (13) 857 (14) 902 (15) 957 (17)
$$M	=00000004	2754 (44)	1004 (17) 1036 (18) 1041 (18) 1051 (18) 1058 (19) 1094 (19) 1119 (20) 1126 (21) 1169 (21) 1178 (21) 1211 (22) 1248 (23) 1286 (24) 1292 (24) 1303 (24) 1309 (24) 1317 (24) 1324 (25) 1368 (25) 1391 (25) 1404 (25) 1412 (25) 1457 (26) 1494 (27) 150 (2) 1525 (28) 1554 (29) 1566 (29) 1579 (29) 1595 (29) 1609 (29) 1657 (30) 1689 (31) 1699 (31) 1743 (32) 1752 (32) 1776 (32) 1790 (32) 1839 (33) 1848 (33) 1872 (33) 1886 (33) 1933 (34) 1942 (34) 1966 (34) 1983 (34) 2030 (35) 2039 (35) 2063 (35) 208 (2) 2080 (35) 2126 (36) 2135 (36) 2159 (36) 2176 (36) 2223 (37) 2232 (37) 2258 (37) 2273 (37) 232 (2) 2320 (38) 2329 (38) 2354 (38) 2369 (38) 241 (2) 2414 (39) 2423 (39) 2450 (39) 2486 (40) 2553 (41) 2625 (42) 2688 (43) 2754 (44) 309 (3) 317 (3) 325 (3) 332 (4) 384 (4) 396 (4) 401 (4) 411 (4) 418 (4) 443 (5) 448 (5) 458 (5) 489 (6) 516 (7) 521 (7) 531 (7) 569 (8) 574 (8) 584 (8) 613 (9) 618 (9) 630 (9) 658 (10) 663 (10) 673 (10) 702 (11) 707 (11) 717 (11) 758 (12) 763 (12) 773 (12) 795 (13) 800 (13) 810 (13) 840 (14) 845 (14) 855 (14) 885 (15) 890 (15)

		900 (15)	939 (16)	944 (16)	954 (16)
		989 (17)	994 (17)		
$$N	=00000000	2754 (44)	1004 (17)	1036 (18)	1041 (18)	1051 (18)
			1094 (19)	#-1108 (20)	1119 (20)	1169 (21)
			1178 (21)	1211 (22)	1248 (23)	1286 (24)
			1292 (24)	1303 (24)	1309 (24)	1317 (24)
			1368 (25)	#-1374 (25)	1391 (25)	1404 (25)
			1412 (25)	1457 (26)	1494 (27)	1525 (28)
			1554 (29)	1566 (29)	1579 (29)	1595 (29)
			1609 (29)	1657 (30)	1689 (31)	1699 (31)
			1743 (32)	1752 (32)	1776 (32)	1790 (32)
			1795 (32)	1839 (33)	1848 (33)	1872 (33)
			1886 (33)	1891 (33)	1933 (34)	1942 (34)
			1966 (34)	1983 (34)	1988 (34)	2030 (35)
			2039 (35)	2063 (35)	208 (2)	2080 (35)
			2085 (35)	2126 (36)	2135 (36)	2159 (36)
			2176 (36)	2181 (36)	2223 (37)	2232 (37)
			2258 (37)	2273 (37)	2278 (37)	232 (2)
			2320 (38)	2329 (38)	235 (2)	2354 (38)
			2369 (38)	2374 (38)	241 (2)	2414 (39)
			#-242 (2)	2423 (39)	2450 (39)	2486 (40)
			2553 (41)	2625 (42)	2688 (43)	2754 (44)
			309 (3)	317 (3)	325 (3)	384 (4)
			396 (4)	401 (4)	411 (4)	418 (4)
			443 (5)	448 (5)	458 (5)	489 (6)
			#-494 (6)	516 (7)	521 (7)	531 (7)
			569 (8)	574 (8)	584 (8)	613 (9)
			618 (9)	630 (9)	658 (10)	663 (10)
			673 (10)	702 (11)	707 (11)	717 (11)
			758 (12)	763 (12)	773 (12)	795 (13)
			800 (13)	810 (13)	840 (14)	845 (14)
			855 (14)	88 (1)	885 (15)	890 (15)
			900 (15)	939 (16)	944 (16)	954 (16)
			989 (17)	994 (17)		
$$S	=FFFFFFFF	2757 (44)	1011 (18)	1057 (19)	1058 (19)	1078 (19)
			1098 (20)	1125 (21)	1126 (21)	1137 (21)
			1182 (22)	1215 (23)	1252 (24)	1323 (25)
			1324 (25)	1352 (25)	1416 (26)	1465 (27)
			149 (2)	1499 (28)	150 (2)	1530 (29)
			1613 (30)	1662 (31)	1703 (32)	171 (2)
			1799 (33)	1895 (34)	1992 (35)	2089 (36)
			2185 (37)	2282 (38)	2378 (39)	2455 (40)
			246 (3)	2490 (41)	2559 (42)	2632 (43)
			2693 (44)	331 (4)	332 (4)	358 (4)
			431 (5)	465 (6)	501 (7)	538 (8)
			591 (9)	637 (10)	680 (11)	724 (12)
			780 (13)	817 (14)	859 (15)	904 (16)
			961 (17)			
$ENV	=00000001	143 (1)				
$ER	=FFFFFFFF	2757 (44)	1011 (18)	1078 (19)	1098 (20)	1137 (21)
			1182 (22)	1215 (23)	1252 (24)	1352 (25)
			#-1368 (25)	1416 (26)	1465 (27)	1499 (28)
			1530 (29)	1613 (30)	1662 (31)	1703 (32)
			171 (2)	1799 (33)	1895 (34)	1992 (35)
			2089 (36)	2185 (37)	2282 (38)	2378 (39)
			2455 (40)	246 (3)	2490 (41)	2559 (42)
			2632 (43)	2693 (44)	358 (4)	431 (5)

```
                                                  465    (6)     501    (7)     538    (8)     591    (9)
                                                  637    (10)    680    (11)    724    (12)    780    (13)
                                                  817    (14)    859    (15)    904    (16)    961    (17)
$MO                 =00000001     2760   (44)     2760   (44)
$S$                 =000001F8-R   2693   (44)     1007   (18)    1011   (18)    1053   (18)    1078   (19)
                                                  1096   (19)    1098   (20)    1121   (20)    1137   (21)
                                                  1180   (21)    1182   (22)    1213   (22)    1215   (23)
                                                  1250   (23)    1252   (24)    1319   (24)    1352   (25)
                                                  1414   (25)    1416   (26)    1463   (26)    1465   (27)
                                                  1497   (27)    1499   (28)    1528   (28)    1530   (29)
                                                  1611   (29)    1613   (30)    1660   (30)    1662   (31)
                                                  1701   (31)    1703   (32)    171    (2)     1797   (32)
                                                  1799   (33)    1893   (33)    1895   (34)    1990   (34)
                                                  1992   (35)    2087   (35)    2089   (36)    2183   (36)
                                                  2185   (37)    2280   (37)    2282   (38)    2376   (38)
                                                  2378   (39)    244    (2)     2453   (39)    2455   (40)
                                                  246    (3)     2488   (40)    2490   (41)    2557   (41)
                                                  2559   (42)    2630   (42)    2632   (43)    2691   (43)
                                                  2693   (44)    2756   (44)    327    (3)     358    (4)
                                                  426    (4)     431    (5)     460    (5)     465    (6)
                                                  496    (6)     501    (7)     533    (7)     538    (8)
                                                  586    (8)     591    (9)     632    (9)     637    (10)
                                                  675    (10)    680    (11)    719    (11)    724    (12)
                                                  775    (12)    780    (13)    812    (13)    817    (14)
                                                  857    (14)    859    (15)    902    (15)    904    (16)
                                                  957    (17)    961    (17)
$ST                 =00000000     2757   (44)     1004   (17)    1005   (17)    1007   (18)    1011   (18)
                                                  1036   (18)    1041   (18)    1042   (18)    1051   (18)
                                                  1052   (18)    1053   (18)    1054   (18)    1058   (19)
                                                  1078   (19)    1094   (19)    1095   (19)    1096   (19)
                                                  1098   (20)    1119   (20)    1120   (20)    1121   (20)
                                                  1122   (20)    1126   (21)    1137   (21)    1169   (21)
                                                  1170   (21)    1178   (21)    1179   (21)    1180   (21)
                                                  1182   (22)    1211   (22)    1212   (22)    1213   (22)
                                                  1215   (23)    1248   (23)    1249   (23)    1250   (23)
                                                  1252   (24)    1286   (24)    1287   (24)    1292   (24)
                                                  1293   (24)    1303   (24)    1304   (24)    1309   (24)
                                                  1310   (24)    1317   (24)    1318   (24)    1319   (24)
                                                  1320   (24)    1324   (25)    1352   (25)    1368   (25)
                                                  1391   (25)    1392   (25)    1404   (25)    1405   (25)
                                                  1412   (25)    1413   (25)    1414   (25)    1416   (26)
                                                  1457   (26)    1458   (26)    1463   (26)    1465   (27)
                                                  1494   (27)    1495   (27)    1497   (27)    1499   (28)
                                                  150    (2)     1525   (28)    1526   (28)    1528   (28)
                                                  1530   (29)    1554   (29)    1555   (29)    1566   (29)
                                                  1567   (29)    1579   (29)    1580   (29)    1595   (29)
                                                  1596   (29)    1609   (29)    1610   (29)    1611   (29)
                                                  1613   (30)    1657   (30)    1658   (30)    1660   (30)
                                                  1662   (31)    1689   (31)    1690   (31)    1699   (31)
                                                  1700   (31)    1701   (31)    1703   (32)    171    (2)
                                                  1743   (32)    1752   (32)    1753   (32)    1776   (32)
                                                  1777   (32)    1790   (32)    1791   (32)    1797   (32)
                                                  1799   (33)    1839   (33)    1848   (33)    1849   (33)
                                                  1872   (33)    1873   (33)    1886   (33)    1887   (33)
                                                  1893   (33)    1895   (34)    1933   (34)    1942   (34)
                                                  1943   (34)    1966   (34)    1967   (34)    1983   (34)
                                                  1984   (34)    1990   (34)    1992   (35)    2030   (35)
```

2039 (35) 2040 (35) 2063 (35) 2064 (35)
208 (2) 2080 (35) 2081 (35) 2087 (35)
2089 (36) 209 (2) 2126 (36) 2135 (36)
2136 (36) 2159 (36) 2160 (36) 2176 (36)
2177 (36) 2183 (36) 2185 (37) 2223 (37)
2232 (37) 2233 (37) 2258 (37) 2259 (37)
2273 (37) 2274 (37) 2280 (37) 2282 (38)
232 (2) 2320 (38) 2329 (38) 2330 (38)
2354 (38) 2355 (38) 2369 (38) 2370 (38)
2376 (38) 2378 (39) 241 (2) 2414 (39)
2423 (39) 2424 (39) 243 (2) 244 (2)
2450 (39) 2451 (39) 2453 (39) 2455 (40)
246 (3) 2486 (40) 2487 (40) 2488 (40)
2490 (41) 2553 (41) 2554 (41) 2557 (41)
2559 (42) 2625 (42) 2626 (42) 2630 (42)
2632 (43) 2688 (43) 2690 (43) 2691 (43)
2693 (44) 2754 (44) 2755 (44) 2756 (44)
2757 (44) 309 (3) 310 (3) 317 (3)
318 (3) 325 (3) 326 (3) 327 (3)
328 (3) 332 (4) 358 (4) 384 (4)
385 (4) 396 (4) 401 (4) 402 (4)
411 (4) 412 (4) 418 (4) 419 (4)
426 (4) 431 (5) 443 (5) 448 (5)
449 (5) 458 (5) 459 (5) 460 (5)
465 (6) 489 (6) 490 (6) 496 (6)
501 (7) 516 (7) 521 (7) 522 (7)
531 (7) 532 (7) 533 (7) 538 (8)
569 (8) 574 (8) 575 (8) 584 (8)
585 (8) 586 (8) 591 (9) 613 (9)
618 (9) 619 (9) 630 (9) 631 (9)
632 (9) 637 (10) 658 (10) 663 (10)
664 (10) 673 (10) 674 (10) 675 (10)
680 (11) 702 (11) 707 (11) 708 (11)
717 (11) 718 (11) 719 (11) 724 (12)
758 (12) 763 (12) 764 (12) 773 (12)
774 (12) 775 (12) 780 (13) 795 (13)
800 (13) 801 (13) 810 (13) 811 (13)
812 (13) 817 (14) 840 (14) 845 (14)
846 (14) 855 (14) 856 (14) 857 (14)
859 (15) 885 (15) 890 (15) 891 (15)
900 (15) 901 (15) 902 (15) 904 (16)
939 (16) 944 (16) 945 (16) 954 (16)
955 (16) 957 (17) 961 (17) 989 (17)
994 (17) 995 (17)
$TN =00000014 2760 (44) 1004 (17) 1036 (18) 1041 (18) 1051 (18)
1054 (18) 1057 (19) 1058 (19) 1094 (19)
1119 (20) 1122 (20) 1125 (21) 1126 (21)
1169 (21) 1178 (21) 1211 (22) 1248 (23)
1286 (24) 1292 (24) 1303 (24) 1309 (24)
1317 (24) 1320 (24) 1323 (25) 1324 (25)
1368 (25) 1375 (25) 1391 (25) 1404 (25)
1412 (25) 1457 (26) 149 (2) 1494 (27)
150 (2) 1525 (28) 1554 (29) 1566 (29)
1579 (29) 1595 (29) 1609 (29) 1657 (30)
1689 (31) 1699 (31) 1743 (32) 1752 (32)
1776 (32) 1790 (32) 1839 (33) 1848 (33)
1872 (33) 1886 (33) 1933 (34) 1942 (34)

				1966 (34) 1983 (34) 2030 (35) 2039 (35)
				2063 (35) 208 (2) 2080 (35) 2126 (36)
				2135 (36) 2159 (36) 2176 (36) 2223 (37)
				2232 (37) 2258 (37) 2273 (37) 232 (2)
				2320 (38) 2329 (38) 2354 (38) 2369 (38)
				241 (2) 2414 (39) 2423 (39) 2450 (39)
				2486 (40) 2553 (41) 2625 (42) 2688 (43)
				2754 (44) 2757 (44) 2760 (44) 309 (3)
				317 (3) 325 (3) 328 (3) 331 (4)
				332 (4) 384 (4) 396 (4) 401 (4)
				411 (4) 418 (4) 443 (5) 448 (5)
				458 (5) 489 (6) 516 (7) 521 (7)
				531 (7) 569 (8) 574 (8) 584 (8)
				613 (9) 618 (9) 630 (9) 658 (10)
				663 (10) 673 (10) 702 (11) 707 (11)
				717 (11) 758 (12) 763 (12) 773 (12)
				795 (13) 800 (13) 810 (13) 840 (14)
				845 (14) 855 (14) 885 (15) 890 (15)
				900 (15) 939 (16) 944 (16) 954 (16)
				989 (17) 994 (17)
ALL	=00000001	88	(1)	1058 (19) 1126 (21) 1324 (25) 150 (2)
				332 (4)
ALL_ONES	00000000-XR			#-1544 (29)
ARRAY_MSG1	00000000-XR			396 (4)
ARRAY_MSG15	00000000-XR			1983 (34) 2080 (35) 2176 (36) 2273 (37)
				2369 (38) 2553 (41) 2625 (42)
ARRAY_MSG24	00000000-XR			2450 (39) 2486 (40) 2688 (43) 2754 (44)
ARRAY_MSG3	00000000-XR			1036 (18) 1041 (18) 1051 (18) 1286 (24)
				443 (5) 448 (5) 458 (5) 613 (9)
				618 (9) 630 (9)
ARRAY_MSG30	00000000-XR			418 (4)
ARRAY_MSG4	00000000-XR			1004 (17) 241 (2) 309 (3) 317 (3)
				325 (3) 521 (7) 800 (13) 810 (13)
				845 (14) 855 (14) 890 (15) 900 (15)
				944 (16) 954 (16) 994 (17)
ARRAY_MSG5	00000000-XR			1119 (20) 1211 (22) 1309 (24) 1317 (24)
				663 (10) 673 (10)
ARRAY_MSG54	00000000-XR			208 (2)
ARRAY_MSG55	00000000-XR			1303 (24)
ARRAY_MSG57	00000000-XR			516 (7) 795 (13) 840 (14) 885 (15)
				939 (16) 989 (17)
ARRAY_MSG58	00000000-XR			1094 (19) 489 (6) 658 (10)
ARRAY_MSG59	00000000-XR			569 (8) 702 (11) 758 (12)
ARRAY_MSG6	00000000-XR			574 (8) 584 (8) 707 (11) 717 (11)
				763 (12) 773 (12)
ARRAY_MSG60	00000000-XR			531 (7)
ARRAY_MSG61	00000000-XR			1169 (21) 1178 (21) 1248 (23) 1292 (24)
				232 (2) 401 (4)
ARRAY_MSG62	00000000-XR			384 (4) 411 (4)
ATTENTION	00000000-XR			#-1088 (19) #-1379 (25) #-436 (5)
BASE_016	00000000-R	149	(2)	150 (2)
BASE_017	00000000-R	331	(4)	332 (4)
BASE_018	00000000-R	1057	(19)	1058 (19)
BASE_019	00000000-R	1125	(21)	1126 (21)
BASE_020	00000000-R	1323	(25)	1324 (25)
BCR	00000000-XR			#-1023 (18) #-1152 (21) #-1161 (21) #-1197 (22)
				#-1229 (23) #-1241 (23) #-1274 (24) #-1639 (30)

```
                                                        #-1729   (32)   #-1761   (32)   #-1825   (33)   #-1857   (33)
                                                        #-186    (2)    #-1919   (34)   #-1951   (34)   #-2016   (35)
                                                        #-2048   (35)   #-2112   (36)   #-2144   (36)   #-215    (2)
                                                        #-2209   (37)   #-2241   (37)   #-2306   (38)   #-2338   (38)
                                                        #-2400   (39)   #-2432   (39)   #-2470   (40)   #-2535   (41)
                                                        #-2607   (42)   #-265    (3)    #-2673   (43)   #-2737   (44)
                                                        #-293    (3)    #-546    (8)    #-601    (9)    #-621    (9)
                                                        #-641    (10)   #-683    (11)   #-735    (12)   #-744    (12)
                                                        #-828    (14)   #-872    (15)   #-916    (16)   #-924    (16)
                                                        #-967    (17)   #-973    (17)
BIT...                  =00000003       143     (1)      143     (1)
BIT0                    =00000001       140     (1)
BIT1                    =00000002       140     (1)
BIT10                   =00000400       140     (1)
BIT11                   =00000800       140     (1)
BIT12                   =00001000       140     (1)    #-1017   (18)
BIT13                   =00002000       140     (1)
BIT14                   =00004000       140     (1)
BIT15                   =00008000       140     (1)    #-1514   (28)
BIT16                   =00010000       140     (1)
BIT17                   =00020000       140     (1)
BIT18                   =00040000       140     (1)
BIT19                   =00080000       140     (1)
BIT2                    =00000004       140     (1)
BIT20                   =00100000       140     (1)
BIT21                   =00200000       140     (1)
BIT22                   =00400000       140     (1)
BIT23                   =00800000       140     (1)
BIT24                   =01000000       140     (1)
BIT25                   =02000000       140     (1)
BIT26                   =04000000       140     (1)
BIT27                   =08000000       140     (1)
BIT28                   =10000000       140     (1)
BIT29                   =20000000       140     (1)
BIT3                    =00000008       140     (1)
BIT30                   =40000000       140     (1)
BIT31                   =80000000       140     (1)
BIT4                    =00000010       140     (1)
BIT5                    =00000020       140     (1)
BIT6                    =00000040       140     (1)
BIT7                    =00000080       140     (1)
BIT8                    =00000100       140     (1)
BIT9                    =00000200       140     (1)
BLKSND_COMD              00000000-XR                    #-554    (8)    #-684    (11)
BYTE2                    00000000-XR                    #-1162   (21)
BYTE3                    00000000-XR                    #-1162   (21)
CEP_FUNCTIONAL          =00000000       143     (1)
CEP_REPAIR              =00000001       143     (1)      143     (1)
CPE                      00000000-XR                    #-1603   (29)
CR                       00000000-XR                    #-1012   (18)   #-1013   (18)   #-1014   (18)   #-1081   (19)
                                                        #-1102   (20)   #-1145   (21)   #-1154   (21)   #-1155   (21)
                                                        #-1189   (22)   #-1199   (22)   #-1203   (22)   #-1222   (23)
                                                        #-1231   (23)   #-1232   (23)   #-1236   (23)   #-1263   (24)
                                                        #-1272   (24)   #-1273   (24)   #-1276   (24)   #-1297   (24)
                                                        #-1354   (25)   #-1380   (25)   #-1393   (25)   #-1437   (26)
                                                        #-1444   (26)   #-1481   (27)   #-1515   (28)   #-1527   (28)
                                                        #-1545   (29)   #-1556   (29)   #-1570   (29)   #-1585   (29)
```

```
                                                                  #-1629   (30)   #-1659   (30)   #-1679   (31)   #-1680   (31)
                                                                  #-1691   (31)   #-1713   (32)   #-1721   (32)   #-174    (2)
                                                                  #-1754   (32)   #-1810   (33)   #-1817   (33)   #-1850   (33)
                                                                  #-187    (2)    #-1903   (34)   #-1911   (34)   #-194    (2)
                                                                  #-1944   (34)   #-2000   (35)   #-2008   (35)   #-2041   (35)
                                                                  #-2096   (36)   #-2104   (36)   #-211    (2)    #-2137   (36)
                                                                  #-2192   (37)   #-2201   (37)   #-2234   (37)   #-2289   (38)
                                                                  #-2298   (38)   #-2331   (38)   #-2386   (39)   #-2392   (39)
                                                                  #-2425   (39)   #-2460   (40)   #-2498   (41)   #-2522   (41)
                                                                  #-253    (3)    #-254    (3)    #-2567   (42)   #-2593   (42)
                                                                  #-2640   (43)   #-2664   (43)   #-2701   (44)   #-271    (3)
                                                                  #-2727   (44)   #-275    (3)    #-276    (3)    #-372    (4)
                                                                  #-373    (4)    #-386    (4)    #-387    (4)    #-433    (5)
                                                                  #-434    (5)    #-435    (5)    #-474    (6)    #-483    (6)
                                                                  #-502    (7)    #-503    (7)    #-506    (7)    #-539    (8)
                                                                  #-540    (8)    #-548    (8)    #-592    (9)    #-593    (9)
                                                                  #-594    (9)    #-638    (10)   #-643    (10)   #-644    (10)
                                                                  #-681    (11)   #-682    (11)   #-687    (11)   #-725    (12)
                                                                  #-726    (12)   #-742    (12)   #-781    (13)   #-782    (13)
                                                                  #-784    (13)   #-818    (14)   #-819    (14)   #-826    (14)
                                                                  #-863    (15)   #-864    (15)   #-870    (15)   #-908    (16)
                                                                  #-909    (16)   #-910    (16)   #-962    (17)   #-963    (17)
                                                                  #-964    (17)
DATA_016                        0000000C-R   150     (2)            150    (2)
DATA_017                        00000014-R   332     (4)            332    (4)
DATA_018                        00000024-R   1058    (19)           1058   (19)
DATA_019                        00000020-R   1126    (21)           1126   (21)
DATA_020                        00000010-R   1324    (25)           1324   (25)
DATA_IBUFFER                    00000000-XR                         1264   (24)   #-1718   (32)     1755   (32)   #-1781   (32)
                                                                    1788   (32)   #-1814   (33)     1851   (33)   #-1877   (33)
                                                                    1884   (33)   #-1908   (34)     1945   (34)   #-1973   (34)
                                                                    1980   (34)   #-2005   (35)     2042   (35)   #-2070   (35)
                                                                    2077   (35)   #-2101   (36)     2138   (36)   #-2166   (36)
                                                                    2173   (36)   #-2198   (37)     2235   (37)   #-2263   (37)
                                                                    2270   (37)   #-2295   (38)     2332   (38)   #-2359   (38)
                                                                    2366   (38)     2523   (41)     2527   (41)   #-2546   (41)
                                                                    2550   (41)     2594   (42)     2598   (42)   #-2618   (42)
                                                                    2622   (42)     277    (3)
DATA_OBUFFER                    00000000-XR                       #-1630   (30)     1631   (30)   #-1717   (32)   #-1720   (32)
                                                                    1722   (32)   #-1780   (32)     1787   (32)   #-1813   (33)
                                                                  #-1816   (33)     1818   (33)   #-1876   (33)     1883   (33)
                                                                  #-1907   (34)   #-1910   (34)     1912   (34)   #-1972   (34)
                                                                    1979   (34)   #-2004   (35)   #-2007   (35)     2009   (35)
                                                                  #-2069   (35)     2076   (35)   #-2100   (36)   #-2103   (36)
                                                                    2105   (36)   #-2165   (36)     2172   (36)   #-2197   (37)
                                                                    2202   (37)   #-2262   (37)     2269   (37)   #-2294   (38)
                                                                    2299   (38)   #-2358   (38)     2365   (38)   #-2390   (39)
                                                                    2393   (39)     2426   (39)   #-2438   (39)   #-2462   (40)
                                                                    2464   (40)     2506   (41)   #-2545   (41)     2549   (41)
                                                                    2575   (42)   #-2617   (42)     2621   (42)     2648   (43)
                                                                    2665   (43)     2709   (44)     2728   (44)
DATA_XFER_ABRT                  00000000-XR                       #-1029   (18)   #-201    (2)    #-291    (3)    #-390    (4)
                                                                  #-505    (7)    #-550    (8)    #-607    (9)    #-696    (11)
                                                                  #-750    (12)   #-833    (14)   #-878    (15)   #-932    (16)
                                                                  #-981    (17)
DATA_XFER_DONE                  00000000-XR                       #-1030   (18)   #-200    (2)    #-292    (3)    #-390    (4)
                                                                  #-505    (7)    #-550    (8)    #-607    (9)    #-642    (10)
```

```
                                                #-696    (11)    #-751    (12)    #-834    (14)    #-879    (15)
                                                #-933    (16)    #-982    (17)
DATA_XFER_LATE                   00000000-XR    #-549    (8)     #-695    (11)
DEFAULT                         =00000000   88  (1)      1058    (19)      1126    (21)      1324    (25)      150    (2)
                                                 332    (4)
DMD                              00000000-XR    #-1645   (30)
DOCC                             00000000-XR    #-1644   (30)
DR                               00000000-XR    #-1157   (21)    #-1159   (21)    #-1160   (21)    #-1201   (22)
                                                #-1234   (23)    #-1238   (23)    #-1239   (23)    #-1279   (24)
                                                #-188    (2)     #-191    (2)     #-192    (2)     #-193    (2)
                                                #-195    (2)     #-196    (2)     #-217    (2)     #-219    (2)
                                                #-220    (2)     #-236    (2)     #-237    (2)     #-238    (2)
                                                #-268    (3)     #-269    (3)     #-270    (3)     #-272    (3)
                                                #-273    (3)     #-296    (3)     #-298    (3)     #-299    (3)
                                                #-301    (3)     #-302    (3)     #-376    (4)     #-388    (4)
                                                #-389    (4)     #-437    (5)     #-439    (5)     #-509    (7)
                                                #-510    (7)     #-552    (8)     #-554    (8)     #-557    (8)
                                                #-558    (8)     #-561    (8)     #-562    (8)     #-646    (10)
                                                #-648    (10)    #-649    (10)    #-650    (10)    #-651    (10)
                                                #-684    (11)    #-689    (11)    #-690    (11)    #-691    (11)
                                                #-693    (11)    #-694    (11)    #-727    (12)    #-740    (12)
                                                #-783    (13)    #-789    (13)    #-820    (14)
DRIVE0                           00000000-XR    #-1043   (18)    #-1100   (20)    #-1153   (21)    #-1198   (22)
                                                #-1230   (23)    #-1277   (24)    #-1440   (26)    #-189    (2)
                                                #-210    (2)     #-266    (3)     #-294    (3)     #-374    (4)
                                                #-403    (4)     #-472    (6)     #-504    (7)     #-523    (7)
                                                #-576    (8)     #-622    (9)     #-709    (11)    #-785    (13)
                                                #-892    (15)    #-946    (16)    #-996    (17)
DRIVES_PRESENT                   00000000-XR      1431   (26)
DRIVE_OFFSET                     00000000-XR    #-1106   (20)    #-1460   (26)    #-492    (6)
DRV_ERRMASK                      00000000-XR    #-1396   (25)
DRV_INITMASK                     00000000-XR    #-1407   (25)
DS$BGNSUB                        00000000-XR      1011   (18)      1078   (19)      1098   (20)      1137   (21)
                                                 1182   (22)      1215   (23)      1252   (24)      1352   (25)
                                                 1416   (26)      1465   (27)      1499   (28)      1530   (29)
                                                 1613   (30)      1662   (31)      1703   (32)      171    (2)
                                                 1799   (33)      1895   (34)      1992   (35)      2089   (36)
                                                 2185   (37)      2282   (38)      2378   (39)      2455   (40)
                                                 246    (3)       2490   (41)      2559   (42)      2632   (43)
                                                 2693   (44)      358    (4)       431    (5)       465    (6)
                                                 501    (7)       538    (8)       591    (9)       637    (10)
                                                 680    (11)      724    (12)      780    (13)      817    (14)
                                                 859    (15)      904    (16)      961    (17)
DS$BREAK                         00000000-XR      1054   (18)      1122   (20)      1320   (24)      2757   (44)
                                                 328    (3)
DS$CKLOOP                        00000000-XR      1005   (17)      1042   (18)      1052   (18)      1095   (19)
                                                 1120   (20)      1170   (21)      1179   (21)      1212   (22)
                                                 1249   (23)      1287   (24)      1293   (24)      1304   (24)
                                                 1310   (24)      1318   (24)      1392   (25)      1405   (25)
                                                 1413   (25)      1458   (26)      1495   (27)      1526   (28)
                                                 1555   (29)      1567   (29)      1580   (29)      1596   (29)
                                                 1610   (29)      1658   (30)      1690   (31)      1700   (31)
                                                 1753   (32)      1777   (32)      1791   (32)      1849   (33)
                                                 1873   (33)      1887   (33)      1943   (34)      1967   (34)
                                                 1984   (34)      2040   (35)      2064   (35)      2081   (35)
                                                 209    (2)       2136   (36)      2160   (36)      2177   (36)
                                                 2233   (37)      2259   (37)      2274   (37)      2330   (38)
```

		2355 (38) 2370 (38) 2424 (39) 243 (2)
		2451 (39) 2487 (40) 2554 (41) 2626 (42)
		2690 (43) 2755 (44) 310 (3) 318 (3)
		326 (3) 385 (4) 402 (4) 412 (4)
		419 (4) 449 (5) 459 (5) 490 (6)
		522 (7) 532 (7) 575 (8) 585 (8)
		619 (9) 631 (9) 664 (10) 674 (10)
		708 (11) 718 (11) 764 (12) 774 (12)
		801 (13) 811 (13) 846 (14) 856 (14)
		891 (15) 901 (15) 945 (16) 955 (16)
		995 (17)
DS$CLRVEC	00000000-XR	363 (4) 422 (4)
DS$ENDSUB	00000000-XR	1007 (18) 1053 (18) 1096 (19) 1121 (20)
		1180 (21) 1213 (22) 1250 (23) 1319 (24)
		1414 (25) 1463 (26) 1497 (27) 1528 (28)
		1611 (29) 1660 (30) 1701 (31) 1797 (32)
		1893 (33) 1990 (34) 2087 (35) 2183 (36)
		2280 (37) 2376 (38) 244 (2) 2453 (39)
		2488 (40) 2557 (41) 2630 (42) 2691 (43)
		2756 (44) 327 (3) 426 (4) 460 (5)
		496 (6) 533 (7) 586 (8) 632 (9)
		675 (10) 719 (11) 775 (12) 812 (13)
		857 (14) 902 (15) 957 (17)
DS$ERRHARD	00000000-XR	1004 (17) 1036 (18) 1041 (18) 1051 (18)
		1094 (19) 1119 (20) 1169 (21) 1178 (21)
		1211 (22) 1248 (23) 1286 (24) 1292 (24)
		1303 (24) 1309 (24) 1317 (24) 1368 (25)
		1391 (25) 1404 (25) 1412 (25) 1457 (26)
		1494 (27) 1525 (28) 1554 (29) 1566 (29)
		1579 (29) 1595 (29) 1609 (29) 1657 (30)
		1689 (31) 1699 (31) 1743 (32) 1752 (32)
		1776 (32) 1790 (32) 1839 (33) 1848 (33)
		1872 (33) 1886 (33) 1933 (34) 1942 (34)
		1966 (34) 1983 (34) 2030 (35) 2039 (35)
		2063 (35) 208 (2) 2080 (35) 2126 (36)
		2135 (36) 2159 (36) 2176 (36) 2223 (37)
		2232 (37) 2258 (37) 2273 (37) 232 (2)
		2320 (38) 2329 (38) 2354 (38) 2369 (38)
		241 (2) 2414 (39) 2423 (39) 2450 (39)
		2486 (40) 2553 (41) 2625 (42) 2688 (43)
		2754 (44) 309 (3) 317 (3) 325 (3)
		384 (4) 396 (4) 401 (4) 411 (4)
		418 (4) 443 (5) 448 (5) 458 (5)
		489 (6) 516 (7) 521 (7) 531 (7)
		569 (8) 574 (8) 584 (8) 613 (9)
		618 (9) 630 (9) 658 (10) 663 (10)
		673 (10) 702 (11) 707 (11) 717 (11)
		758 (12) 763 (12) 773 (12) 795 (13)
		800 (13) 810 (13) 840 (14) 845 (14)
		855 (14) 885 (15) 890 (15) 900 (15)
		939 (16) 944 (16) 954 (16) 989 (17)
		994 (17)
DS$PRINTB	00000000-XR	1795 (32) 1891 (33) 1988 (34) 2085 (35)
		2181 (36) 2278 (37) 2374 (38)
DS$PRINTF	00000000-XR	1374 (25)
DS$PRINTX	00000000-XR	1108 (20) 235 (2) 242 (2) 494 (6)
DS$SETVEC	00000000-XR	364 (4) 432 (5)

```
DS$WAITUS                       00000000-XR                1082    (19)     1158    (21)     1202    (22)     1235    (23)
                                                           1517    (28)     1647    (30)     1650    (30)     1735    (32)
                                                           1767    (32)     1831    (33)     1863    (33)     1925    (34)
                                                           1957    (34)     2022    (35)     2054    (35)     2118    (36)
                                                           2150    (36)     2215    (37)     222     (2)      2249    (37)
                                                           2312    (38)     2346    (38)     2406    (39)     2441    (39)
                                                           2477    (40)     2543    (41)     2615    (42)     2681    (43)
                                                           2746    (44)     508     (7)      553     (8)      563     (8)
                                                           752     (12)
DSA$GL_FLAGS                     0000FE00                 #-1364   (25)     #-1372   (25)
DSA$GL_PASSNO                    0000FE54                 #-1362   (25)     #-1370   (25)
DSA$M_TRACE                     =00000400                 #-1364   (25)     #-1372   (25)
DS_MSK                           00000000-XR              #-1449   (26)
DT_ABORT                         00000000-XR              #-194    (2)      #-211    (2)      #-271    (3)      #-387    (4)
DT_BUSY                          00000000-XR              #-1045   (18)     #-1190   (22)     #-377    (4)      #-404    (4)
                                                          #-524    (7)      #-578    (8)      #-623    (9)      #-710    (11)
                                                          #-893    (15)     #-947    (16)     #-998    (17)
DVA                              00000000-XR              #-1383   (25)
ENABLE_PS                        00000000-XR              #-2247   (37)     #-2344   (38)     #-2541   (41)     #-2613   (42)
                                                          #-2679   (43)     #-2743   (44)
ENV$M_DOMAIN                    =00000002      143  (1)
ENV$M_LEVEL                     =00000001      143  (1)
ENV$M_SUPER                     =000003FC      143  (1)
ENV$S_DOMAIN                    =00000001      143  (1)
ENV$S_LEVEL                     =00000001      143  (1)
ENV$S_SUPER                     =00000008      143  (1)
ENV$V_DOMAIN                    =00000001      143  (1)       143    (1)
ENV$V_LEVEL                     =00000000      143  (1)       143    (1)
ENV$V_SUPER                     =00000002      143  (1)
ENV$_CPU                        =00000000      143  (1)       143    (1)
ENV$_FUNCTIONAL                 =00000000      143  (1)       143    (1)
ENV$_REPAIR                     =00000001      143  (1)       143    (1)
ENV$_SUPER                      =00000001      143  (1)
ENV$_SYSTEM                     =00000001      143  (1)       143    (1)
ERR_STAT                         00000000-XR              #-608    (9)
EXP_INTERRUPT                    00000000-XR              #-1014   (18)     #-1270   (24)     #-1295   (24)     #-386    (4)
                                                          #-435    (5)      #-483    (6)      #-506    (7)      #-548    (8)
                                                          #-594    (9)      #-644    (10)     #-687    (11)     #-742    (12)
                                                          #-784    (13)     #-826    (14)     #-870    (15)     #-910    (16)
                                                          #-964    (17)
FMT_ALLDRVPRES                   00000000-XR              1108    (20)
FMT_ANY_REG                      00000000-XR              1542    (29)     1969    (34)     2066    (35)     2162    (36)
FMT_ERRSUM                       00000000-XR              1795    (32)     1891    (33)     1988    (34)     2085    (35)
                                                          2181    (36)     2278    (37)     2374    (38)
FMT_EXCP_ERR                     00000000-XR              242     (2)
FMT_MBE_ASR                      00000000-XR              1435    (26)     1478    (27)
FMT_MBE_CR1                      00000000-XR              1382    (25)
FMT_MBE_ER                       00000000-XR              1600    (29)     1745    (32)     1841    (33)     1935    (34)
                                                          2032    (35)     2128    (36)     2225    (37)     2322    (38)
                                                          2416    (39)
FMT_MBE_SR                       00000000-XR              1395    (25)     1512    (28)     1711    (32)     1807    (33)
                                                          1901    (34)     1998    (35)     2094    (36)     2190    (37)
                                                          2287    (38)     2384    (39)
FMT_NOINTEST                     00000000-XR              494     (6)
FMT_NO_MBE_SEL                   00000000-XR              1374    (25)
FMT_SILOERROR                    00000000-XR              235     (2)
FMT_STATUS_REG                   00000000-XR              1625    (30)     1676    (31)     1769    (32)     1865    (33)
```

```
                                        1959  (34)     199  (2)      2056  (35)     2152  (36)
                                        2251  (37)    2347  (38)     2442  (39)     2478  (40)
                                        2682  (43)    2747  (44)      348  (4)
IBC_MSG                 00000000-XR   #-1165  (21)  #-1174  (21)  #-1208  (22)  #-1244  (23)
                                      #-1283  (24)  #-1289  (24)  #-1300  (24)  #-1306  (24)
                                      #-1313  (24)
INH_BYTE_CNT            00000000-XR   #-1154  (21)  #-1203  (22)  #-1231  (23)  #-1236  (23)
                                      #-1273  (24)
INTERRUPT_EN            00000000-XR   #-1014  (18)  #-1276  (24)  #-1297  (24)  #-386   (4)
                                      #-435   (5)   #-483   (6)   #-506   (7)   #-548   (8)
                                      #-594   (9)   #-644   (10)  #-687   (11)  #-742   (12)
                                      #-784   (13)  #-826   (14)  #-870   (15)  #-910   (16)
                                      #-964   (17)
INT_OCCURRED            00000000-XR   #-1014  (18)    1033  (18)  #-1271  (24)    1282  (24)
                                      #-1296  (24)    1299  (24)  #-386   (4)     393   (4)
                                      #-435   (5)     440   (5)   #-483   (6)     485   (6)
                                      #-506   (7)     513   (7)   #-548   (8)     566   (8)
                                      #-594   (9)     610   (9)   #-644   (10)    655   (10)
                                      #-687   (11)    699   (11)  #-742   (12)    755   (12)
                                      #-784   (13)    792   (13)  #-826   (14)    837   (14)
                                      #-870   (15)    882   (15)  #-910   (16)    936   (16)
                                      #-964   (17)    986   (17)
INVRT_MAP_PAR           00000000-XR   #-820   (14)
INVRT_MB_CPAR           00000000-XR   #-783   (13)
INVRT_MB_DPAR           00000000-XR   #-188   (2)   #-727   (12)  #-740   (12)
IPLR                    00000000-XR   #-1014  (18)  #-1280  (24)  #-386   (4)   #-435   (5)
                                      #-483   (6)   #-506   (7)   #-548   (8)   #-594   (9)
                                      #-644   (10)  #-687   (11)  #-742   (12)  #-784   (13)
                                      #-826   (14)  #-870   (15)  #-910   (16)  #-964   (17)
ISR_TBL                 00000000-XR     362  (4)
LOWBITS_MSK             00000000-XR   #-1406  (25)
LUN                     00000000-XR   #-1004  (17)  #-1036  (18)  #-1041  (18)  #-1051  (18)
                                      #-1094  (19)  #-1119  (20)  #-1169  (21)  #-1178  (21)
                                      #-1211  (22)  #-1248  (23)  #-1286  (24)  #-1292  (24)
                                      #-1303  (24)  #-1309  (24)  #-1317  (24)  #-1368  (25)
                                      #-1391  (25)  #-1404  (25)  #-1412  (25)  #-1457  (26)
                                      #-1494  (27)  #-1525  (28)  #-1554  (29)  #-1566  (29)
                                      #-1579  (29)  #-1595  (29)  #-1609  (29)  #-1657  (30)
                                      #-1689  (31)  #-1699  (31)  #-1743  (32)  #-1752  (32)
                                      #-1776  (32)  #-1790  (32)  #-1839  (33)  #-1848  (33)
                                      #-1872  (33)  #-1886  (33)  #-1933  (34)  #-1942  (34)
                                      #-1966  (34)  #-1983  (34)  #-2030  (35)  #-2039  (35)
                                      #-2063  (35)  #-208   (2)   #-2080  (35)  #-2126  (36)
                                      #-2135  (36)  #-2159  (36)  #-2176  (36)  #-2223  (37)
                                      #-2232  (37)  #-2258  (37)  #-2273  (37)  #-232   (2)
                                      #-2320  (38)  #-2329  (38)  #-2354  (38)  #-2369  (38)
                                      #-241   (2)   #-2414  (39)  #-2423  (39)  #-2450  (39)
                                      #-2486  (40)  #-2553  (41)  #-2625  (42)  #-2688  (43)
                                      #-2754  (44)  #-309   (3)   #-317   (3)   #-325   (3)
                                      #-384   (4)   #-396   (4)   #-401   (4)   #-411   (4)
                                      #-418   (4)   #-443   (5)   #-448   (5)   #-458   (5)
                                      #-489   (6)   #-516   (7)   #-521   (7)   #-531   (7)
                                      #-569   (8)   #-574   (8)   #-584   (8)   #-613   (9)
                                      #-618   (9)   #-630   (9)   #-658   (10)  #-663   (10)
                                      #-673   (10)  #-702   (11)  #-707   (11)  #-717   (11)
                                      #-758   (12)  #-763   (12)  #-773   (12)  #-795   (13)
                                      #-800   (13)  #-810   (13)  #-840   (14)  #-845   (14)
```

```
                                                #-855    (14)   #-885    (15)   #-890    (15)   #-900    (15)
                                                #-939    (16)   #-944    (16)   #-954    (16)   #-989    (17)
                                                #-994    (17)
MAINT_MODE                      00000000-XR     #-1013   (18)   #-1155   (21)   #-1199   (22)   #-1232   (23)
                                                #-1272   (24)   #-1680   (31)   #-187    (2)    #-254    (3)
                                                #-276    (3)    #-373    (4)    #-434    (5)    #-503    (7)
                                                #-540    (8)    #-593    (9)    #-643    (10)   #-682    (11)
                                                #-726    (12)   #-782    (13)   #-819    (14)   #-864    (15)
                                                #-909    (16)   #-963    (17)
MAP_INVALID                     00000000-XR     #-291    (3)    #-878    (15)
MAP_OFFSET                      00000000-XR     #-1149   (21)   #-1194   (22)   #-1226   (23)   #-1267   (24)
                                                #-1628   (30)   #-1715   (32)   #-180    (2)    #-1811   (33)
                                                #-1905   (34)   #-2002   (35)   #-2098   (36)   #-2194   (37)
                                                #-2291   (38)   #-2388   (39)   #-2531   (41)   #-2602   (42)
                                                #-262    (3)    #-2669   (43)   #-2732   (44)   #-280    (3)
                                                #-353    (4)    #-541    (8)
MAP_PE                          00000000-XR     #-833    (14)
MAP_PTR_MSK                     00000000-XR     #-1020   (18)   #-1635   (30)   #-1726   (32)   #-1759   (32)
                                                #-1822   (33)   #-1855   (33)   #-1916   (34)   #-1949   (34)
                                                #-2013   (35)   #-2046   (35)   #-2109   (36)   #-2142   (36)
                                                #-2206   (37)   #-2239   (37)   #-2303   (38)   #-2336   (38)
                                                #-2397   (39)   #-2533   (41)   #-2604   (42)   #-2671   (43)
                                                #-2734   (44)   #-598    (9)    #-733    (12)   #-825    (14)
                                                #-867    (15)   #-914    (16)
MASS_CNTRL_PE                   00000000-XR     #-790    (13)
MASS_DATA_PE                    00000000-XR     #-750    (12)
MBE                             =00000002  88  (1)    1324     (25)
MBE_ASR                         00000000-XR     #-1448   (26)   #-1455   (26)   #-1485   (27)   #-1492   (27)
MBE_CR1                         00000000-XR     #-1384   (25)   #-1389   (25)   #-1516   (28)   #-1646   (30)
                                                #-1734   (32)   #-1766   (32)   #-1830   (33)   #-1862   (33)
                                                #-1924   (34)   #-1956   (34)   #-2021   (35)   #-2053   (35)
                                                #-2117   (36)   #-2149   (36)   #-2214   (37)   #-2248   (37)
                                                #-2311   (38)   #-2345   (38)   #-2405   (39)   #-2439   (39)
                                                #-2476   (40)   #-2542   (41)   #-2614   (42)   #-2680   (43)
                                                #-2744   (44)
MBE_CR2                         00000000-XR     #-1445   (26)   #-1640   (30)   #-1730   (32)   #-1762   (32)
                                                #-1826   (33)   #-1858   (33)   #-1920   (34)   #-1952   (34)
                                                #-2017   (35)   #-2049   (35)   #-2113   (36)   #-2145   (36)
                                                #-2210   (37)   #-2243   (37)   #-2307   (38)   #-2340   (38)
                                                #-2401   (39)   #-2434   (39)   #-2472   (40)   #-2537   (41)
                                                #-2609   (42)   #-2675   (43)   #-2739   (44)
MBE_CUR_ADR                     00000000-XR     #-1355   (25)   #-1438   (26)   #-1479   (27)   #-1513   (28)
                                                #-1543   (29)   #-1627   (30)   #-1678   (31)   #-1714   (32)
                                                #-1809   (33)   #-1904   (34)   #-2001   (35)   #-2097   (36)
                                                #-2193   (37)   #-2290   (38)   #-2387   (39)   #-2507   (41)
                                                #-2530   (41)   #-2601   (42)   #-2649   (43)   #-2668   (43)
                                                #-2731   (44)
MBE_DBR                         00000000-XR     #-1546   (29)   #-1547   (29)   #-1552   (29)   #-1558   (29)
                                                #-1559   (29)   #-1564   (29)   #-1571   (29)   #-1572   (29)
                                                #-1577   (29)   #-1586   (29)   #-1587   (29)   #-1593   (29)
                                                #-2242   (37)   #-2339   (38)   #-2433   (39)   #-2471   (40)
                                                #-2536   (41)   #-2608   (42)   #-2674   (43)   #-2738   (44)
MBE_DTR                         00000000-XR     #-1358   (25)
MBE_ER                          00000000-XR     #-1601   (29)   #-1607   (29)   #-1682   (31)   #-1687   (31)
                                                #-1692   (31)   #-1697   (31)   #-1746   (32)   #-1750   (32)
                                                #-1842   (33)   #-1846   (33)   #-1936   (34)   #-1940   (34)
                                                #-2033   (35)   #-2037   (35)   #-2129   (36)   #-2133   (36)
```

					#-2226	(37)	#-2230	(37)	#-2323	(38)	#-2327	(38)			
					#-2417	(39)	#-2421	(39)							
MBE_MR	00000000-XR				#-1644	(30)	#-1645	(30)	#-1648	(30)	#-1649	(30)			
					#-2247	(37)	#-2344	(38)	#-2541	(41)	#-2613	(42)			
					#-2679	(43)	#-2743	(44)							
MBE_PARAM	00000000-XR				#-1730	(32)	#-1762	(32)	#-1826	(33)	#-1858	(33)			
					#-1920	(34)	#-1952	(34)	#-2017	(35)	#-2049	(35)			
					#-2113	(36)	#-2145	(36)	#-2210	(37)	#-2243	(37)			
					#-2307	(38)	#-2340	(38)	#-2401	(39)	#-2434	(39)			
					#-2472	(40)									
MBE_REG_MSG	00000000-XR				1382	(25)	1395	(25)	1435	(26)	1478	(27)			
					1512	(28)	1542	(29)	1600	(29)	1711	(32)			
					1745	(32)	1807	(33)	1841	(33)	1901	(34)			
					1935	(34)	1969	(34)	1998	(35)	2032	(35)			
					2066	(35)	2094	(36)	2128	(36)	2162	(36)			
					2190	(37)	2225	(37)	2287	(38)	2322	(38)			
					2384	(39)	2416	(39)							
MBE_SR	00000000-XR				#-1397	(25)	#-1402	(25)	#-1518	(28)	#-1523	(28)			
					#-1736	(32)	#-1739	(32)	#-1832	(33)	#-1835	(33)			
					#-1926	(34)	#-1929	(34)	#-2023	(35)	#-2026	(35)			
					#-2119	(36)	#-2122	(36)	#-2216	(37)	#-2219	(37)			
					#-2313	(38)	#-2316	(38)	#-2407	(39)	#-2410	(39)			
MCLK	00000000-XR				#-1648	(30)	#-1649	(30)							
MIOBUFFER	00000000-XR				1146	(21)	1191	(22)	1223	(23)	175	(2)			
					#-213	(2)	#-214	(2)	#-223	(2)	#-224	(2)			
					233	(2)	255	(3)	350	(4)	729	(12)			
					821	(14)	865	(15)	911	(16)	#-925	(16)			
					#-965	(17)	#-974	(17)							
MISSED_XFER	00000000-XR				#-1652	(30)	#-505	(7)							
MM_XFER	00000000-XR				1027	(18)	605	(9)	739	(12)	748	(12)			
					832	(14)	877	(15)	922	(16)	931	(16)			
					971	(17)	979	(17)							
MSG_016	00000002-R	149	(2)		150	(2)									
MSG_017	00000002-R	331	(4)		332	(4)									
MSG_018	00000002-R	1057	(19)		1058	(19)									
MSG_019	00000002-R	1125	(21)		1126	(21)									
MSG_020	00000002-R	1323	(25)		1324	(25)									
NON_XIST_DRIVE	00000000-XR				#-1111	(20)	#-1459	(26)							
NO_MBE_MSG	00000000-XR				1368	(25)									
NO_RESPONSE	00000000-XR				#-1029	(18)									
PF_FIELD	00000000-XR				#-1018	(18)	#-1147	(21)	#-1192	(22)	#-1224	(23)			
					#-1265	(24)	#-177	(2)	#-260	(3)	#-278	(3)			
					#-596	(9)	#-730	(12)							
PGM_INIT	00000000-XR				#-1012	(18)	#-1081	(19)	#-1102	(20)	#-1145	(21)			
					#-1189	(22)	#-1222	(23)	#-1263	(24)	#-1354	(25)			
					#-1380	(25)	#-1393	(25)	#-1437	(26)	#-1444	(26)			
					#-1481	(27)	#-1515	(28)	#-1527	(28)	#-1545	(29)			
					#-1556	(29)	#-1570	(29)	#-1585	(29)	#-1629	(30)			
					#-1659	(30)	#-1679	(31)	#-1691	(31)	#-1713	(32)			
					#-1721	(32)	#-174	(2)	#-1754	(32)	#-1810	(33)			
					#-1817	(33)	#-1850	(33)	#-1903	(34)	#-1911	(34)			
					#-1944	(34)	#-2000	(35)	#-2008	(35)	#-2041	(35)			
					#-2096	(36)	#-2104	(36)	#-2137	(36)	#-2192	(37)			
					#-2201	(37)	#-2234	(37)	#-2289	(38)	#-2298	(38)			
					#-2331	(38)	#-2386	(39)	#-2392	(39)	#-2425	(39)			
					#-2460	(40)	#-2498	(41)	#-2522	(41)	#-253	(3)			
					#-2567	(42)	#-2593	(42)	#-2640	(43)	#-2664	(43)			

		#-2701	(44)	#-2727	(44)	#-275	(3)	#-372	(4)
		#-433	(5)	#-474	(6)	#-502	(7)	#-539	(8)
		#-592	(9)	#-638	(10)	#-681	(11)	#-725	(12)
		#-781	(13)	#-818	(14)	#-863	(15)	#-908	(16)
		#-962	(17)						
PRINT_DTERR	00000000-XR	1790	(32)	1886	(33)	1983	(34)	2080	(35)
		2176	(36)	2273	(37)	2369	(38)	2553	(41)
		2625	(42)						
PRINT_IBCERR	00000000-XR	1169	(21)	1178	(21)	1211	(22)	1248	(23)
		1286	(24)	1292	(24)	1303	(24)	1309	(24)
		1317	(24)						
PRINT_INTERR	00000000-XR	1036	(18)	396	(4)	443	(5)	489	(6)
		516	(7)	569	(8)	613	(9)	658	(10)
		702	(11)	758	(12)	795	(13)	840	(14)
		885	(15)	939	(16)	989	(17)		
PRINT_NULL	00000000-XR	1368	(25)						
PRINT_SBE	00000000-XR	1004	(17)	1041	(18)	1051	(18)	1094	(19)
		1119	(20)	1391	(25)	1404	(25)	1412	(25)
		1457	(26)	1494	(27)	1525	(28)	1554	(29)
		1566	(29)	1579	(29)	1595	(29)	1609	(29)
		1657	(30)	1689	(31)	1699	(31)	1743	(32)
		1752	(32)	1776	(32)	1839	(33)	1848	(33)
		1872	(33)	1933	(34)	1942	(34)	1966	(34)
		2030	(35)	2039	(35)	2063	(35)	208	(2)
		2126	(36)	2135	(36)	2159	(36)	2223	(37)
		2232	(37)	2258	(37)	2320	(38)	2329	(38)
		2354	(38)	2414	(39)	2423	(39)	2450	(39)
		2486	(40)	2688	(43)	2754	(44)	309	(3)
		384	(4)	401	(4)	411	(4)	448	(5)
		458	(5)	521	(7)	531	(7)	574	(8)
		584	(8)	618	(9)	630	(9)	663	(10)
		673	(10)	707	(11)	717	(11)	763	(12)
		773	(12)	800	(13)	810	(13)	845	(14)
		855	(14)	890	(15)	900	(15)	944	(16)
		954	(16)	994	(17)				
PRINT_SILOER	00000000-XR	317	(3)	325	(3)				
PRINT_VECTERR	00000000-XR	418	(4)						
PROGRAM_ERROR	00000000-XR	#-1190	(22)	#-642	(10)				
READ	00000000-XR	#-1043	(18)	#-1766	(32)	#-1862	(33)	#-1956	(34)
		#-2053	(35)	#-2149	(36)	#-216	(2)	#-2248	(37)
		#-2345	(38)	#-2542	(41)	#-295	(3)	#-375	(4)
		#-403	(4)	#-507	(7)	#-523	(7)	#-576	(8)
		#-602	(9)	#-622	(9)	#-647	(10)	#-688	(11)
		#-709	(11)	#-745	(12)	#-892	(15)	#-946	(16)
		#-996	(17)						
READ_REV	00000000-XR	#-2614	(42)						
REG_NAME	00000000-XR	#-1382	(25)	#-1395	(25)	#-1435	(26)	#-1478	(27)
		#-1512	(28)	#-1542	(29)	#-1600	(29)	#-1625	(30)
		#-1676	(31)	#-1711	(32)	#-1745	(32)	#-1769	(32)
		#-1807	(33)	#-1841	(33)	#-1865	(33)	#-1901	(34)
		#-1935	(34)	#-1959	(34)	#-1969	(34)	#-199	(2)
		#-1998	(35)	#-2032	(35)	#-2056	(35)	#-2066	(35)
		#-2094	(36)	#-2128	(36)	#-2152	(36)	#-2162	(36)
		#-2190	(37)	#-2225	(37)	#-2251	(37)	#-2287	(38)
		#-2322	(38)	#-2347	(38)	#-2384	(39)	#-2416	(39)
		#-2442	(39)	#-2478	(40)	#-2682	(43)	#-2747	(44)
		#-348	(4)						

```
REG_NO                          00000000-XR      #-1382  (25)   #-1395  (25)   #-1435  (26)   #-1478  (27)
                                                 #-1512  (28)   #-1542  (29)   #-1600  (29)   #-1625  (30)
                                                 #-1676  (31)   #-1711  (32)   #-1745  (32)   #-1769  (32)
                                                 #-1807  (33)   #-1841  (33)   #-1865  (33)   #-1901  (34)
                                                 #-1935  (34)   #-1959  (34)   #-1969  (34)   #-199   (2)
                                                 #-1998  (35)   #-2032  (35)   #-2056  (35)   #-2066  (35)
                                                 #-2094  (36)   #-2128  (36)   #-2152  (36)   #-2162  (36)
                                                 #-2190  (37)   #-2225  (37)   #-2251  (37)   #-2287  (38)
                                                 #-2322  (38)   #-2347  (38)   #-2384  (39)   #-2416  (39)
                                                 #-2442  (39)   #-2478  (40)   #-2682  (43)   #-2747  (44)
                                                 #-348   (4)
REG_STRING                      00000000-XR      #-1382  (25)   #-1395  (25)   #-1435  (26)   #-1478  (27)
                                                 #-1512  (28)   #-1542  (29)   #-1600  (29)   #-1625  (30)
                                                 #-1676  (31)   #-1711  (32)   #-1745  (32)   #-1769  (32)
                                                 #-1807  (33)   #-1841  (33)   #-1865  (33)   #-1901  (34)
                                                 #-1935  (34)   #-1959  (34)   #-1969  (34)   #-199   (2)
                                                 #-1998  (35)   #-2032  (35)   #-2056  (35)   #-2066  (35)
                                                 #-2094  (36)   #-2128  (36)   #-2152  (36)   #-2162  (36)
                                                 #-2190  (37)   #-2225  (37)   #-2251  (37)   #-2287  (38)
                                                 #-2322  (38)   #-2347  (38)   #-2384  (39)   #-2416  (39)
                                                 #-2442  (39)   #-2478  (40)   #-2682  (43)   #-2747  (44)
                                                 #-348   (4)
RH11TB_MSK                      00000000-XR      #-1031  (18)   #-1046  (18)   #-1204  (22)   #-1379  (25)
                                                 #-202   (2)    #-303   (3)    #-378   (4)    #-391   (4)
                                                 #-405   (4)    #-453   (5)    #-511   (7)    #-525   (7)
                                                 #-564   (8)    #-579   (8)    #-609   (9)    #-624   (9)
                                                 #-652   (10)   #-668   (10)   #-697   (11)   #-711   (11)
                                                 #-753   (12)   #-768   (12)   #-791   (13)   #-805   (13)
                                                 #-835   (14)   #-850   (14)   #-880   (15)   #-894   (15)
                                                 #-934   (16)   #-948   (16)   #-983   (17)   #-999   (17)
RH11TB_PRES                     00000000-XR      #-1079  (19)   #-1143  (21)   #-172   (2)    #-347   (4)
RHSR_CHECK                      00000000-XR      #-1771  (32)   #-1867  (33)   #-1961  (34)   #-2058  (35)
                                                 #-2154  (36)   #-2253  (37)   #-2349  (38)   #-2444  (39)
                                                 #-2480  (40)   #-2683  (43)   #-2748  (44)
RHSR_MSG                        00000000-XR        1625  (30)     1676  (31)     1769  (32)     1865  (33)
                                                   1959  (34)     199   (2)      2056  (35)     2152  (36)
                                                   2251  (37)     2347  (38)     2442  (39)     2478  (40)
                                                   2682  (43)     2747  (44)     348   (4)
RH_CUR_ADR                      00000000-XR      #-1080  (19)   #-1144  (21)   #-1188  (22)   #-1221  (23)
                                                 #-1260  (24)   #-1353  (25)   #-1436  (26)   #-1476  (27)
                                                 #-1510  (28)   #-1540  (29)   #-1626  (30)   #-1677  (31)
                                                 #-1712  (32)   #-173   (2)    #-1808  (33)   #-1902  (34)
                                                 #-1999  (35)   #-2095  (36)   #-2191  (37)   #-2288  (38)
                                                 #-2385  (39)   #-2497  (41)   #-252   (3)    #-2566  (42)
                                                 #-2639  (43)   #-2700  (44)   #-274   (3)    #-349   (4)
RH_ISR                          00000000-XR        432   (5)
RH_VECS                         00000000-XR        360   (4)      421   (4)
RH_VECTOR                       00000000-XR      #-363   (4)    #-413   (4)    #-418   (4)    #-432   (5)
RS_MSK                          00000000-XR      #-1488  (27)
SEARCH                          00000000-XR      #-1516  (28)
SEP_FUNCTIONAL                  =00000002     143     (1)
SEP_REPAIR                      =00000003     143     (1)
SIM_ATTN                        00000000-XR      #-437   (5)    #-439   (5)
SIM_EBL                         00000000-XR      #-1159  (21)   #-1160  (21)   #-195   (2)    #-196   (2)
                                                 #-236   (2)    #-237   (2)    #-272   (3)    #-273   (3)
                                                 #-301   (3)    #-302   (3)    #-388   (4)    #-389   (4)
                                                 #-509   (7)    #-510   (7)    #-561   (8)    #-562   (8)
```

```
                                                #-650    (10)   #-651    (10)   #-693    (11)   #-694    (11)
SIM_OCC                         00000000-XR     #-1157   (21)   #-1201   (22)   #-1234   (23)   #-1279   (24)
                                                #-191    (2)    #-217    (2)    #-268    (3)    #-296    (3)
                                                #-376    (4)    #-552    (8)    #-646    (10)   #-689    (11)
SIM_SCLK                        00000000-XR     #-1238   (23)   #-1239   (23)   #-192    (2)    #-193    (2)
                                                #-219    (2)    #-220    (2)    #-269    (3)    #-270    (3)
                                                #-298    (3)    #-299    (3)    #-557    (8)    #-558    (8)
                                                #-648    (10)   #-649    (10)   #-690    (11)   #-691    (11)
SIZ...                          =00000001  143  (1)      143    (1)
SR                              00000000-XR     #-1032   (18)   #-1044   (18)   #-1089   (19)   #-1104   (20)
                                                #-1111   (20)   #-1114   (20)   #-1205   (22)   1288     (24)
                                                1305     (24)   #-1442   (26)   #-1459   (26)   #-1651   (30)
                                                #-1770   (32)   #-1866   (33)   #-1960   (34)   #-203    (2)
                                                #-2057   (35)   #-2153   (36)   #-2252   (37)   #-2348   (38)
                                                #-2443   (39)   #-2479   (40)   #-2683   (43)   #-2748   (44)
                                                #-304    (3)    #-379    (4)    #-392    (4)    #-406    (4)
                                                #-438    (5)    #-450    (5)    #-451    (5)    #-477    (6)
                                                #-512    (7)    #-526    (7)    #-565    (8)    #-577    (8)
                                                #-606    (9)    #-625    (9)    #-653    (10)   #-665    (10)
                                                #-666    (10)   #-698    (11)   #-712    (11)   #-754    (12)
                                                #-765    (12)   #-766    (12)   #-787    (13)   #-802    (13)
                                                #-803    (13)   #-836    (14)   #-847    (14)   #-848    (14)
                                                #-881    (15)   #-895    (15)   #-935    (16)   #-949    (16)
                                                #-984    (17)   #-997    (17)

T16_S1                          00000012-R  171  (2)
T16_S1_X                        0000020F-R  244  (2)
T16_S2                          0000021A-R  246  (3)
T16_S2_X                        00000401-R  327  (3)
T17_S1                          0000005F-R  358  (4)
T17_S10                         00000B5A-R  780  (13)
T17_S10_X                       00000C46-R  812  (13)
T17_S11                         00000C51-R  817  (14)
T17_S11_X                       00000D70-R  857  (14)
T17_S12                         00000D7B-R  859  (15)
T17_S12_X                       00000EA1-R  902  (15)
T17_S13                         00000EAC-R  904  (16)
T17_S13_X                       00000FF1-R  957  (17)
T17_S14                         00000FFC-R  961  (17)
T17_S14_X                       00001126-R  1007 (18)
T17_S15                         00001131-R  1011 (18)
T17_S15_X                       00001255-R  1053 (18)
T17_S1_X                        00000232-R  426  (4)
T17_S2                          0000023D-R  431  (5)
T17_S2_X                        00000331-R  460  (5)
T17_S3                          0000033C-R  465  (6)
T17_S3_X                        000003D6-R  496  (6)
T17_S4                          000003E1-R  501  (7)
T17_S4_X                        000004F0-R  533  (7)
T17_S5                          000004FB-R  538  (8)
T17_S5_X                        00000657-R  586  (8)
T17_S6                          00000662-R  591  (9)
T17_S6_X                        0000078A-R  632  (9)
T17_S7                          00000795-R  637  (10)
T17_S7_X                        000008B4-R  675  (10)
T17_S8                          000008BF-R  680  (11)
T17_S8_X                        000009F2-R  719  (11)
T17_S9                          000009FD-R  724  (12)
```

T17_S9_X	00000B4F-R	775	(12)						
T18_S1	0000002A-R	1078	(19)						
T18_S1_X	00000092-R	1096	(19)						
T18_S2	0000009D-R	1098	(20)						
T18_S2_X	00000121-R	1121	(20)						
T19_S1	00000026-R	1137	(21)						
T19_S1_X	0000012A-R	1180	(21)						
T19_S2	00000135-R	1182	(22)						
T19_S2_X	000001EE-R	1213	(22)						
T19_S3	000001F9-R	1215	(23)						
T19_S3_X	000002C9-R	1250	(23)						
T19_S4	000002D4-R	1252	(24)						
T19_S4_X	00000480-R	1319	(24)						
T20_S1	00000016-R	1352	(25)						
T20_S10	00000C22-R	1895	(34)						
T20_S10_X	00000E91-R	1990	(34)						
T20_S11	00000E9C-R	1992	(35)						
T20_S11_X	0000110B-R	2087	(35)						
T20_S12	00001116-R	2089	(36)						
T20_S12_X	00001385-R	2183	(36)						
T20_S13	00001390-R	2185	(37)						
T20_S13_X	000015F3-R	2280	(37)						
T20_S14	000015FE-R	2282	(38)						
T20_S14_X	00001865-R	2376	(38)						
T20_S15	00001870-R	2378	(39)						
T20_S15_X	00001A56-R	2453	(39)						
T20_S16	00001A61-R	2455	(40)						
T20_S16_X	00001B2C-R	2488	(40)						
T20_S17	00001B37-R	2490	(41)						
T20_S17_X	00001C47-R	2557	(41)						
T20_S18	00001C52-R	2559	(42)						
T20_S18_X	00001D6F-R	2630	(42)						
T20_S19	00001D7A-R	2632	(43)						
T20_S19_X	00001E78-R	2691	(43)						
T20_S1_X	000001B1-R	1414	(25)						
T20_S2	000001BC-R	1416	(26)						
T20_S20	00001E83-R	2693	(44)						
T20_S20_X	00001F89-R	2756	(44)						
T20_S2_X	0000028D-R	1463	(26)						
T20_S3	00000298-R	1465	(27)						
T20_S3_X	0000032D-R	1497	(27)						
T20_S4	00000338-R	1499	(28)						
T20_S4_X	000003D1-R	1528	(28)						
T20_S5	000003DC-R	1530	(29)						
T20_S5_X	000005A1-R	1611	(29)						
T20_S6	000005AC-R	1613	(30)						
T20_S6_X	000006A5-R	1660	(30)						
T20_S7	000006B0-R	1662	(31)						
T20_S7_X	0000076E-R	1701	(31)						
T20_S8	00000779-R	1703	(32)						
T20_S8_X	000009C2-R	1797	(32)						
T20_S9	000009CD-R	1799	(33)						
T20_S9_X	00000C17-R	1893	(33)						
TEMP1	00000000-XR			#-2590	(42)	#-2629	(42)	#-2724	(44)
TEST_016	00000010-R	150	(2)	150	(2)				
TEST_016_X	0000040F-R	328	(3)						
TEST_017	00000018-R	332	(4)	332	(4)				

TEST_017_X	00001263-R	1054	(18)									
TEST_018	00000028-R	1058	(19)	1058	(19)							
TEST_018_X	0000012F-R	1122	(20)									
TEST_019	00000024-R	1126	(21)	1126	(21)							
TEST_019_X	0000048E-R	1320	(24)									
TEST_020	00000014-R	1324	(25)	1324	(25)							
TEST_020_X	00001F97-R	2757	(44)	#-1375	(25)							
TEST_VEC	00000000-XR			#-413	(4)	#-418	(4)					
TYPE_MBE	00000000-XR			#-1357	(25)							
VALID_BIT	00000000-XR			#-1019	(18)	#-1148	(21)	#-1193	(22)	#-1225	(23)	
				#-1266	(24)	#-1633	(30)	#-1724	(32)	#-1757	(32)	
				#-178	(2)	#-1820	(33)	#-1853	(33)	#-1914	(34)	
				#-1947	(34)	#-2011	(35)	#-2044	(35)	#-2107	(36)	
				#-2140	(36)	#-2204	(37)	#-2237	(37)	#-2301	(38)	
				#-2334	(38)	#-2395	(39)	#-2428	(39)	#-2467	(40)	
				#-2529	(41)	#-2600	(42)	#-261	(3)	#-2667	(43)	
				#-2730	(44)	#-279	(3)	#-286	(3)	#-352	(4)	
				#-542	(8)	#-597	(9)	#-731	(12)	#-823	(14)	
				#-868	(15)	#-917	(16)					
VAR	00000000-XR			#-1022	(18)	#-1151	(21)	#-1171	(21)	#-1196	(22)	
				#-1228	(23)	#-1275	(24)	#-1637	(30)	#-1727	(32)	
				#-1760	(32)	#-1823	(33)	#-183	(2)	#-1856	(33)	
				#-1917	(34)	#-1950	(34)	#-2014	(35)	#-2047	(35)	
				#-2110	(36)	#-212	(2)	#-2143	(36)	#-2207	(37)	
				#-2240	(37)	#-2304	(38)	#-2337	(38)	#-2398	(39)	
				#-2431	(39)	#-2469	(40)	#-2534	(41)	#-2606	(42)	
				#-264	(3)	#-2672	(43)	#-2736	(44)	#-282	(3)	
				#-547	(8)	#-599	(9)	#-620	(9)	#-640	(10)	
				#-686	(11)	#-734	(12)	#-743	(12)	#-827	(14)	
				#-871	(15)	#-873	(15)	#-915	(16)	#-923	(16)	
				#-966	(17)	#-972	(17)					
WRITE	00000000-XR			#-1024	(18)	#-1156	(21)	#-1200	(22)	#-1233	(23)	
				#-1278	(24)	#-1646	(30)	#-1734	(32)	#-1830	(33)	
				#-190	(2)	#-1924	(34)	#-2021	(35)	#-2117	(36)	
				#-2214	(37)	#-2311	(38)	#-2405	(39)	#-267	(3)	
				#-551	(8)	#-645	(10)	#-736	(12)	#-829	(14)	
				#-874	(15)	#-919	(16)	#-968	(17)			
WRITE_CHECK	00000000-XR			#-2476	(40)	#-2680	(43)	#-928	(16)	#-976	(17)	
WRITE_CHECKREV	00000000-XR			#-2439	(39)	#-2744	(44)					
WRITE_CHK_HIGH	00000000-XR			#-981	(17)							
WRITE_CHK_LOW	00000000-XR			#-932	(16)							

Macros Cross Reference

MACRO	SIZE	DEFINITION	REFERENCES...
$D1_BSUBT	1	171 (2)	1011 (18) 1078 (19) 1098 (20) 1137 (21) 1182 (22)
			1215 (23) 1252 (24) 1352 (25) 1416 (26) 1465 (27)
			1499 (28) 1530 (29) 1613 (30) 1662 (31) 1703 (32)
			171 (2) 1799 (33) 1895 (34) 1992 (35) 2089 (36)
			2185 (37) 2282 (38) 2378 (39) 2455 (40) 246 (3)
			2490 (41) 2559 (42) 2632 (43) 2693 (44) 358 (4)
			431 (5) 465 (6) 501 (7) 538 (8) 591 (9)
			637 (10) 680 (11) 724 (12) 780 (13) 817 (14)
			859 (15) 904 (16) 961 (17)
$D1_BTEST	1	150 (2)	1058 (19) 1126 (21) 1324 (25) 150 (2) 332 (4)
$D1_ERR	1	1368 (25)	1368 (25)
$D1_ERR_S	1	208 (2)	1004 (17) 1036 (18) 1041 (18) 1051 (18) 1094 (19)
			1119 (20) 1169 (21) 1178 (21) 1211 (22) 1248 (23)
			1286 (24) 1292 (24) 1303 (24) 1309 (24) 1317 (24)
			1368 (25) 1391 (25) 1404 (25) 1412 (25) 1457 (26)
			1494 (27) 1525 (28) 1554 (29) 1566 (29) 1579 (29)
			1595 (29) 1609 (29) 1657 (30) 1689 (31) 1699 (31)
			1743 (32) 1752 (32) 1776 (32) 1790 (32) 1839 (33)
			1848 (33) 1872 (33) 1886 (33) 1933 (34) 1942 (34)
			1966 (34) 1983 (34) 2030 (35) 2039 (35) 2063 (35)
			208 (2) 2080 (35) 2126 (36) 2135 (36) 2159 (36)
			2176 (36) 2223 (37) 2232 (37) 2258 (37) 2273 (37)
			232 (2) 2320 (38) 2329 (38) 2354 (38) 2369 (38)
			241 (2) 2414 (39) 2423 (39) 2450 (39) 2486 (40)
			2553 (41) 2625 (42) 2688 (43) 2754 (44) 309 (3)
			317 (3) 325 (3) 384 (4) 396 (4) 401 (4)
			411 (4) 418 (4) 443 (5) 448 (5) 458 (5)
			489 (6) 516 (7) 521 (7) 531 (7) 569 (8)
			574 (8) 584 (8) 613 (9) 618 (9) 630 (9)
			658 (10) 663 (10) 673 (10) 702 (11) 707 (11)
			717 (11) 758 (12) 763 (12) 773 (12) 795 (13)
			800 (13) 810 (13) 840 (14) 845 (14) 855 (14)
			885 (15) 890 (15) 900 (15) 939 (16) 944 (16)
			954 (16) 989 (17) 994 (17)
$D1_ESUBT	1	244 (2)	1007 (18) 1053 (18) 1096 (19) 1121 (20) 1180 (21)
			1213 (22) 1250 (23) 1319 (24) 1414 (25) 1463 (26)
			1497 (27) 1528 (28) 1611 (29) 1660 (30) 1701 (31)
			1797 (32) 1893 (33) 1990 (34) 2087 (35) 2183 (36)
			2280 (37) 2376 (38) 244 (2) 2453 (39) 2488 (40)
			2557 (41) 2630 (42) 2691 (43) 2756 (44) 327 (3)
			426 (4) 460 (5) 496 (6) 533 (7) 586 (8)
			632 (9) 675 (10) 719 (11) 775 (12) 812 (13)
			857 (14) 902 (15) 957 (17)
$D1_ETEST	1	328 (3)	1054 (18) 1122 (20) 1320 (24) 2757 (44) 328 (3)
$D1_EXTEST	1	1375 (25)	1375 (25)
$D1_PRINT_S	1	235 (2)	1108 (20) 1374 (25) 1795 (32) 1891 (33) 1988 (34)
			2085 (35) 2181 (36) 2278 (37) 235 (2) 2374 (38)
			242 (2) 494 (6)
$D1_SBTTL	2	149 (2)	1057 (19) 1125 (21) 1323 (25) 149 (2) 331 (4)
$D2_BDATA	1	150 (2)	1058 (19) 1126 (21) 1324 (25) 150 (2) 332 (4)

$D2_BTEST	1	150	(2)	1058	(19)	1126	(21)	1324	(25)	150	(2)	332	(4)
$D2_SBTTL	1	149	(2)	1057	(19)	1125	(21)	1323	(25)	149	(2)	331	(4)
$DEF	1	143	(1)										
$DEFINI	1	141	(1)	141	(1)	146	(2)	147	(2)				
$DS_BGNMOD	1	143	(1)	143	(1)								
$DS_BGNSUB	1	171	(2)	1011	(18)	1078	(19)	1098	(20)	1137	(21)	1182	(22)
				1215	(23)	1252	(24)	1352	(25)	1416	(26)	1465	(27)
				1499	(28)	1530	(29)	1613	(30)	1662	(31)	1703	(32)
				171	(2)	1799	(33)	1895	(34)	1992	(35)	2089	(36)
				2185	(37)	2282	(38)	2378	(39)	2455	(40)	246	(3)
				2490	(41)	2559	(42)	2632	(43)	2693	(44)	358	(4)
				431	(5)	465	(6)	501	(7)	538	(8)	591	(9)
				637	(10)	680	(11)	724	(12)	780	(13)	817	(14)
				859	(15)	904	(16)	961	(17)				
$DS_BGNTEST	1	150	(2)	1058	(19)	1126	(21)	1324	(25)	150	(2)	332	(4)
$DS_BITDEF	2	140	(1)	140	(1)								
$DS_BREAK	1	328	(3)	1054	(18)	1122	(20)	1320	(24)	2757	(44)	328	(3)
$DS_CKLOOP	1	209	(2)	1005	(17)	1042	(18)	1052	(18)	1095	(19)	1120	(20)
				1170	(21)	1179	(21)	1212	(22)	1249	(23)	1287	(24)
				1293	(24)	1304	(24)	1310	(24)	1318	(24)	1392	(25)
				1405	(25)	1413	(25)	1458	(26)	1495	(27)	1526	(28)
				1555	(29)	1567	(29)	1580	(29)	1596	(29)	1610	(29)
				1658	(30)	1690	(31)	1700	(31)	1753	(32)	1777	(32)
				1791	(32)	1849	(33)	1873	(33)	1887	(33)	1943	(34)
				1967	(34)	1984	(34)	2040	(35)	2064	(35)	2081	(35)
				209	(2)	2136	(36)	2160	(36)	2177	(36)	2233	(37)
				2259	(37)	2274	(37)	2330	(38)	2355	(38)	2370	(38)
				2424	(39)	243	(2)	2451	(39)	2487	(40)	2554	(41)
				2626	(42)	2690	(43)	2755	(44)	310	(3)	318	(3)
				326	(3)	385	(4)	402	(4)	412	(4)	419	(4)
				449	(5)	459	(5)	490	(6)	522	(7)	532	(7)
				575	(8)	585	(8)	619	(9)	631	(9)	664	(10)
				674	(10)	708	(11)	718	(11)	764	(12)	774	(12)
				801	(13)	811	(13)	846	(14)	856	(14)	891	(15)
				901	(15)	945	(16)	955	(16)	995	(17)		
$DS_CLRVEC_S	1	363	(4)	363	(4)	422	(4)						
$DS_DSADEF	5	141	(1)	141	(1)								
$DS_ENDDATA	1	150	(2)	1058	(19)	1126	(21)	1324	(25)	150	(2)	332	(4)
$DS_ENDMOD	1	2760	(44)	2760	(44)								
$DS_ENDSUB	1	244	(2)	1007	(18)	1053	(18)	1096	(19)	1121	(20)	1180	(21)
				1213	(22)	1250	(23)	1319	(24)	1414	(25)	1463	(26)
				1497	(27)	1528	(28)	1611	(29)	1660	(30)	1701	(31)
				1797	(32)	1893	(33)	1990	(34)	2087	(35)	2183	(36)
				2280	(37)	2376	(38)	244	(2)	2453	(39)	2488	(40)
				2557	(41)	2630	(42)	2691	(43)	2756	(44)	327	(3)
				426	(4)	460	(5)	496	(6)	533	(7)	586	(8)
				632	(9)	675	(10)	719	(11)	775	(12)	812	(13)
				857	(14)	902	(15)	957	(17)				
$DS_ENDTEST	1	328	(3)	1054	(18)	1122	(20)	1320	(24)	2757	(44)	328	(3)
$DS_ENVDEF	2	143	(1)	143	(1)								
$DS_ERRHARD_S	1	206	(2)	1002	(17)	1034	(18)	1039	(18)	1049	(18)	1092	(19)
				1117	(20)	1166	(21)	1175	(21)	1209	(22)	1245	(23)
				1284	(24)	1290	(24)	1301	(24)	1307	(24)	1314	(24)
				1366	(25)	1390	(25)	1403	(25)	1411	(25)	1456	(26)
				1493	(27)	1524	(28)	1553	(29)	1565	(29)	1578	(29)
				1594	(29)	1608	(29)	1656	(30)	1688	(31)	1698	(31)
				1742	(32)	1751	(32)	1775	(32)	1789	(32)	1838	(33)

				1847	(33)	1871	(33)	1885	(33)	1932	(34)	1941	(34)		
				1965	(34)	1981	(34)	2029	(35)	2038	(35)	206	(2)		
				2062	(35)	2078	(35)	2125	(36)	2134	(36)	2158	(36)		
				2174	(36)	2222	(37)	2231	(37)	2257	(37)	2271	(37)		
				231	(2)	2319	(38)	2328	(38)	2353	(38)	2367	(38)		
				240	(2)	2413	(39)	2422	(39)	2448	(39)	2484	(40)		
				2551	(41)	2623	(42)	2686	(43)	2752	(44)	307	(3)		
				314	(3)	322	(3)	382	(4)	394	(4)	399	(4)		
				409	(4)	415	(4)	441	(5)	446	(5)	456	(5)		
				487	(6)	514	(7)	519	(7)	529	(7)	567	(8)		
				572	(8)	582	(8)	611	(9)	616	(9)	628	(9)		
				656	(10)	661	(10)	671	(10)	700	(11)	705	(11)		
				715	(11)	756	(12)	761	(12)	771	(12)	793	(13)		
				798	(13)	808	(13)	838	(14)	843	(14)	853	(14)		
				883	(15)	888	(15)	898	(15)	937	(16)	942	(16)		
				952	(16)	987	(17)	992	(17)						
$DS_EXIT	1	1375	(25)	1375	(25)										
$DS_HPODEF	2	146	(2)	146	(2)										
$DS_PAGE	1	144	(1)	1055	(18)	1123	(20)	1321	(24)	144	(1)	2758	(44)		
				329	(3)										
$DS_PRINTB_S	1	1795	(32)	1795	(32)	1891	(33)	1988	(34)	2085	(35)	2181	(36)		
				2278	(37)	2374	(38)								
$DS_PRINTF_S	1	1374	(25)	1374	(25)										
$DS_PRINTX_S	1	234	(2)	1108	(20)	234	(2)	242	(2)	494	(6)				
$DS_RH750_DEF	1	147	(2)	147	(2)										
$DS_SBTTL	1	149	(2)	1057	(19)	1125	(21)	1323	(25)	149	(2)	331	(4)		
$DS_SECDEF	1	88	(1)	88	(1)										
$DS_SETVEC_S	1	364	(4)	364	(4)	432	(5)								
$DS_WAITUS_S	1	222	(2)	1082	(19)	1158	(21)	1202	(22)	1235	(23)	1517	(28)		
				1647	(30)	1650	(30)	1735	(32)	1767	(32)	1831	(33)		
				1863	(33)	1925	(34)	1957	(34)	2022	(35)	2054	(35)		
				2118	(36)	2150	(36)	2215	(37)	222	(2)	2249	(37)		
				2312	(38)	2346	(38)	2406	(39)	2441	(39)	2477	(40)		
				2543	(41)	2615	(42)	2681	(43)	2746	(44)	508	(7)		
				553	(8)	563	(8)	752	(12)						
$EQU	1	143	(1)	140	(1)	143	(1)								
$EQULS1	1	143	(1)	143	(1)										
$EQULST	1	143	(1)	143	(1)										
$GBLINI	2	140	(1)	140	(1)	143	(1)								
$PUSHADR	1	208	(2)	1004	(17)	1036	(18)	1041	(18)	1051	(18)	1082	(19)		
				1094	(19)	1119	(20)	1158	(21)	1169	(21)	1178	(21)		
				1202	(22)	1211	(22)	1235	(23)	1248	(23)	1286	(24)		
				1292	(24)	1303	(24)	1309	(24)	1317	(24)	1368	(25)		
				1391	(25)	1404	(25)	1412	(25)	1457	(26)	1494	(27)		
				1517	(28)	1525	(28)	1554	(29)	1566	(29)	1579	(29)		
				1595	(29)	1609	(29)	1647	(30)	1650	(30)	1657	(30)		
				1689	(31)	1699	(31)	1735	(32)	1743	(32)	1752	(32)		
				1767	(32)	1776	(32)	1790	(32)	1831	(33)	1839	(33)		
				1848	(33)	1863	(33)	1872	(33)	1886	(33)	1925	(34)		
				1933	(34)	1942	(34)	1957	(34)	1966	(34)	1983	(34)		
				2022	(35)	2030	(35)	2039	(35)	2054	(35)	2063	(35)		
				208	(2)	2080	(35)	2118	(36)	2126	(36)	2135	(36)		
				2150	(36)	2159	(36)	2176	(36)	2215	(37)	222	(2)		
				2223	(37)	2232	(37)	2249	(37)	2258	(37)	2273	(37)		
				2312	(38)	232	(2)	2320	(38)	2329	(38)	2346	(38)		
				2354	(38)	2369	(38)	2406	(39)	241	(2)	2414	(39)		
				2423	(39)	2441	(39)	2450	(39)	2477	(40)	2486	(40)		

				2543	(41)	2553	(41)	2615	(42)	2625	(42)	2681	(43)
				2688	(43)	2746	(44)	2754	(44)	309	(3)	317	(3)
				325	(3)	384	(4)	396	(4)	401	(4)	411	(4)
				418	(4)	443	(5)	448	(5)	458	(5)	489	(6)
				508	(7)	516	(7)	521	(7)	531	(7)	553	(8)
				563	(8)	569	(8)	574	(8)	584	(8)	613	(9)
				618	(9)	630	(9)	658	(10)	663	(10)	673	(10)
				702	(11)	707	(11)	717	(11)	752	(12)	758	(12)
				763	(12)	773	(12)	795	(13)	800	(13)	810	(13)
				840	(14)	845	(14)	855	(14)	885	(15)	890	(15)
				900	(15)	939	(16)	944	(16)	954	(16)	989	(17)
				994	(17)								
$VIELD	1			143	(1)								
$VIELD1	1	143	(1)	143	(1)								
ERRPREP	2	92	(1)	1381	(25)	1394	(25)	1434	(26)	1477	(27)	1511	(28)
				1541	(29)	1599	(29)	1624	(30)	1675	(31)	1710	(32)
				1744	(32)	1768	(32)	1806	(33)	1840	(33)	1864	(33)
				1900	(34)	1934	(34)	1958	(34)	1968	(34)	197	(2)
				1997	(35)	2031	(35)	2055	(35)	2065	(35)	2093	(36)
				2127	(36)	2151	(36)	2161	(36)	2189	(37)	2224	(37)
				2250	(37)	2286	(38)	2321	(38)	2347	(38)	2383	(39)
				2415	(39)	2442	(39)	2478	(40)	2682	(43)	2747	(44)
				348	(4)								
EXPECT_INTER	1	125	(1)	1014	(18)	386	(4)	435	(5)	483	(6)	506	(7)
				548	(8)	594	(9)	644	(10)	687	(11)	742	(12)
				784	(13)	826	(14)	870	(15)	910	(16)	964	(17)
NO_INTER	1	132	(1)										

+-------------------------+
! Performance indicators !
+-------------------------+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.07	00:00:00.64
Command processing	115	00:00:00.26	00:00:00.64
Pass 1	1616	00:00:28.80	00:00:32.16
Symbol table sort	0	00:00:00.57	00:00:00.59
Pass 2	944	00:00:08.60	00:00:11.29
Symbol table output	2	00:00:00.18	00:00:00.17
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	95	00:00:04.64	00:00:05.52
Assembler run totals	2810	00:00:43.15	00:00:51.05

The working set limit was 2048 pages.
543035 bytes (1061 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 411 non-local and 227 local symbols.
2761 source lines were read in Pass 1, producing 108 object records in Pass 2.
80 pages of virtual memory were used to define 48 macros.

```
                    +----------------------------+
                    ! Macro library statistics !
                    +----------------------------+

Macro library name                    Macros defined
------------------                    --------------
SYS$COMMON: SYSLIB DIAG.MLB;953                 37
SYS$COMMON: SYSLIB STARLET.MLB;2                 6
TOTALS (all libraries)                          43

540 GETS were required to define 43 macros.

There were no errors, warnings or information messages.

MACRO/CRO/LIS=ECCAA3.SEQ ECCAA3
```