

DH11

DIAGNOSTIC CZDHMD0

AH-8494D-MC
COPYRIGHT © 76-78
FICHE 1 OF 2

JUL 1978
digital
MADE IN USA

This image shows a microfiche card with a grid of 144 frames (12 rows by 12 columns). Each frame contains a small, high-contrast image of a document page. The pages appear to be technical or diagnostic in nature, with some containing text and others containing diagrams or tables. The overall appearance is that of a dense collection of microfilm pages.

DH11

DIAGNOSTIC
CZDHMD0

AH-8494D-MC

JUL 1978

COPYRIGHT © 76-78

digital

FICHE 2 OF 2

MADE IN USA

TEST	DESCRIPTION	UNIT	VALUE	STATUS
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

.REM

PRODUCT CODE: AC-8492D-MC
PRODUCT NAME: CZDHMDO DH11 DIAGNOSTIC
DATE: JUNE 1978
AUTHOR: E. CROWLEY
MAINTAINED BY: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORTION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1976, 1978 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:
DIGITAL PDP UNIBUS MASSBUD
DEC DECUS DECTAPE

TABLE OF CONTENTS

- 1.0 GENERAL PROGRAM DESCRIPTION
 - 1.1 PROGRAM PURPOSE
 - 1.1.1 LOGIC TEST SUMMARY
 - 1.1.2 CZDHM CORE MEMORY MAP
 - 1.2 SYSTEM REQUIREMENTS
 - 1.2.1 HARDWARE REQUIREMENTS
 - 1.2.2 SOFTWARE REQUIREMENTS
 - 1.3 RELATED DOCUMENTS AND STANDARDS
 - 1.4 DIAGNOSTIC HIERARCHY PREREQUISITES
 - 1.5 FAILURE ASSUMPTIONS
- 2.0 OPERATING INSTRUCTIONS
 - 2.1 LOADING AND STARTING PROCEDURES
 - 2.1.1 LOADING PROCEDURES
 - 2.1.2 STARTING PROCEDURES
 - 2.2 SPECIAL ENVIRONMENTS
 - 2.2.1 ACT11/APT11
 - 2.2.2 "XXDP" SYSTEMS
 - 2.2.3 SWITCHLESS FEATURE
 - 2.3 PROGRAM OPTIONS
 - 2.3.1 CONSOLE SWITCH REGISTER
 - 2.3.2 CORE MEMORY LOCATIONS
 - 2.3.3 REGISTER USAGE
 - 2.4 EXECUTION TIMES
- 3.0 ERROR INFORMATION
 - 3.1 ERROR REPORTING PROCEDURES
 - 3.1.1 STANDARD SYSMAC.SML ERROR REPORTING CONVENTIONS
 - 3.1.2 ERROR MESSAGE TABLE
 - 3.1.3 DATA HEADER MNEUMONIC DEFINITIONS
 - 3.2 POWER FAIL PRINTOUT
 - 3.3 ERROR HALTS
- 4.0 PERFORMANCE AND PROGRESS REPORTS

4.1 PERFORMANCE REPORTS
4.2 PROGRESS REPORTS

- 5.0 DH11 DEVICE INFORMATION
 - 5.1 ADDRESS AND VECTOR ASSIGNMENTS
 - 5.2 REGISTER DEFINITIONS
 - 5.2.1 SYSTEM CONTROL REGISTER
 - 5.2.2 NEXT RECEIVED CHARACTER REGISTER
 - 5.2.3 LINE PARAMETER REGISTER
 - 5.2.4 CURRENT ADDRESS REGISTER
 - 5.2.5 BYTE COUNT REGISTER
 - 5.2.6 BUFFER ACTIVE REGISTER
 - 5.2.7 BREAK CONTROL REGISTER
 - 5.2.8 SILO STATUS REGISTER
 - 5.3 DH11 FUNCTIONAL LOGIC PARTITIONING
 - 5.4 DH11 MODULE ALLOCATION CHART
- 6.0 MAINTENANCE PROCEDURES
 - 6.1 INTRODUCTION
 - 6.2 PRELIMINARY CHECKS
 - 6.3 MAINTENANCE CONNECTORS
 - 6.4 COMPLETE DH11 SUB-SYSTEM CHECKOUT
 - 6.5 MAINTENANCE HEADER DESCRIPTION

1.0 GENERAL PROGRAM DESCRIPTION

1.1 PROGRAM PURPOSE

"CZDHM" IS A COMPREHENSIVE DIAGNOSTIC TEST PROGRAM DESIGNED TO AID IN THE ACCEPTANCE TESTING, INSTALLATION CHECKOUT, AND CORRECTIVE MAINTENANCE OF THE DH11 16. LINE ASYNCHRONOUS SERIAL LINE MULTIPLEXOR. IT CONSISTS OF 48. LOGICALLY SEQUENCED DIAGNOSTIC TESTS DESIGNED TO TEST AND VERIFY THAT THE DH11 IS OPERATING IN ACCORDANCE WITH ITS DESIGN SPECIFICATIONS.

THE PROGRAM IS CONFIGURABLE BY THE AUTOSIZER OR BY CONSOLE DIALOGUE TO ENABLE IT TO AUTOMATICALLY TEST AND VERIFY ALL 16. LINES ON UP TO 16. CONTIGUOUS DH11'S (WITH NON-CONTIGUOUS/CONTIGUOUS VECTOR ASSIGNMENTS). INDIVIDUAL UNITS AND INDIVIDUAL LINES WITHIN A UNIT MAY BE SELECTED OR DESELECTED TO FACILITATE FAULT ISOLATION TO A PARTICULAR DH11 OR A FUNCTIONAL AREA OF LOGIC AFFECTING A PARTICULAR LINE WITHIN A UNIT. WHENEVER AN ERROR IS DETECTED A COMPREHENSIVE ERROR REPORT IS TYPED THAT ALLOWS THE USER TO ISOLATE THE FAULT TO A FUNCTIONAL AREA OF LOGIC. EXTENSIVE DOCUMENTATION IS PROVIDED TO PERMIT THE USER TO PROCEED FROM THE ERROR REPORT TO ADDITIONAL LOGIC CHECKS TO MAKE IN ORDER TO ISOLATE THE PROBLEM TO A REPLACEABLE UNIT.

IN ORDER TO FACILITATE INSTALLATION CHECKOUT, TESTS 101, AND 105 THROUGH 107 (TEST GROUP 1) OF THE MODEM CONTROL DIAGNOSTIC, CZDHK, HAVE BEEN INCLUDED IN THIS PROGRAM. IN THIS WAY ALL THE LEVEL CONVERTERS AND CABLES CAN BE CHECKED WITH JUST ONE PROGRAM USING THE H315 TURNAROUND CONNECTOR.

1.1.1 LOGIC TEST SUMMARY

T1 CHECK SSYN RESPONSE FROM ALL DH11 REGISTERS
T2 TEST THAT "MASTER CLR" CAN CLEAR THE "SCR", "LPR", "BKR", AND "SSR" REGS
T3 TEST "SCR" REG R/W BITS CAN SET/CLR (NORMAL MODE)
T4 TEST "SCR" REG. READ ONLY BITS (NORMAL MODE)
T5 TEST "SCR" REG. BITS THAT CAN BE SET/CLR IN MAINT. MODE
T6 TEST THAT ALL R/W BITS IN "LPR" CAN BE SET/CLR
T7 TEST THAT ALL R/W BITS IN "BKR" CAN BE SET/CLR
T10 TEST THAT ALL R/W BITS IN "SSR" CAN BE SET/CLR
T11 TEST THAT CLR/SET OF BIT "N" IN "LPR" DOES NOT CLEAR ANY OTHER BITS
T12 TEST THAT CLR/SET OF BIT "N" IN "BKR" DOES NOT CLEAR ANY OTHER BITS
T13 TEST THAT CLR/SET OF BIT "N" IN "SSR" DOES NOT CLEAR ANY OTHER BITS
T14 "CAR" MEMORY ADDRESSING TEST
T15 "BCR" MEMORY ADDRESSING TEST
T16 "CAR" REGISTER TEST - ALL 1'S / ALL 0'S - ALL LINES
T17 "BCR" REGISTER TEST - ALL 1'S / ALL 0'S - ALL LINES
T20 "CAR" MEMORY PATTERNS TEST / 0'S DISTURB
T21 "BCR" MEMORY PATTERNS TEST / 0'S DISTURB
T22 "CAR" MEMORY PATTERNS TEST / 1'S DISTURB
T23 "BCR" MEMORY PATTERNS TEST / 1'S DISTURB
T24 TEST THAT "CAR" MEMORY EXT BITS SET/CLR PROPERLY
T25 TEST INTR. ENAB. BITS - INTR. CONDITION DISABLED
T26 TEST CHAR. AVAIL. I.E. WITH INTR. CONDITION ACTIVE
T27 TEST SILO OVFLW. I.E. WITH INTR. CONDITION ACTIVE
T30 TEST NON EX MEM I.E. WITH INTR. CONDITION ACTIVE
T31 TEST XMITTR DONE I.E. WITH INTR. CONDITION ACTIVE
T32 BASIC TRANSMITTER "NPR" LOGIC TEST 1
T33 TRANSMITTR NPR LOGIC TEST 2
T34 TEST THAT CHARACTER AVAILABLE CAN CAUSE RCVR INTERRUPT
T35 TEST THAT THE SILO STATUS REG COUNTS UP CORRECTLY
T36 TEST THAT SILO STATUS REGISTER DOWN COUNTS CORRECTLY
T37 TEST SILO ALARM LEVEL FOR COUNTS 0,1,2,4,8,16, AND 32
T40 TRANSMITTER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS
T41 RECEIVER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS
T42 VERIFY STORAGE OVERFLOW - NON MAINT MODE - ALL SELECTED LINES
T43 BASIC DATA TEST - ALL SELECTED LINES/ALL CHAR LENGTHS
T44 SINGLE LINE DATA TEST - ALL SELECTED LINES
T45 BASIC PARITY LOGIC TEST - ALL SELECTED LINES - ODD PARITY
T46 MULTI-LINE PARITY DATA TEST - ALL SELECTED LINES
T47 AUTO ECHO TEST 1 - ALL SELECTED LINES
T50 AUTO ECHO TEST 2 - ALL SELECTED LINES
T51 AUTO ECHO TEST 3 - ALL SELECTED LINES
T52 BREAK BIT TEST - ALL SELECTED LINES
T53 HALF DUPLEX TEST - ALL SELECTED LINES
T54 VERIFY THAT OVERRUN CAN SET PROPERLY - ALL SELECTED LINES
T55 ABBREVIATED MODEM CONTROL DIAGNOSTIC. (DZDHK T101)
T56 MODEM CONTROL DIAGNOSTIC CONTINUED (DZDHK T105)
T57 MODEM CONTROL DIAGNOSTIC CONTINUED (DZDHK T106)
T60 MODEM CONTROL DIAGNOSTIC CONTINUED (DZDHK T107)

1.1.2 CZDMM CORE MEMORY MAP

```
-----  
000000: *  
*          VECTOR AREA          *  
*  
*****  
*          STACK AREA          *  
*  
001100: *  
*          SYSMAC CONSTANTS     *  
*          AND VARIABLES        *  
*  
*****  
BEGIN:  *  
*          START-UP CODE       *  
*  
*****  
START1: *  
*          START-UP CODE       *  
*  
*****  
TST1:   *  
*          DH11 LOGIC TESTS    *  
*          TST1(8)-TST54(8)    *  
*  
*****  
$EOP:   *  
*          STANDARD SYSMAC     *  
*          UTILITY ROUTINES    *  
*  
*****  
CKRS11: *  
*          COMMON DH11 UTILITIES *  
*  
*****  
DHADR:  *  
*          DH11 PROGRAM CONSTANTS *  
*          AND VARIABLES        *  
*  
*****  
*  
*****  
* CONT. *          * CONT. *  
*****
```

```
*****  
* CONT. *  
*****  
*  
*****  
EM1: *  
* SYSMAC ERROR MESSAGE *  
* BUFFERS *  
*  
*****  
TITLE: *  
* DH11 MISCELLANEOUS *  
* MESSAGE BUFFERS *  
*  
*****  
RBUF: *  
* TRANSMIT AND RECEIVE *  
* DATA BUFFERS *  
*  
*****
```

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

A. ANY PDP11 COMPUTER SYSTEM WITH 12K OF CORE MEMORY
AND A CONSOLE TERMINAL DEVICE (VT50,LA36 ETC)

NOTE: FOR PAPER TAPE SYSTEMS USING THE PDP11 ABSOLUTE
LOADER, THE PROGRAM CAN LOAD AND RUN IN 8K OF CORE

B. A DH11 16. LINE ASYNCHRONOUS SERIAL LINE MULTIPLEXOR

C. TEST CONNECTORS AND MODULE (THE NO. OF EACH REQUIRED
IS DETERMINED BY THE PARTICULAR TEST APPLICATION.
REFER TO SECTION 6.3 FOR A COMPLETE DISCUSSION OF
THE MAINTENANCE CONNECTORS)

1. H315 TEST CONNECTOR
2. H8611 TEST CONNECTOR(FOR DH11-AD)
3. M974 TEST MODULE
4. H861 TEST CONNECTOR

1.2.2 SOFTWARE REQUIREMENTS

A. ACT11 THE PROGRAM CONTAINS THE REQUIRED ACT11/APT11
APT11 SOFTWARE HOOKS TO PROPERLY INTERFACE WITH THE
ACT/APT SYSTEMS. THE PROGRAM CONTAINS AN AUTOSIZER
AND CAN BE RUN IN QUICK VERIFY MODE USING
"CHAINS".

B. XXDP THE PROGRAM MAY BE LOADED AND RUN FROM ANY
"XXDP" MEDIUM PROVIDED THE SYSTEM HAS AT LEAST
12K OF CORE STORAGE.

1.3 RELATED DOCUMENTS AND STANDARDS

- A. DH11-0 ENGINEERING DRAWINGS
- B. DH11 MANUAL EK-DH11-MM-002
- C. PDP11 PERIPHERALS HANDBOOK
- D. PDP11 PROCESSOR HANDBOOK
- E. MD-11-DZQAC-C1 SYSMAC.SML
- F. MD-11-DZQXA "XXDP" USER'S GUIDE
- G. DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS
PROGRAMMING PRACTICES DOC NO. 175-003-009-00

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

CZDHM ASSUMES THAT THE FOLLOWING DIAGNOSTICS
HAVE BEEN RUN PRIOR TO ITS EXECUTION AND THAT NO ERRORS WERE
DETECTED:

CZDMM-D-0
CZDMMD.P11

MACY11 30A(1052) 10-MAR-78 08:05 PAGE 13
09-MAR-78 15:32

K 1

SEQ 0011
SEQ 0010

A. CPU/CORE MEMORY DIAGNOSTICS

1.5 FAILURE ASSUMPTIONS

CZDHM ASSUMES THAT THE PROGRAM CAN BE LOADED INTO CORE AND STARTED. IT ALSO ASSUMES THE CPU/MEMORY HARDWARE IS FUNCTIONING ERROR FREE.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.1 LOADING PROCEDURES

A. PAPER TAPE SYSTEMS

USE THE STANDARD PDP11 ABSOLUTE LOADER PROCEDURE FOR LOADING PAPER TAPES. AFTER LOADING THE PROGRAM MUST BE MANUALLY STARTED AS DESCRIBED IN SECTION 2.1.2.

B. "XXDP" SYSTEMS (REFER TO "XXDP" USER'S GUIDE MD-11-DZDQXA)

1. MOUNT THE APPROPRIATE MEDIUM (DECTAPE, DISK ETC) CONTAINING THE "XXDP" MONITOR AND CZDHM.
2. BOOT THE SYSTEM TO LOAD THE MONITOR
3. ONCE LOADED THE "XXDP" MONITOR PRINTS AN INTRODUCTORY MESSAGE AND RESPONDS WITH A ".".
4. TYPE: "CZDHM" FOLLOWED BY EITHER A <CR> CARRIAGE RETURN OR AN "ALTMODE" TO LOAD THE PROGRAM.

IF A <CR> WAS TYPED THE USER MUST MANUALLY START THE PROGRAM AFTER LOADING.

IF THE "ALTMODE" TERMINATOR WAS USED THE PROGRAM WILL SELF START AFTER LOADING.

NOTE: WHENEVER THE DH11 CONFIGURATION IS CHANGED THE DIAGNOSTIC SHOULD BE RELOADED.

2.1.2 STARTING PROCEDURES

A. TO AUTOMATICALLY START THE PROGRAM USING THE AUTOSIZER
(START AT LOC 000200(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION (REFER TO SECTION 6.3)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000200(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000000 (WORST CASE TESTING)

SET THE SR=000002 (TO TYPE THE DEVICE MAP)

SET THE SR=004000 (QUICK PASS)

SET THE SR=002000 (TO SKIP AN ABBREVIATED MODEM CONTROL TEST.
REFER TO SECTIONS 1.1 AND 6.3)

SET THE SR=000400 (HALT AFTER PARAMETER SET-UP)

6. SET THE HALT/ENABLE SWITCH TO ENABLE
7. DEPRESS START - THE PROGRAM WILL TEST ALL LINES ON ALL DH'S FOUND.

NOTE: THE CSR REGISTER ADDRESS OF THE MODEM CONTROL('S) IS LOADED ONLY FROM THE AUTOSIZER, HOWEVER, AFTER INITIAL LOAD, THE PROGRAM CAN BE STARTED AT 210(8) TO CHANGE SELECTION PARAMETERS AS DESCRIBED IN SECTION 2.1.2 D.

B. TO TYPE IN ALL REQUIRED PARAMETERS (START AT LOC 000200(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION (REFER TO SECTION 6.3)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000200(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000001 (FOR INPUT DIALOGUE)

AFTER INPUT DIALOGUE BEGINS BUT PRIOR TO ACTUAL TESTING:
SET THE SR=000000 (WORST CASE TESTING)

SET THE SR=004000 (QUICK PASS)

SET SR=000400 (HALT AFTER PARAMETER SET-UP)

6. SET THE HALT/ENABLE SWITCH TO ENABLE
7. DEPRESS START - THE PROGRAM TYPES THE TITLE AND THEN ASKS FOR THE NUMBER OF ADDRESSES BETWEEN VECTORS. TYPE EITHER 10(8) OR 20(8) DEPENDING UPON THE PARTICULAR CONFIGURATION TO BE TESTED:

NOTES: IF THE MODEM CONTROL VECTORS ARE INTERLEAVED WITH THE DH11 VECTORS (2040 FRONT END)

THE DISPLACEMENT IS 20(8) ADDRESSES.

FOR STANDARD DH11'S WITH CONTIGUOUS
VECTORS THE DISPLACEMENT IS 10(8) ADDRESSES.

IF <CR> ONLY WAS TYPED, THE DEFAULT
WILL BE 20(8) ADDRESSES.

8. THE PROGRAM WILL ASK FOR THE DEVICE ADDRESS.
TYPE IN THE ADDRESS (OCTAL) OF THE FIRST DH11
IN THE SYSTEM FOLLOWED BY A <CR>.

IF AN INVALID ADDRESS IS TYPED THE PROGRAM
WILL TYPE AN ERROR MESSAGE AND ASK YOU TO
TRY AGAIN.

9. THE PROGRAM WILL ASK FOR THE VECTOR ADDRESS.
TYPE IN RECEIVER VECTOR ADDRESS (OCTAL) OF
THE FIRST DH11 FOLLOWED BY A <CR>.

IF AN INVALID VECTOR ADDRESS IS TYPED THE
PROGRAM WILL TYPE AN ERROR MESSAGE AND ASK
YOU TO TRY AGAIN.

10. NEXT THE PROGRAM WILL ASK FOR THE DEVICE SELECTION
PARAMETER. TYPE IN AN OCTAL NO. ENCODED AS FOLLOWS:

BIT00=1 TEST DH11 #00
BIT01=1 TEST DH11 #01
BIT02=0 DO NOT TEST DH11 #02
..
..
BIT15=1 TEST DH11 #15

EXAMPLES:

177777<CR> TEST ALL 16. DH11'S
100000<CR> TEST ONLY DH11 #17(8)
000005<CR> TEST DH11 #00 AND 02

IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT
TO THE LAST TYPED IN DEVICE SELECT PARAMETER. IF
THIS IS THE INITIAL LOAD IT WILL DEFAULT TO
000003 (DH11 #00 AND 01)

11. NEXT THE PROGRAM WILL ASK FOR THE LINE SELECTION
PARAMETERS. TYPE AN ENCODED OCTAL NO. AS
FOLLOWS:

BIT00=1 TEST LINE #00
BIT01=1 TEST LINE #01
BIT02=0 DO NOT TEST LINE #02
..
..
BIT15=1 TEST LINE #15

EXAMPLES:

177777<CR> TEST ALL 16. LINES
100000<CR> TEST LINE 17(8) ONLY
000005<CR> TEST LINES 00 AND 02

IF A <CR> RETURN ONLY IS TYPED THE PROGRAM WILL
DEFAULT TO 16. LINES.

NOTE

IF MORE THAN ONE DH11 IS TESTED THE SAME COMBINATION
OF LINES WILL BE TESTED ON ALL DH11'S SELECTED.

12. IF SR8=1, THE PROGRAM WILL HALT AND PRINT THE
FOLLOWING MESSAGE:

'DEPRESS CONTINUE TO START TESTING'

AT THIS POINT SET UP THE DESIRED SWITCH REG-
ISTER OPTIONS (REFER TO PARA 2.3.1) AND DEPRESS
'CONTINUE' TO START THE TESTING.

THE PURPOSE OF THIS HALT IS TO ALLOW THE USER TO
DUMP THE PROGRAM AFTER SETTING UP THE CONFIGURATION
PARAMETERS FOR HIS SYSTEM.

13. PROGRAM WILL BEGIN EXECUTION. REFER TO SECTIONS
2.4, 3.0, AND 4.0 FOR ERROR AND STATUS REPORTS.

C. DEFAULT PARAMETER START ** (START AT LOC 000204(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE
PARTICULAR TEST APPLICATION (REFER TO SECTION 6.3)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000204(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000000 (WORST CASE TESTING)
6. SET THE HALT/ENABLE SWITCH TO ENABLE
7. DEPRESS START

** IF THIS IS THE INITIAL LOAD,
THE DEFAULT PARAMETERS ASSUME TWO DH11'S
WITH THE FOLLOWING ADDRESS ASSIGNMENTS

DH11 #0 DEVADR=760020, VECTOR=330, BR5
DH11 #1 DEVADR=760040, VECTOR=350, BR5

OTHERWISE, THE PROGRAM WILL DEFAULT TO
THE PARAMETERS USED IN THE PREVIOUS EXECUTION.

8. PROGRAM EXECUTION BEGINS. REFER TO SECTIONS 2.4, 3.0.

AND 4.0 FOR EXECUTION TIMES, ERROR REPORTS, AND
PROGRESS REPORTS.

D. TO CHANGE DEVICE AND LINE SELECT PARAMETERS ONLY (START AT LOC 000210(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION (REFER TO SECTION 6.3)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000210(8)
4. DEPRESS LOAD ADDRESS
5. SFT THE SR=000000 (WORST CASE TESTING)

SET THE SR=004000 (QUICK PASS)

SET THE SR=002000 (TO SKIP AN ABBREVIATED MODEM CONTROL TEST.
THIS ASSUMES THE AUTOSIZER WAS PREVIOUSLY
USED TO LOAD THE MODEM CONTROL CSR ADDRESSES.)

SET THE SR=000400 (HALT AFTER PARAMETER SET-UP)

6. SET THE HALT/ENABLE SWITCH TO ENABLE
7. DEPRESS START - THE PROGRAM TYPES THE TITLE AND THEN ASKS FOR DEVICE SELECTION PARAMETER. PROCEED AS IN (B-10) ABOVE.
9. PROGRAM WILL ASK FOR LINE SELECTION PARAMETERS. PROCEED AS IN (B-11) ABOVE.

NOTE: THE DEVICE SELECTION AND LINE SELECTION PARAMETERS APPLY TO BOTH THE DH11 AND THE MODEM CONTROL, THAT IS, IF DH #7, LINE #3 IS CHOSEN THEN MODEM CONTROL #7 LINE #3 WILL ALSO BE TESTED.

10. IF SR8=1, THE PROGRAM WILL HALT AND PRINT THE FOLLOWING MESSAGE:

"DEPRESS CONTINUE TO START TESTING"

AT THIS POINT SET UP THE DESIRED SWITCH REGISTER OPTIONS (REFER TO PARA 2.3.1) AND DEPRESS "CONTINUE" TO START THE TESTING.

THE PURPOSE OF THIS HALT IS TO ALLOW THE USER TO DUMP THE PROGRAM AFTER SETTING UP THE CONFIGURATION PARAMETERS FOR HIS SYSTEM.

11. PROGRAM WILL BEGIN EXECUTION. REFER TO SECTIONS 2.4, 3.0, AND 4.0 FOR EXECUTION TIMES ERROR AND STATUS REPORTS.

2.2 SPECIAL ENVIRONMENTS

2.2.1 ACT11/ WHEN UNDER CONTROL OF THE ACT11/APT11
APT11 SYSTEMS THE PROGRAM MAY BE LOADED IN DUMP
MODE AND CAN BE RUN AS PART OF A QUICK
VERIFY CHAIN SINCE AN AUTOSIZER IS USED.

2.2.2 XXDP THE PROGRAM MAY BE LOADED AND RUN FROM
ANY "XXDP" MEDIUM PROVIDED THERE IS AT LEAST
12K OF CORF. IT MAY BE RUN AS PART OF
AN "XXDP" CHAIN.

2.2.3 SWITCHLESS FEATURE

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH
REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS
THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER.
IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES
AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH
REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH
REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY
DOING THE FOLLOWING:

- 1) TYPE CONTROL G < G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO
LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS
OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE "'NEW='" HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE
OF THE FOLLOWING AT THE TTY:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>.
(ONLY OCTAL NUMBERS WILL BE ACCEPTED AND ONLY 6 NUMBERS
WILL BE ALLOWED)
IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH
REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U < U> IS DEPRESSED THEN THE PROGRAM WILL DO A <CR>.
RETYPE THE DESIRED NUMBER.

2.3 PROGRAM OPTIONS

2.3.1 CONSOLE SWITCH REGISTER

THE FOLLOWING TABLE ILLUSTRATES THE FUNCTIONS OF THE CONSOLE SWITCH REGISTER DURING PROGRAM START AND DURING DH TESTING:

SWITCH REGISTER	START	TESTING
15 = 1	-----	HALT ON ERROR (AFTER TYPING ERROR MESSAGE)
14 = 1	-----	LOOP CONTINUOUSLY ON CURRENT TEST.
13 = 1	-----	INHIBIT ERROR TYPINGS.
11 = 1	-----	INHIBIT SUB-TEST ITERATIONS (QUICK PASS)
10 = 1	-----	INHIBIT MODEM CONTROL ABBREVIATED TESTS.
9 = 1	-----	LOCK ON HARD ERRORS
8 = 1	HALTS AFTER CONFIGURATION TO PERMIT DUMPING PRE- CONFIGURED COPIES OF THE PROGRAM.	SEARCH FOR AND LOCK ON TEST SELECTED BY CONTENTS OF SR <07:00>
<07:00>		CONTAINS TEST NUMBER TO SEARCH FOR WHEN SR 08 = 1
1 = 1	TYPES DEVICE MAP GENERATED BY THE AUTOSIZER.	-----
0 = 1	ALLOWS THE USER TO INPUT DH PARAMETERS MANUALLY. (INHIBITS THE AUTOSIZER)	-----

2.3.2 CORE MEMORY LOCATIONS

A. DH11 CONFIGURATION TABLES AND VARIABLES

WHEN THE AUTOSIZER OPTION IS USED, THIS PROGRAM CAN RUN NON-STANDARD DH11 CONFIGURATIONS (NON-CONTIGUOUS ADDRESSES). THE USER CAN ALSO PATCH IN HIS OWN ADDRESSES TO MATCH HIS CONFIGURATION AND THEN USE THE DEFAULT START TO RUN THE UPDATED PROGRAM. THE TABLES AND LOCATIONS TO MODIFY ARE DESCRIBED BELOW:

1. DHADTB: 16. WORD DEVICE ADDRESS TABLE

THE USER CAN DEPOSIT THE ADDRESSES FOR HIS NON-STANDARD CONFIGURATION IN THIS TABLE. THE POSITION OF THE ENTRY IN THE TABLE CORRESPONDS DIRECTLY TO THE DEVICE NO. (IE DH11 #00 - WORD 00, DH11 #01 - WORD 01 ETC.)

2. DHVCTB: 16. WORD DEVICE VECTOR ADDRESS TABLE

THE USER CAN DEPOSIT THE VECTOR ADDRESSES FOR HIS NON-STANDARD CONFIGURATION IN THIS TABLE. AGAIN THE POSITION IN THE TABLE CORRESPONDS DIRECTLY TO DEVICE NUMBER.

3. BRLVL: 16. WORD BR LEVEL TABLE

THIS TABLE STORES THE BR LEVELS ASSUMED BY THE INTERRUPT SERVICE ROUTINES FOR EACH DH11. THE RCVR BR LEVEL IS STORED IN THE LOW BYTE AND THE XMITTER BR LEVEL IN THE HIGH BYTE. AGAIN THE POSITION IN THE TABLE CORRESPONDS DIRECTLY TO THE DH11 DEVICE NO.

4. DHSEL: DEVICE SELECTION PARAMETER

THIS WORD MUST BE SET UP TO CORRESPOND TO THE DEFAULT CONFIGURATION DEFINED BY THE TABLE SET-UPS. REFER TO SECTION 2.1.2.(B10) FOR A DESCRIPTION OF ITS ENCODING.

5. LINSEL: LINE SELECTION PARAMETER

THIS WORD IS PROGRAM LOADED AS A 177777(8) TO SPECIFY THAT ALL LINES (16.) ARE TO BE TESTED. IT MAY BE MODIFIED AT CONFIGURATION TIME TO SPECIFY ANY COMBINATION OF LINES TO TEST. REFER TO SECTION 2.1.3.(B11) FOR A DESCRIPTION OF ITS ENCODING.

NOTE: ONCE THE PROGRAM IS STARTED IT IS TABLE DRIVEN AND USES "DHASEL", "LINSEL" AND THE CONTENTS OF THE THREE TABLES ABOVE TO DEFINE THE CONFIGURATION TO TEST.

NCTE: IT IS RECOMMENDED THAT WHEN NON-STANDARD CONFIGURATIONS ARE

ENCOUNTERED, THE MODEM CONTROL DIAGNOSTIC, CZDHK, SHOULD BE RUN, RATHER THAN ALTERING THE MODEM CONTROL TABLES IN THIS PROGRAM.

B. SUB-TEST ITERATION COUNT

THERE IS A LOCATION TAGGED '\$MXCNT:' THAT DETERMINES HOW MANY TIMES EACH SUB-TEST IS REPEATED (SR11=0) IT IS PROGRAM LOADED TO 000010(8) BUT CAN BE CHANGED TO MODIFY THE ITERATION COUNT.

NOTE THAT MODIFYING THIS LOCATION WILL CHANGE THE PROGRAM EXECUTION TIME DEFINED IN PARA 2.4(B).

2.3.3 REGISTER USAGE

IN MOST OF THE TESTS THE GENERAL REGISTERS CONTAIN STANDARD INFORMATION AS SHOWN BELOW. ON PROGRAM HALTS THE REGISTERS CAN BE EXAMINED DIRECTLY TO DISPLAY THIS INFORMATION.

R0	TEST NUMBER IN OCTAL
R1	ADDRESS OF THE "SCR" REG (DEVICE ADDRESS)
R2	ADDRESS OF THE DH11 REGISTER BEING TESTED
R3	ACTUAL CONTENTS OF THE DH11 REG BEING TESTED
R4	WHAT THE CONTENTS OF THE DH11 REG BEING TESTED SHOULD HAVE BEEN
R5	GENERAL USE - REFER TO THE LISTING FOR ITS USE
R6	CONTENTS OF THE STACK POINTER
R7	CONTENTS OF THE PROGRAM COUNTER

2.4 EXECUTION TIMES

A. SR11 = 0 SUB-TEST ITERATIONS

WITH ONE DH11 SELECTED FOR TESTING 16. LINES ONE COMPLETE ERROR FREE PASS TAKES APPROXIMATELY 8 MINUTES.

B. SR11 = 1 INHIBIT ITERATIONS

WITH ONE DH11 SELECTED FOR TESTING 16. LINES ONE COMPLETE ERROR FREE PASS TAKES APPROXIMATELY ONE MINUTE

NOTE: THE ABOVE TIMES WERE DETERMINED WHEN THE PROGRAM WAS RUN ON A PDP-11/45 AND A PDP-11/40 CPU.

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURES

3.1.1 STANDARD SYSMAC.SML ERROR REPORTING CONVENTIONS

THE PROGRAM UTILIZES THE STANDARD PDP11 DIAGNOSTICS ERROR UTILITIES. THE TEST ROUTINE CALLS THESE UTILITIES USING AN "ERROR N" INSTRUCTION (CODED EMT) WHERE "N" IS THE NUMBER OF THE ERROR MESSAGE. THE UTILITY ROUTINE USES "N" TO ACCESS THE PROPER ERROR INFORMATION VIA THE ERROR TABLE DESCRIBED IN SECTION 3.1.2 BELOW. EACH MESSAGE RESULTS IN THREE LINES OF TYPEOUT AS FOLLOWS:

LINE 1 A BRIEF DESCRIPTION OF THE FAILING FUNCTION
LINE 2 LABELS TO IDENTIFY THE DATA TYPED ON LINE 3
LINE 3 THE ACTUAL ERROR DATA (UP TO 8 OCTAL OR DECIMAL NOS.)

EXAMPLE:

SYSTEM CONTROL REGISTER ERROR							
(PC)	(PS)	(SP)	TEST	DEVADR	REGADR	WAS	S/B
002720	000002	001074	000003	160020	160020	000000	000001

THE ERROR TABLE ITEMS SHOWN IN THE NEXT SECTION DESCRIBE ALL THE DH ERROR MESSAGES WITHIN CZDMM AND ARE INTERPRETED AS FOLLOWS:

EM ADDRESS OF THE MESSAGE FOR LINE 1
DH ADDRESS OF THE DATA HEADER MESSAGE FOR LINE 2
DT ADDRESS OF THE TABLE OF ADDRESSES THAT POINT TO THE DATA WORDS TO BE PRINTED
DF ADDRESS THAT POINTS TO THE DATA DESCRIPTOR TABLE THAT DEFINES WETHER AN ITEM IS OCTAL OR DECIMAL. IF THIS ENTRY IS "0" ALL DATA WORDS ARE IN OCTAL.

SECTION 3.1.3 DEFINES THE MEANING OF THE MNEUMONICS USED IN THE VARIOUS DATA HEADERS.

THERE ARE ONLY TWO MESSAGES IN THE MODEM CONTROL PORTION OF THIS PROGRAM:

ONE INFORMS THE USER THAT NO MODEM CONTROL'S WERE FOUND BY THE AUTOSIZER AND THE PROGRAM THEN CONTINUES TESTING THE DH11'S. THE OTHER INSTRUCTS THE USER TO RUN THE MODEM CONTROL DIAGNOSTIC, CZDHK, DUE TO AN ERROR. THE PROGRAM THEN CONTINUES.

3.1.2 ERROR MESSAGE TABLES

;ERROR TABLE ITEM FOR ERROR MESSAGE 1

EM1 ;"DH11 REGISTER REFERENCE CAUSED TIMEOUT"
DH1 ;" (PC) (PS) (SP) TEST DEVADR REGADR ""
DT1 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 2

EM2 ;"SYSTEM CONTROL REGISTER ERROR"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 3

EM3 ;"DH11 MASTER CLEAR FAILED TO CLR SPECIFIED REG"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 4

EM4 ;"LINE PARAMETER REGISTER ERROR"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 5

EM5 ;"BREAK CONTROL REGISTER ERROR"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 6

EM6 ;"SILO STATUS REGISTER ERROR"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 7

EM7 ;"CURRENT ADDRESS REGISTER ERROR - LINE #XX"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
DT2 ;\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 10

```
EM10      ;'BYTE COUNTER REGISTER ERROR - LINE #XX'  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 11

```
EM11      ;'UNEXPECTED DH11 RCVR INTERRUPT''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 12

```
EM12      ;'UNEXPECTED DH11 XMITTR INTERRUPT''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 13

```
EM13      ;'CHAR AVAILABLE FAILED TO GENERATE RCVR INTERRUPT''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 14

```
EM14      ;'TRANSMITTER NPR LOGIC ERROR - LINE # ''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 15

```
EM15      ;'XMITTR FAILED TO INTERRUPT - LINE # ''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 16

```
EM16      ;'RCVR FAILED TO INTERRUPT''  
DH2       ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''  
DT2       ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         ;PRINT ALL OCTAL
```


;ERROR TABLE ITEM FOR ERROR MESSAGE 17

```
EM17      : "TRANSMITTER TIMING ERROR - LINE # "  
DH6       : " (PC) (PS) (SP) TEST DEVADR SPEED TIMEB TIMEC"  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 20

```
EM20      : RECEIVER TIMING ERROR - LINE # "  
DH6       : " (PC) (PS) (SP) TEST DEVADR SPEED TIMEB TIMEC"  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 21

```
EM21      : "RCVR FAILED TO INTERRUPT - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 22

```
EM22      : "CHAR AVAIL FAILED TO SET ON TIME - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 23

```
EM23      : "BASIC DATA TEST ERROR - LINE # "  
DH7       : " (PC) (PS) (SP) TEST DEVADR CHRLNG WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 24

```
EM24      : "AUTO ECHO TEST ERROR - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 25

```
EM25      : "BREAK BIT TEST ERROR - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 26

```
EM26      : "HALF-DUPLEX TEST ERROR - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 27

```
EM27      : "UNEXPECTED BUS ERROR TRAP"  
DH3       : (PC) (PS) (SP) TEST TRPPC TRPPS  
DT3       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 30

```
EM30      : "UNEXPECTED RSVD INSTR TRAP"  
DH3       : (PC) (PS) (SP) TEST TRPPC TRPPS  
DT3       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 31

```
EM31      : "AUTO ECHO DATA COMPARE ERROR - LINE # "  
DH4       : (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 32

```
EM32      : "AUTO ECHO TEST TIMEOUT - LINE # "  
DH5       : " (PC) (LPRG) TEST"  
DT4       : $ERRPC,$TMP0,$TMP2  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 33

```
EM33      : "PARITY LOGIC TEST ERROR - LINE # "  
DH2       : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 34

```
EM34      : "MULTI-LINE PARITY DATA TEST ERROR - LINE # - SUBTEST # "  
DH4       : " (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "  
DT2       : $ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0         : PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 35

EM35 ;"MULTI-LINE PARITY DATA TEST TIMEOUT"
DH14 ;" (PC) (LPRG) LINACT "
DT6 ;"\$ERRPC,\$TMPO,\$TMP3"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 36

EM36 ;CHAR AVAILABLE TIMEOUT"
DH5 ;" (PC) (LPRG) TEST"
DT4 ;"\$ERRPC,\$TMPO,\$TMP2"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 37

EM37 ;"DATA COMPARE ERROR - LINE # "
DH4 ;" (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "
DT2 ;"\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 40

EM40 ;"BUFFER ACTIVE REG ERROR - LINE # "
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "
DT2 ;"\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 41

EM41 ;"RCVR FALSE INTERRUPT"
DH5 ;" (PC) (LPRG) TEST"
DT4 ;"\$ERRPC,\$TMPO,\$TMP2"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 42

EM42 ;"SILO OVERFLOW ERROR"
DH5 ;" (PC) (LPRG) TEST"
DT4 ;"\$ERRPC,\$TMPO,\$TMP2"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 43

EM43 ;"SILO OVERFLOW FAILED TO GENERATE RCVR INTERRUPT"
DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "
DT2 ;"\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4"
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 44

EM44 : 'NON EX MEMORY FAILED TO GENERATE XMITTR INTERRUPT'
DH2 : ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B
DT2 : \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 45

EM45 : 'XMIT DONE FAILED TO GENERATE XMITTR INTERRUPT''
DH2 : ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B
DT2 : \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 46

EM46 : 'CURRENT ADDRESS MEMORY PATTERNS TEST ERROR - LINE # '
DH10 : ' (PC) LINEWR PATTRN TEST DEVADR REGADR WAS S B''
DT5 : \$ERRPC,\$TMP0,\$TMP1,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 47

EM47 : 'BYTE COUNT MEMORY PATTERNS TEST ERROR - LINE # ''
DH10 : ' (PC) LINEWR PATTRN TEST DEVADR REGADR WAS S/B''
DT5 : \$ERRPC,\$TMP0,\$TMP1,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 50

EM50 : 'TEST TIMEOUT WAITING FOR XMIT DONE - LINE # '
DH2 : ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B''
DT2 : '\$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4''
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 51

EM51 : 'NPR LOGIC TEST 2 ERROR''
DH11 : ' (PC) LINACT LINCHK TEST DEVADR REGADR WAS S/B''
DT5 : \$ERRPC,\$TMP0,\$TMP1,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4'
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 52

EM52 : 'BASIC DATA COMPARE ERROR'
DH2 : ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S B
DT2 : \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 : PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 53

```
EM50      ;"TEST TIMEOUT WAITING FOR XMIT DONE - LINE # "  
DH12     ;" (PC)  SPEED  (SP)  TEST  DEVADR  REGADR  WAS  S/B"  
DT2      ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0        ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 54

```
EM22     ;"CHAR AVAIL FAILED TO SET ON TIME - LINE # "  
DH12     ;" (PC)  SPEED  (SP)  TEST  DEVADR  REGADR  WAS  S/B"  
DT2      ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
; 0      ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR EKRROR MESSAGE 55

```
EM22     ;"CHAR AVAIL FAILED TO SET ON TIME - LINE # "  
DH13     ;" (PC)  (PS)  (SP)  TEST  DEVADR  CHRLNG  SCRWAS  SCRS/B"  
DT2      ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0        ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 56

```
EM56     ;"OVERRUN BIT FAILED TO SET - LINE # "  
DH2      ; (PC)  (PS)  (SP)  TEST  DEVADR  REGADR  WAS  S/B"  
DT2      ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0        ;PRINT ALL OCTAL
```

;ERROR TABLE ITEM FOR ERROR MESSAGE 57

```
EM57     ;"STORAGE OVERFLOW BIT FAILED - LINE # "  
DH2      ; (PC)  (PS)  (SP)  TEST  DEVADR  REGADR  WAS  S/B"  
DT2      ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4  
0        ;PRINT ALL OCTAL
```

3.1.3 DATA HEADER MNEUMONIC DEFINITIONS

ALL NUMBERS PRINTED AS ERROR DATA ARE IN OCTAL

(PC) ADDRESS OF THE ERROR CALL (ERROR PC)

(PS) CONTENTS OF THE PSW AT THE TIME OF THE ERROR

(SP) CONTENTS OF THE STACK POINTER AT THE TIME OF THE ERROR

TEST TEST NUMBER

DFVADR DEVICE ADDRESS - 1ST ADDRESS IN THE SELECTED DH11

REGADR ADDRESS OF THE DH11 REGISTER BEING TESTED

WAS WHAT THE ACTUAL DATA READ WAS (DH11 REG OR CORE LOC.)

S/B WHAT THE DATA READ SHOULD HAVE BEEN

SPEED SPEED CODE IN THE "LPR" REG AT THE TIME OF THE ERROR
REFER TO SECTION 5.2.3 FOR SPEED CODE TABLES

TIMEB CONTENTS OF SOFTWARE COUNTER USED IN TIMING TESTS

TIMEC CONTENTS OF SOFTWARE COUNTER USED IN TIMING TESTS
NOTE: "TIMEB" SHOULD ALWAYS BE LESS THAN "TIMEC"

CHRLNG CHARACTER LENGTH CODE IN THE "LPR" AT THE TIME OF THE ERROR
00=5 BITS, 01=6 BITS, 02=7 BITS, 03= 8 BITS

TRPPC CONTENTS OF THE PC (R7) AT THE TIME OF A BUS ERROR
OR RSVD INSTR TRAP.

TRPPS CONTENTS OF THE PSW AT THE TIME OF A BUS ERROR
OR RSVD INSTR TRAP.

(LPRG) CONTENTS OF THE "LPR" REGISTER AT THE TIME OF THE ERROR

LINACT FLAGS USED BY MULTI-LINE TESTS TO INDICATE LINES STILL ACTIVE

WASADR CORE MEMORY ADDRESS OF THE "WAS" DATA (ACTUAL DATA READ)

SBADR CORE MEMORY ADDRESS OF THE S/B DATA (GOOD DATA)

SCRWAS CONTENTS OF THE "SCR" REGISTER

SCRS/B WHAT THE CONTENTS OF THE "SCR" REGISTER SHOULD HAVE BEEN

LINCHK LINE NO. BEING CHECKED DURING "CAR" AND "BCR" MEMORY TESTS

LINWR LINE NO. BEING WRITTEN INTO DURING "CAR" AND "BCR" MEMORY TESTS

PATRN TEST PATTERN BEING WRITTEN INTO EITHER THE "CAR" OR "BCR" MEMORIES

3.2 POWER FAIL PRINTOUT

IF A POWER FAILURE OCCURS WHILE THE PROGRAM IS RUNNING,
THE FOLLOWING PRINTOUT OCCURS:

"POWER"

AFTER THE PRINTOUT THE PROGRAM WILL BE RESTARTED AUTOMATICALLY
FROM THE BEGINNING. NO ATTEMPT IS MADE TO CONTINUE THE PROGRAM
FROM THE POINT OF THE POWER FAIL INTERRUPTION.

3.3 ERROR HALTS

A. SYSMAC ERROR SERVICE ROUTINE HALT

WHEN SR15=1 A "HALT" IS EXECUTED IN THE SYSMAC ERROR
UTILITY AFTER THE ERROR TYPEOUT. TO RESUME TESTING
FROM THE POINT OF THE "HALT" SIMPLY DEPRESS CONTINUE.

B. POWER FAIL HALT

WHEN A POWER DOWN IS DETECTED, THE PROGRAM HALTS IN
THE POWER FAIL UTILITY ROUTINE. IF FOR SOME REASON
THE AUTO-START FEATURE FAILS TO RESTART THE PROGRAM,
THE PROGRAM WILL "LOCK" ON THIS HALT IF CONTINUE IS
DEPRESSED. IN THIS CASE THE PROGRAM MUST BE RESTARTED.

C. TRAP CATCHER HALTS

ALL INACTIVE VECTORS ARE SET UP WITH THE STANDARD
PDP11 TRAP CATCHER AS DESCRIBED BELOW:

VN / VN+2
VN+2/ HALT

IF A TRAP OR INTERRUPT OCCURS TO A VECTOR THAT HAS
NOT BEEN SET UP BY THE TEST ROUTINE, A "HALT" OCCURS
IN THE VECTOR AREA. THE ADDRESS DISPLAY INDICATES
WHICH VECTOR THE PROGRAM TRAPPED TO AND THE LAST ENTRY
PUSHED ON TO THE STACK INDICATES WHERE THE PROGRAM WAS
WHEN THE TRAP OR INTERRUPT OCCURRED.

4.0 PERFORMANCE AND PROGRESS REPORTS

4.1 PERFORMANCE REPORTS

(NONE PROVIDED)

4.2 PROGRESS REPORTS

A. WHEN THE PROGRAM IS STARTED OR RESTARTED IT PRINTS THE TITLE MESSAGE:

"CZDMM-'X' DH11 DIAGNOSTIC"

WHERE 'X' IS THE REVISION LEVEL LETTER DESIGNATION.

B. WHEN THE PROGRAM BEGINS TESTING ON EACH DH11 IT TYPES THE FOLLOWING MESSAGE:

"TESTING DH11 #NN"

WHERE 'NN' IS THE DEVICE NO. IN OCTAL (00-17)

C. WHEN THE PROGRAM COMPLETES A PASS (TESTED ALL SELECTED LINES ON ALL SELECTED DH11'S) IT TYPES:

"END PASS #XXXXX"

WHERE 'X' IS THE PASS COUNT IN DECIMAL.

D. WHEN THE PROGRAM IS IN THE CONFIGURATION DIALOGUE (START AT 200, OR 210) AND SR8 AND SRO=1, THE PROGRAM WILL HALT AFTER ACCEPTING THE INPUT PARAMETERS AND TYPE THE FOLLOWING MESSAGE:

"DEPRESS CONTINUE TO START TESTING"

THE PURPOSE OF THIS HALT IS TO ALLOW THE USER TO DUMP THE UPDATED PROGRAM ON THE LOAD MEDIUM FOR NON-STANDARD CONFIGURATIONS. (SEE SECTION 2.2.2 AND 2.3.2)

5.0 DH11 DEVICE INFORMATION

5.1 ADDRESS AND VECTOR ASSIGNMENTS

THE DH11 USES FLOATING ADDRESSES AND IS LOCATED AFTER DJ11'S IN THE FLOATING ADDRESS SPACE THAT BEGINS A BECAUSE THE DH11 HAS EIGHT REGISTERS, IT MUST BE ASSIGNED AN ADDRESS THAT IS A MULTIPLE OF 20 (OCTAL). SYSTEM SHOULD HAVE CONSECUTIVE ADDRESSES.

EXAMPLE #1: A SYSTEM WITH NO DJ11'S BUT TWO DH11'S.

760 010 CANNOT USE FOR DH11'S BECAUSE NOT MULTIPLE OF 20.
760 020 FIRST DH11
760 040 SECOND DH11
760 060 DH11 GAP (INDICATES THAT THERE ARE NO MORE DH11'S).

EXAMPLE #2: A SYSTEM WITH ONE DJ11, TWO DH11'S:

760 010 FIRST DJ11
760 020 DJ11 GAP (INDICATES THAT THERE ARE NO MORE DJ11'S).
760 030 CANNOT USE FOR DH11'S BECAUSE NOT MULTIPLE OF 20.
760 040 FIRST DH11
760 060 SECOND DH11
760 100 DH11 GAP (INDICATES THAT THERE ARE NO MORE DH11'S).

THE DH11 VECTORS (2) FOLLOW THOSE OF THE DJ11 IN THE FLOATING VECTOR SPACE THAT STARTS AT ADDRESS 300. AT 300 ARE USED IN THE FOLLOWING ORDER: DC11; KL11/DL11-A, B; DP11; DM11-A; DN11; DM11-BB; DR11-A; DR11-PA611 PUNCHES; DT11; DX11; DL11-C, D, E; DH11.

THE RECEIVER VECTOR IS THE LOWER NUMBERED VECTOR. THE PRIORITY OF THE RECEIVER AND TRANSMITTER INTERRUPT SELECTABLE BY MEANS OF TWO STANDARD PDP11 PRIORITY JUMPER PLUGS. BR LEVEL 5 IS STANDARD.

5.2 REGISTER DEFINITION

THE FOLLOWING SECTION DESCRIBES THE BIT ASSIGNMENTS WITHIN EACH REGISTER: BITS MARKED UNUSED AND WRITE 0 AS ZERO. ATTEMPTING TO WRITE INTO UNUSED OR READ ONLY BITS HAS NO EFFECT ON THOSE BITS. INIT REFERS TO T GENERATED BY THE PROCESSOR (E.G. UPON EXECUTION OF A RESET INSTRUCTION). TRANSMIT AND RECEIVE ARE WITH R

5.2.1 THE SYSTEM CONTROL REGISTER - ADDRESS X00

THE SYSTEM CONTROL REGISTER IS A BYTE-ADDRESSABLE REGISTER. THE BIT ASSIGNMENT IS AS FOLLOWS:

BITS DESCRIPTION

00-03 LINE SELECTION

EACH OF THE 16 LINES SERVED BY THE DH11 HAS ITS OWN STORAGE FOR LINE PARAMETER INFORMATION, CURRENT BYTE COUNT. THESE STORAGE LOCATIONS ARE LOADED BY THE PROGRAM VIA THE LINE PARAMETER REGISTER, C REGISTER, AND BYTE COUNT REGISTER, BUT THE HARDWARE MUST FIRST BE TOLD WHICH LINE IS TO HAVE ITS CURRENT ADDRESS, OR BYTE COUNT CHANGED. THIS ROUTING IS ACCOMPLISHED BY SETTING THE LINE SELECT TO THE BINARY ADDRESS (0000-1111) OF THE DESIRED LINE. THESE BITS ARE READ/WRITE.

04, 05 MEMORY EXTENSION

THE INFORMATION STORED IN THESE BITS BECOMES BITS 16 AND 17 RESPECTIVELY OF ANY CURRENT ADDRESS PROGRAM INTO THE CURRENT ADDRESS REGISTER. THESE BITS ARE READ/WRITE BUT, WHEN READ, REPRESENT 0 OF BITS 4 AND 5 OF THE SYSTEM CONTROL REGISTER, NOT THE STATUS OF ADDRESS BITS 16 AND 17 OF THE SILO STATUS REGISTER FOR FURTHER INFORMATION. THIS ARRANGEMENT PERMITS INTERRUPT SERVICE TO SAVE THE CONTENTS OF THE SYSTEM CONTROL REGISTER ACCURATELY.

06 RECEIVER INTERRUPT ENABLE

THIS BIT, WHEN SET, ENABLES RECEIVER INTERRUPTS (BIT 7)

07 RECEIVER INTERRUPT

THIS BIT, WHEN SET, INDICATES THAT THE NUMBER OF CHARACTERS STORED IN THE SILO EXCEEDS THE "ALARM" SPECIFIED BY THE LOW BYTE OF THE SILO STATUS REGISTER. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE WHERE IT IS READ/WRITE. SETTING OF THIS BIT WILL GENERATE AN INTERRUPT REQUEST IF BIT 6 (ABOVE) IS ALSO SET.

08 CLEAR NON-EXISTENT MEMORY INTERRUPT

THIS BIT, WHEN SET, CLEARS THE NON-EXISTENT MEMORY INTERRUPT FLIP-FLOP (BIT 10) AND CLEARS ITSELF. THIS BIT IS READ/WRITE.

09 MAINTENANCE

THIS BIT, WHEN SET, PLACES THE DH11 IN MAINTENANCE MODE.

10 NON-EXISTENT MEMORY

THIS BIT IS SET WHENEVER THE NPR HARDWARE PLACES THE ADDRESSES OF A MEMORY LOCATION ON THE UNIBUS. NO SLAVE SYNC IS RECEIVED IN 20 S. THIS INDICATES THAT THE ADDRESSED LOCATION OR DEVICE DOES NOT EXIST. THIS BIT CAUSES AN INTERRUPT REQUEST IF SET WHILE TRANSMITTER AND NON-EXISTENT MEMORY INTERRUPT IS ACTIVE. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRITE.

11 MASTER CLEAR

THIS BIT, WHEN SET, GENERATES "INITIALIZE" WITHIN THE DH11, CLEARING THE SILO, THE UARTS, AND THE EXACT BITS CLEARED ARE DISCUSSED IN THE SECTION ON INITIALIZATION. READ/WRITE.

12 STORAGE INTERRUPT ENABLE

THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 14 TO GENERATE AN INTERRUPT REQUEST. THIS BIT IS

13 TRANSMITTER AND NON-EX-MEM INTERRUPT ENABLE

THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 10 OR 15 TO GENERATE AN INTERRUPT REQUEST. THIS B

14 STORAGE INTERRUPT

THIS BIT IS SET WHEN THE RECEIVER SCANNER FINDS A RECEIVER HOLDING BUFFER WITH A CHARACTER IN IT STORE THAT CHARACTER IN THE SILO, AND CANNOT DO SO BECAUSE OF A LACK OF SPACE. WHEN SET THIS BIT AN INTERRUPT REQUEST IF BIT 12 IS SET. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRITE.

15 TRANSMITTER INTERRUPT

THIS BIT IS SET WHEN THE DH11 CONCLUDES AN NPR CYCLE THAT INCREMENTED A BYTE COUNT TO ZERO, INDI CHARACTER IN A MESSAGE BUFFER WAS LOADED INTO A UART TRANSMITTER HOLDING REGISTER. THIS BIT WILL REQFST IF BIT 13 IS SET. THIS BIT IS READ/WRITE. (IT IS SET DURING AN NPR CYCLE.)

5.2.2 NEXT RECEIVED CHARACTER REGISTER ADDRESS X02

BITS DESCRIPTION

00-07 NEXT RECEIVED CHARACTER

THESE BITS CONTAIN THE NEXT RECEIVED CHARACTER, RIGHT JUSTIFIED. THE LEAST SIGNIFICANT BIT IS BIT

08-11 LINE NUMBER

THESE BITS INDICATE THE LINE NUMBER ON WHICH THE NEXT RECEIVED CHARACTER WAS RECEIVED. BIT 8 IS LEAST SIGNIFICANT BIT.

12 PARITY ERROR

THIS BIT IS SET IF THE PARITY OF THE RECEIVED CHARACTER DOES NOT AGREE WITH THAT DESIGNATED FOR

13 FRAMING ERROR

THIS BIT IS SET IF THE RECEIVER SAMPLES A LINE FOR THE FIRST STOP BIT, AND FINDS THE LINE IN A S (LOGICAL 0). THIS CONDITION USUALLY INDICATES THE RECEPTION OF A BREAK.

14 DATA OVERRUN

THIS BIT IS SET WHEN THE RECEIVED CHARACTER WAS PRECEDED BY A CHARACTER THAT WAS LOST DUE TO THE RECEIVER SCANNER TO SERVICE THE UART RECEIVER HOLDING BUFFER. REFER TO THE SECTION ON PROGRAMMIN FURTHER DETAILS ON DOUBLE-BUFFERED RECEPTION.

15 VALID DATA PRESENT

THIS BIT INDICATES THAT THE DATA PRESENTED IN BITS 14-00 IS VALID. IT PERMITS A CHARACTER HANDLI CHARACTERS FROM THE SILO UNTIL IT IS EMPTY. THIS IS DONE BY READING THIS REGISTER AND CHECKING B IS OBTAINED FOR WHICH BIT 15 IS A ZERO. THE ENTIRE NEXT RECEIVED CHARACTER REGISTER IS READ-ONLY ONLY ON A WORD BASIS.

5.2.3 LINE PARAMETER REGISTER ADDRESS X04

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE LINE SELECTION BITS OF THE SYSTEM CONTROL REGISTER HAVE BEEN SET TO THE LINE TO WHICH THESE PARAMETERS APPLY. THIS REGISTER IS WRITE ONLY.

BITS DESCRIPTION

00-01 CHARACTER LENGTH

THESE BITS SHOULD BE SET AS SHOWN TO RECEIVE AND TRANSMIT CHARACTERS OF THE LENGTH (EXCLUDING PARITY)

BIT 01 00

0	0	5 BIT
0	1	6 BIT
1	0	7 BIT
1	1	8 BIT

02 TWO STOP BITS

THIS BIT, WHEN SET, CONDITIONS A LINE TRANSMITTING WITH 6, 7, OR 8-BIT CODE TO TRANSMIT CHARACTER MARKS. IF THE LINE IS TRANSMITTING 5-BIT CODE, ASSERTION OF THIS BIT CAUSES THE CHARACTERS TO BE 1.5 STOP MARKS. IF THIS BIT IS NOT ASSERTED, 1 STOP MARK IS SENT.

03 NOT USED

04 PARITY ENABLED

IF THIS BIT IS SET, CHARACTERS TRANSMITTED ON THIS LINE WILL HAVE AN APPROPRIATE PARITY BIT AFFI
RECEIVED ON THIS LINE WILL HAVE THEIR PARITY CHECKED.

05 ODD PARITY

IF THIS BIT AND BIT 4 ARE SET, CHARACTERS OF ODD PARITY WILL BE GENERATED ON THIS LINE AND INCOM
WILL BE EXPECTED TO HAVE ODD PARITY. IF THIS BIT IS NOT SET, BUT BIT 4 IS SET, CHARACTERS OF EVE
GENERATED ON THIS LINE AND INCOMING CHARACTERS WILL BE EXPECTED TO HAVE EVEN PARITY. IF BIT 4 IS
OF THIS BIT IS IMMATERIAL.

06-09 RECEIVER SPEED

THE STATE OF THESE BITS DETERMINES THE OPERATING SPEED FOR THIS LINE'S RECEIVER. THE SPEED TABLE
BELOW IS APPLICABLE.

10-13 TRANSMITTER SPEED

THE STATE OF THESE BITS DETERMINES THE OPERATING SPEED FOR THIS LINE'S TRANSMITTER. THE SPEED
TABLE ON THE NEXT PAGE IS APPLICABLE.

SPEED TABLE FOR RECEIVER AND TRANSMITTER SPEEDS:

	BIT				
TRANSMITTER	13	12	11	10	
RECEIVER	9	8	7	6	
	--	--	--	--	
	0	0	0	0	ZERO BAUD
	0	0	0	1	50 BAUDS
	0	0	1	0	75 BAUDS
	0	0	1	1	110 BAUDS
	0	1	0	0	134.5 BAUDS
	0	1	0	1	150 BAUDS
	0	1	1	0	200 BAUDS
	0	1	1	1	300 BAUDS
	1	0	0	0	600 BAUDS
	1	0	0	1	1200 BAUDS
	1	0	1	0	1800 BAUDS
	1	0	1	1	2400 BAUDS
	1	1	0	0	4800 BAUDS
	1	1	0	1	9600 BAUDS
	1	1	1	0	EXTERNAL INPUT A
	1	1	1	1	EXTERNAL INPUT B

14 HALF DUPLEX/FULL DUPLEX

IF THIS BIT IS SET, THIS LINE WILL OPERATE IN HALF-DUPLEX MODE. IF NOT SET, THIS LINE WILL OPERATE IN FULL-DUPLEX MODE.

IN THIS APPLICATION HALF-DUPLEX MEANS THAT THE DH11 RECEIVER IS BLINDED DURING TRANSMISSION OF A

15 AUTO-ECHO ENABLE

WHEN THIS BIT IS SET, CHARACTERS RECEIVED ON THIS LINE WILL BE HARDWARE ECHOED. SEE THE DISCUSSION FOR FURTHER DETAILS.

5.2.4 CURRENT ADDRESS REGISTER ADDRESS X06

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE SYSTEM CONTROL REGISTER (SCR) HAS HAD THE APPROPRIATE BITS DESIRED LINE NUMBER. WHEN THIS REGISTER IS LOADED, ADDRESS BITS 00-15 ARE TRANSFERRED INTO SEMICONDUCTOR MEMORIES IN THE DH11 FROM BITS 00-15 OF THIS REGISTER. ADDRESS BITS 16-17 ARE TRANSFERRED INTO SEMICONDUCTOR MEMORIES IN THE DH11 FROM BITS 4-5 OF THE SYSTEM CONTROL REGISTER.

INTERRUPTS MUST BE INHIBITED OR THE SCR SAVED BETWEEN THE SETTING OF THE SCR BITS 0-3 AND THE READ OF WR ADDRESS REGISTER.

WHEN THIS REGISTER IS READ, IT WILL INDICATE THE CURRENT ADDRESS OF THE LINE SELECTED BY THE SYSTEM CONTROL BITS 16 AND 17 WILL APPEAR IN THE SILO STATUS REGISTER, BITS 6 AND 7.

5.2.5 BYTE COUNT REGISTER ADDRESS X10

IN THE SAME FASHION AS THE LINE PARAMETER AND CURRENT ADDRESS REGISTERS, THIS REGISTER SHOULD NOT BE LOADED FIRST SELECTING A LINE NUMBER BY MEANS OF THE LOWER-ORDER FOUR BITS OF THE SYSTEM CONTROL REGISTER. THIS REGISTER IS LOADED WITH THE TWO'S COMPLEMENT OF THE NUMBER OF CHARACTERS (BYTES) TO BE TRANSMITTED ON THAT LINE. THE REGISTER IS READ/WRITE.

INTERRUPTS MUST BE INHIBITED OR THE SCR SAVED BETWEEN THE SETTING OF THE SCR BITS 0-3 AND THE READ OR WR BYTE COUNT REGISTER

5.2.6 BUFFER ACTIVE REGISTER (BAR) ADDRESS X12

THIS REGISTER CONTAINS ONE BIT FOR EACH LINE. THE BITS ARE INDIVIDUALLY SET USING BIS INSTRUCTIONS. SETTING A BIT INDICATES TRANSMISSION ON THE ASSOCIATED LINE. THE BIT IS CLEARED BY THE HARDWARE WHEN THE LAST CHARACTER TO BE TRANSMITTED IS LOADED INTO THE TRANSMITTER DATA HOLDING REGISTER OF THE UART FOR THAT LINE. IT SHOULD BE NOTED THAT THE CLEARING OF A BAR DOES INDICATE THAT A MESSAGE MAY BE SENT, IT DOES NOT INDICATE THAT THE LAST CHARACTERS FROM THE PRECEDING MESSAGE HAVE BEEN COMPLETELY SENT. SPECIFICALLY, TWO MORE CHARACTERS WILL BE SENT AFTER THE BAR BIT CLEARS. THESE ARE THE LAST TWO CHARACTERS OF THE MESSAGE; ONE OF THEM WAS JUST STARTING WHEN THE BAR BIT CLEARED AND ONE WAS THAT FINAL CHARACTER THAT WAS LOADED INTO THE HOLDING REGISTER, THUS CLEARING THE BAR BIT. THIS IS A NORMAL CONSEQUENCE OF DOUBLE-BUFFERED TRANSMISSION AND IS MENTIONED HERE FOR THE BENEFIT OF PROGRAMMERS WHO WANT TO WRITE PROGRAMS THAT CONTROL SUCH MODEM LEADS AS REQUEST TO SEND. REQUEST TO SEND (RTS) SHOULD NOT BE DROPPED UNTIL AT LEAST TWO CHARACTER TIMES AFTER THE BAR BIT FOR A GIVEN LINE CLEARS.

THIS TIMING MAY BE EFFECTED BY SENDING TWO EXTRA (NULL) CHARACTERS IN A MESSAGE AND DROPPING RTS WHEN BAR BIT CLEARS. CLEARING A BAR BIT SHOULD NOT BE USED TO ABORT TRANSMISSION ON A LINE. RATHER, THE BYTE COUNT FOR THAT LINE SHOULD BE SET TO ZERO. THE BUFFER ACTIVE REGISTER BITS ARE READ/WRITE.

5.2.7 BREAK CONTROL REGISTER ADDRESS X14

THIS REGISTER CONTAINS ONE BIT FOR EACH LINE. SETTING A BIT IN THIS REGISTER WILL IMMEDIATELY GENERATE A BREAK SIGNAL ON THE LINE CORRESPONDING TO THAT BIT NUMBER. CLEARING THE BIT WILL TERMINATE THE BREAK CONDITION. THE BREAK SIGNAL MAY BE TIMED BY SENDING CHARACTERS DURING THE BREAK INTERVAL, SINCE THESE CHARACTERS WILL NEVER ACTUALLY BE TRANSMITTED. FURTHER COMMENTS CONCERNING THE TRANSMISSION OF BREAK SIGNALS MAY BE FOUND IN THE BREAK SIGNALS SECTION.

5.2.8 SILO STATUS REGISTER ADDRESS X16

THIS REGISTER IS ACTUALLY TWO BYTE-SIZED REGISTERS. THE BIT ASSIGNMENTS ARE:

BIT DESCRIPTION

00-05 SILO ALARM LEVEL

THE PROGRAM MAY LOAD AN INTEGRAL POWER OF 2 BETWEEN 0 AND 63 INTO THIS LOCATION (E.G., 0, 1, 2, WHEN THE NUMBER OF CHARACTERS STORED IN THE SILO EXCEEDS THAT NUMBER, AN INTERRUPT REQUEST (SYST REGISTER BIT 7) IS GENERATED, IF SYSTEM CONTROL REGISTER BIT 6 IS SET. THESE BITS ARE READ/WRITE

06-07 READ EXTENDED MEMORY

THESE BITS ARE READ ONLY AND CONTAIN THE A16 AND A17 BITS OF THE CURRENT LINE ADDRESS WHICH THE SELECTION BITS OF THE SYSTEM CONTROL REGISTER ARE POINTING.

08-13 SILO FILL LEVEL

THESE BITS ARE AN UP-DOWN COUNTER THAT INDICATES THE ACTUAL NUMBER OF CHARACTERS IN THE SILO. IT BE NOTED THAT THERE ARE SIX BITS, HENCE NUMBERS BETWEEN 0 AND 63 CAN BE REPRESENTED. A FULL SILO ENTRIES AND THE FILL LEVEL APPEARS AS 00000, BUT ONE MAY EASILY TELL THE DIFFERENCE BETWEEN AN E SILO (00000) AND A FULL SILO (00000) BY CHECKING THE STORAGE OVERFLOW BIT (BIT 14 OF SYSTEM CONT THESE BITS ARE READ ONLY.

5.3 DH11 FUNCTIONAL LOGIC PARTITIONING

THIS SECTION LISTS ALL OF THE PRINTS FOR ALL OF THE
MODULES IN THE DH11 SUBSYSTEM. IT BRIEFLY SUMMARIZES THE
FUNCTIONAL LOGIC DESCRIBED ON EACH PRINT. THIS INFORM-
ATION MAY PROVE USEFUL FOR A MODULE OR CHIP REPLACEMENT GUIDE
WHEN THE FUNCTIONAL AREA OF LOGIC THAT IS FAULTY IS KNOWN TO
BE INTERMITTENTLY FAILING.

M7277 CURRENT ADDR REG MEMORY AND ADDR SELECT

- SH3: CONTROL STROBE MUX FOR THE "LPR" REGISTER
TRANSMITTER DATA MUX WITH AUTO-ECHO CONTROL
LOGIC SELECTION
MSYN / SSYN TIMING CHAIN
DH11 MASTER CLEAR LOGIC
- SH4: UNIBUS ADDRESS SELECTION LOGIC WITH JUMPERS
TRANSMITTER SCAN COUNTER WITH XMITTER STATUS
MULTIPLEXOR (BAR N * TBMT N)
- SH5: BYTE COUNT AND CURRENT ADDRESS MEMORY WRITE
TIMING LOGIC
CURRENT ADDRESS MEMORY LOGIC BITS <17:08>
UNIBUS ADDRESS DRIVERS BITS <17:08>
- SH6: BYTE COUNT AND CURRENT ADDRESS MEMORY ADDRESS
SELECT MULTIPLEXOR
CURRENT ADDRESS MEMORY LOGIC FOR BITS <07:00>
UNIBUS ADDRESS DRIVERS FOR BITS <07:00>
TRANSMITTER EVEN/ODD BYTE DATA MULTIPLEXOR

M7279 FIFO BUFFER

- SH1: INPUT DATA MULTIPLEXOR FOR SILO MEMORY
- SH2: SILO MEMORY CHIPS (FOUR 64 X 4 CHIPS)
SILO MEMORY READ/WRITE TIMING LOGIC
"SSR" REGISTER BITS <13:08>
SILO ALARM LEVEL COMPARATOR

CZDMM-D-0
CZDMM.D.P11

MACY11 30A(1052) 10-MAR-78 08:05 PAGE 43
09-MAR-78 15:32

B 4

SEQ 0041
SEQ 0040

RECEIVED "DATA READY" STATUS FLAG

M7288 LINE PARAMETER CONTROL

- SH3: CLOCK TIMING SIGNAL BUFFERS
- SH4: TRANSMITTER CLOCK SELECTION MULTIPLEXORS LINES<03:00>
- SH5: RECEIVER CLOCK SELECTION MULTIPLEXORS LINES <03:00>
AUTO ECHO AND HALF DUPLEX CONTROL LINES<03:00>
- SH6: TRANSMITTER CLOCK SELECTION MULTIPLEXORS LINES<07:04>
- SH7: RECEIVER CLOCK SELECTION MULTIPLEXORS LINES <07:04>
AUTO ECHO AND HALF DUPLEX CONTROL LINES <07:04>
- SH8: TRANSMITTER CLOCK SELECTION MULTIPLEXORS LINES<11:08>
- SH9: RECEIVER CLOCK SELECTION MULTIPLEXORS LINES <11:08>
AUTO ECHO AND HALF DUPLEX CONTROL LINES <11:08>
- SH10: TRANSMITTER CLOCK SELECTION MULTIPLEXORS LINES<15:12>
- SH11: RECEIVER CLOCK SELECTION MULTIPLEXORS LINES <15:12>
AUTO ECHO AND HALF DUPLEX CONTROL LINES <15:12>

M7280 #1 MULTIPLE UART CARD FOR LINES <0-7>

- SH2: UART CHIPS BIT<1:0>
RECEIVER SCAN MULTIPLEXORS (CLR R DONE, STB RD, MASTER DA, AND MASTER OR)
- SH3: UART CHIPS BIT<3:2>
- SH4: UART CHIPS BIT<5:4>
RECEIVER SCAN MULTIPLEEXORS (MASTER FE AND MASTER PE)
- SH5: UART CHIPS BIT<7:6>
TRANSMITTER SCAN MULTIPLEXOR
-12 VOLT DC REGULATOR

M7280 #2 MULTIPLE UART CARD FOR LINES <8-15>

- SH2: UART CHIPS BIT<9:8>
RECEIVER SCAN MULTIPLEXORS (CIR R DONE, STB RD, MASTER DA, AND MASTER OR)
- SH3: UART CHIPS BIT<11:10>
- SH4: UART CHIPS BIT<13:12>
RECEIVER SCAN MULTIPLEEXORS (MASTER FE AND MASTER PE)
- SH5: UART CHIPS BIT<15:14>
TRANSMITTER SCAN MULTIPLEXOR
-12 VOLT DC REGULATOR

M7289 SYSTEM CONTROL AND RCVR SCAN

- SH3: TRANSMITTER AND RECEIVER SCAN LOGIC
- SH4: TRANSMITTER AND RECEIVER SCAN TIMING
- SH5: HALF-DUPLEX CONTROL LOGIC
UART DATA MULTIPLEXORS
- SH6: SYSTEM CONTROL REGISTER

M4540 DC11 - DH11 CLOCK

- SH1: CRYSTAL OSCILLATOR AND FREQUENCY DIVIDERS

M796 UNIBUS MASTER CONTROL

- SH1: NPR CONTROL LOGIC

M7281 #1 INTERRUPT CONTROL

- 'A' SECTION: RECEIVER INTERRUPT CONTROL LOGIC
- 'B' SECTION: TRANSMITTER INTERRUPT CONTROL LOGIC

M7281 #2 NPR CONTROL

- 'A' SECTION: USED TO GAIN CONTROL OF THE BUS FOR NPR XFERS
- 'B' SECTION: (NOT USED)

M7278 REGISTER AND BYTE CONTROL

- SH3: CLEAR 'BAR' REG MULTIPLEXOR
BYTE COUNT MEMORY AND CONTROL - BITS<15:08>
UNIBUS RECEIVERS BIT<15:08>
- SH4: BYTE COUNT MEMORY AND CONTROL LOGIC BIT<07:00>
UNIBUS DATA RECEIVERS BIT<07:00>
- SH5 - SH8: 'BAR', 'BCR', 'LPR', AND 'SSR' REGISTERS PLUS THE
DATA OUTPUT MUX AND UNIBUS DRIVERS FOR DATA LINES.
 - SH5: BIT<15:12> SH7: BIT<07:04>
 - SH6: BIT<11:08> SH8: BIT<03:00>

5.4 DH11 MODULE ALLOCATION CHART

VIEW FROM WIRING SIDE

		SLOT								
		1	2	3	4	5	6	7	8	9
		M920	M7821	M7277	M7287	M7289	M7821	M7360	M7288	M920
		CABLE								CABLE
ROW A	UNIBUS CONNECTOR (NOTE #3)		NPR CNTL	REG 8 BYTE CNT	CURRENT ADDRS 8 ADDRS	SYSTEM CNTL 8 RCV SCAN	INTR CNTL	PRIORITY SELECTOR (NOTE #9)	LINE PARAMETER CNTL	UNIBUS CONNECTOR (NOTES #1) 8 #2)
			M796				M405	M971		
								CABLE		
B			UNIBUS MASTER CNTL				EXTERNAL B CLOCK (NOTE #5)	DATA CABLE (NOTES #6 8 #9)		
		M7247	M7247				M7280	M7280		M7279
C	* CONTROL MUX LINES 8-15 (NOTE #7)		* CONTROL MUX LINES 0-7 (NOTE #8)				MULTIPLE UART LINES 0-7	MULTIPLE UART LINES 8-15		FIFO BUFFER
D										
		M105	M7246							M405
E	* ADDRESS SELECTOR (NOTE #7)		* CONTROL SCAN (NOTES #4) 8-#8							EXTERNAL A CLOCK (NOTE #5)
		M7821								M4540
F	* INTR CNTL (NOTE #7)									DH11 DC11 CLOCK

FIGURE 2-4 DH11 MODULE UTILIZATION DIAGRAM
PAGE 2

NOTES:

1. IF END OF BUS, REPLACE M920 WITH M930.
2. IF LAST UNIT IN BASIC BOX, REPLACE M920 WITH BC11A CABLE WHEN EXPANDING TO PERIPHERAL BOX.
3. IF FIRST UNIT IN EXPANDER BOX, REPLACE M920 WITH BC11A CABLE.
4. E02 MUST BE G727 GRANT CONTINUITY IF MODEM CONTROL MODULE SET IS NOT INSTALLED. * DENOTES MODEM CONTROL OPTION, WITH DH11-AA OR AC.
5. MODULE SLOTS PROVIDE FOR ADDITIONAL CLOCK RATES.
6. FOR DIAGNOSTIC CHECKOUT OF DH11-AA, AB, OR AC, REPLACES M971 WITH M974.
7. THIS SLOT CONTAINS MODEM CONTROL MODULE M7807 WITH DH11-AD.
8. THIS SLOT CONTAINS MODEM CONTROL MODULE M7808 WITH DH11-AD.
9. THIS SLOT CONTAINS EIA CONVERTER AND PRIORITY MODULE M5906 FOR DH11-AD OR AE.

6.0 MAINTENANCE PROCEDURES

6.1 INTRODUCTION

THIS SECTION DESCRIBES HOW TO USE CZDMM AS A TROUBLESHOOTING TOOL. IT OUTLINES SOME PRELIMINARY CHECKS TO MAKE BEFORE STARTING DETAILED DEBUG PROCEDURES. IT ATTEMPTS TO PROVIDE THE USER WITH SUGGESTIONS FOR PROCEEDING FROM THE ERROR PRINTOUT TO DETAILED ISOLATION OF THE FAULT.

6.2 PRELIMINARY CHECKS

A. VISUAL INSPECTION

PERFORM A VISUAL INSPECTION OF THE DH11 SUBSYSTEM TO INSURE THAT:

- 1) ALL MODULES ARE INSTALLED IN THEIR PROPER SLOTS (REFER TO PARA 5.3) AND ARE PROPERLY SEATED.
- 2) THE CABLING IS CORRECT AND ALL CABLE CONNECTORS ARE FIRMLY SEATED.
- 3) THE REQUIRED MAINTENANCE CONNECTORS ARE PROPERLY INSTALLED. (REFER TO SECTION 6.3)

B. POWER CHECKS

USE A SCOPE TO CHECK THE FOLLOWING POWER SUPPLY AND CONTROL SIGNALS ON THE DH11 BACKPLANE:

+15 VDC	GRAY WIRE
-15 VDC	BLUE WIRE
+5 VDC	RED WIRE
AC LO	YELLOW WIRE
DC LO	PURPLE WIRE

NOTES: USE THE BLACK WIRE FOR GROUND REFERENCE

"AC LO" AND "DC LO" SHOULD BOTH BE "HIGH" - "LOW" GOING GLITCHES ON THESE LINES CAN CAUSE UNUSUAL AND SUBTLE SYMPTOMS.

6.3 MAINTENANCE CONNECTORS

MOST OF THE TESTS IN MD-DZDMM USE HARDWARE DIAGNOSTIC AIDS TO TURN THE DATA AROUND. THESE AIDS REQUIRE THAT THE USER INSTALL SPECIFIC TURNAROUND CONNECTORS OR MODULES BEFORE RUNNING THE PROGRAM. DEPENDENT UPON THE SPECIFIC DH11 CONFIGURATION AND THE TYPE OF TESTING DESIRED, CERTAIN MAINTENANCE AIDS MUST BE INSTALLED AS OUTLINED BELOW:

A. DH11-AA, AB, OR AC CONFIGURATIONS

- 1) TESTING LOGIC FOR ALL LINES WITHOUT DATA CABLES OR LEVEL CONVERTERS.
 - A. REMOVE THE DATA CABLE FROM SLOT B7 IN EACH DH11 TO BE TESTED.
 - B. INSTALL AN M974 MAINT JUMPER MODULE INTO SLOT B7 OF EACH DH11 TO BE TESTED.
- 2) TESTING ALL 16. LINES INCLUDING DATA CABLES WHICH CONNECT TO DISTRIBUTION PANEL. DOES NOT TEST LEVEL CONVERTER CIRCUITS LOCATED IN DISTRIBUTION PANEL.
 - A. INSTALL THE M974 MAINT JUMPER MODULE INTO SLOT B3 OF THE MULTIPLEXOR DISTRIBUTION PANEL FOR EACH DH11 TO BE TESTED. ALL LEVEL CONVERTERS IN THE DISTRIBUTION PANEL MUST BE REMOVED FOR THIS TEST.
- 3) TESTING ONE OR MORE SINGLE LINES INCLUDING EIA LEVEL CONVERTERS AND DEVICE CABLES WHICH ARE NOT TESTED IN 1 AND 2 ABOVE.
 - A. INSTALL AN H315 TEST CONNECTOR AT THE END OF THE DEVICE CABLE FOR EACH LINE TO BE TESTED.

B. DH11-AD CONFIGURATION

1. TESTING ALL 16. LINES WITHOUT DATA CABLES
 - A. DISCONNECT THE DATA CABLES (2) FROM THE TWO CONNECTORS ON THE M5906 MODULE (SLOT AB7 OF THE DH11 BACKPLANE).
 - B. INSTALL TWO H8611 TEST CONNECTORS ON THE M5906 IN PLACE OF THE CABLES.
 - C. IF MODEM CONTROL SECTION IS TO BE TESTED, DISCONNECT 4 BC08R CABLES FROM DISTRIBUTION PANEL AND CONNECT CABLES TO H861 TURNAROUND CONNECTOR.
2. TESTING ONE OR MORE SINGLE LINES INCLUDING DATA CABLES
 - A. DISCONNECT THE DEVICE CABLE FROM THE DH11-AD DISTRIBUTION PANEL FOR EACH LINE TO BE TESTED.
 - B. INSTALL AN H315 TEST CONNECTOR IN ITS PLACE ON THE DH11-AD DISTRIBUTION PANEL.

NOTE: TO TEST THE DEVICE CABLE AS WELL, INSTALL THE H315 TEST CONNECTOR AT THE END OF THE

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052) 10-MAR-78 08:05 PAGE 51
09-MAR-78 15:32

J 4

SEQ 0049
SEQ 0048

DEVICE CABLE AND LEAVE THE DEVICE CABLE
CONNECTED TO THE DISTRIBUTION PANEL.

6.4 COMPLETE DH11 SUBSYSTEM CHECKOUT

COMPLETE DH11 SUB-SYSTEM VERIFICATION INVOLVES RUNNING THE FOLLOWING PROGRAMS IN THE SEQUENCE SUGGESTED:

A. CZDHM DH11 DIAGNOSTIC

LOAD AND RUN THE BASIC DIAGNOSTIC CONFIGURED TO TEST ALL 16. LINES ON ALL DH11'S INSTALLED IN THE SYSTEM AND ALLOW IT TO COMPLETE AT LEAST TWO COMPLETE PASSES. (THE FIRST PASS IS A QUICK VERIFY WITHOUT SUB-TEST ITERATIONS AND THE SECOND PASS INCLUDES SUB-TEST ITERATIONS). IF ANY ERRORS ARE REPORTED, STOP HERE, AND REFER TO PARA 6.5 FOR SUBSEQUENT PROCEDURES.

B. CZDHN DATA RELIABILITY TEST

LOAD AND RUN THE DATA RELIABILITY SUB-PROGRAM OF CZDHN CONFIGURED TO TEST ALL 16. LINES ON ALL DH11'S INSTALLED IN THE SYSTEM AND ALLOW IT TO COMPLETE AT LEAST ONE PASS WITH SR07=0 (QUICK TEST MODE). IN THIS MODE ONE PASS TAKES APPROXIMATELY 5 MINUTES. IF ANY ERRORS ARE REPORTED, REFER TO THE DOCUMENTATION FOR CZDHN FOR SUBSEQUENT PROCEDURES.

FOR MORE COMPLETE DATA RELIABILITY TESTING, THE PROGRAM MAY BE RUN WITH SR07=1 (COMPLETE TEST) BUT THIS WILL REQUIRE A RUN TIME OF APPROX 15. MINUTES FOR EACH SELECTED LINE, SO IN MOST CASES IT IS ONLY USED FOR OVER-NIGHT RUNS.

C. SYSTEMS EXERCISER 'DHAX' EXERCISER MODULE

WHERE 'X' DESIGNATES THE REVISION LEVEL IN USE.

ASSUMING THAT BOTH THE DIAGNOSTIC AND THE DATA RELIABILITY INDICATE ERROR FREE PERFORMANCE, THE FINAL STEP IS TO RUN THE SYSTEM'S EXERCISER PROGRAM THAT INCLUDES A DH11 EXERCISER MODULE. THIS IS NECESSARY TO DETECT CERTAIN CLASSES OF BUS PROBLEMS THAT ONLY MANIFEST THEMSELVES WHEN THE DH11 IS RUN CONCURRENTLY WITH ALL THE OTHER DEVICES IN THE SYSTEM

IF ALL TESTS UP TO THIS POINT INDICATE ERROR FREE PERFORMANCE OF THE DH11, THE SUB-SYSTEM SHOULD BE CAPABLE OF RUNNING SYSTEM SOFTWARE. THERE ARE, HOWEVER, CERTAIN SUBTLE OR INTERMITTENT PROBLEMS THAT COULD STILL CAUSE THE DH11 SUB-SYSTEM TO FAIL IN THE OPERATING SYSTEM ENVIRONMENT. IN THESE RARE CASES, THE USER WILL HAVE TO USE THE SYMPTOMS GATHERED FROM THE FAILING MODE TO ISOLATE THE PROBLEM. ONCE A SYMPTOM IS RECOGNIZED, THE TWO OTHER PROGRAMS IN CZDHN, THE ECHO TEST AND THE DATA PATTERNS/CABLE TESTS MAY

CZDMM-D-0
CZDMM.D.P11

MACY11 30A(1052) 10-MAR-78 08:05 PAGE 53
09-MAR-78 15:32

L 4

SEQ 0051
SEQ 0050

PROVE USEFUL AS A TROUBLESHOOTING AID TO DUPLICATE THE
PROBLEM FOR FAULT ISOLATION.

6.5 MAINTENANCE HEADER DESCRIPTION

EACH TEST IN THE LISTING IS PREFACED BY A STANDARD MAINTENANCE HEADER TO PROVIDE INFORMATION THAT WILL FACILITATE RAPID ISOLATION OF THE FAULT THAT CAUSED A PARTICULAR TEST TO FAIL. EACH HEADER HAS THE SAME FORMAT (SEE EXAMPLE BELOW) AS FOLLOWS:

TEST ABSTRACT: THIS IS A CAPSULE SUMMARY OF WHAT THE TEST IS DESIGNED TO TEST AND HOW IT OPERATES.

ERRORS: LISTS THE PARTICULAR ERROR CALLS INVOKED BY THE TEST WHEN A FAULT IS DETECTED.
(REFER TO PARA 3.1.2 AND 3.1.3 FOR A DETAILED DESCRIPTION OF THE ERROR CALL INFORMATION)

SYNC: LISTS ONE OR MORE SIGNALS THAT MAY BE USED TO SYNCHRONIZE THE OSCILLOSCOPE WHEN AN ERROR LOOP IS ESTABLISHED (SR09=1). FOR (H) SIGNALS USE (-) SLOPE TO TRIGGER ON THE TRAILING EDGE OF THE SIGNAL AND FOR (L) SIGNALS USE (+) SLOPE TO TRIGGER ON THE TRAILING EDGE.

DEBUG: CONTAINS SUGGESTIONS OF THINGS TO CHECK AND WHERE POSSIBLE THE MOST PROBABLE MODULE IS GIVEN.

KEY LOGIC: CONTAINS A LIST OF LOGIC SIGNALS AND/OR LOGIC COMPONENTS WITH MODULE NAMES AND PRINT NUMBERS TO RELATE THE TEST ROUTINE FUNCTION TO THE FUNCTIONAL AREAS OF LOGIC WITHIN THE PRINTS. WHERE POSSIBLE SIGNAL PIN NOS. ARE LISTED.

EXAMPLE: MAINTENANCE HEADER FOR TEST 1

TEST ABSTRACT:

THIS TEST ATTEMPTS TO REFERENCE EACH OF THE EIGHT REGISTERS IN THE DH11 SELECTED FOR TEST USING ITS ASSIGNED UNIBUS ADDRESS. IF ANY ADDRESS FAILS TO RESPOND A BUS ERROR TRAP VECTORS THE TEST TO THE ERROR SET-UP AND CALL ROUTINE. AFTER THE ERROR IS TYPED THE TEST WILL TEST THE NEXT DH11 ADDRESS IN SEQUENCE UNTIL ALL EIGHT ARE TESTED.

ERRORS:

- 1.) ERROR 1 REPORTS THAT THE REGISTER WHOSE ADDRESS IS IN R2 FAILED TO RESPOND WITH "SSYN" WHEN REFERENCED.

SYNC: (NONE)

DEBUG:

- 1.) PROBLEM IS MOST LIKELY THE M7277 MODULE.
- 2.) IF ALL EIGHT REGISTERS FAIL TO RESPOND, MAKE SURE THAT YOU CONFIGURED THE PROGRAM PROPERLY BEFORE STARTING. IF YOU DID, CHECK THE SETTINGS OF THE ADDRESS SELECT JUMPERS ON THE M7277 MODULE.
- 3.) IF ONE OR MORE RESPONDED PROPERLY, SET UP AN ERROR SCOPE LOOP AND BACKTRACK THROUGH THE LOGIC STARTING WITH THE KEY LOGIC SIGNALS LISTED BELOW.

KEY LOGIC:

M7277	SH3	SSYN H	CE2
		DEVICE RESPONDING L	E72-6
	SH4	DEVICE SELECTED H	E09-11

```

2191          .LIST ME,SEQ,BIN
2192          $SWR=165400
2193          .ENABLE ABS
2194          .TITLE CZDMM-D-0
2195          ;*COPYRIGHT (C) 1976,1978
2196          ;*DIGITAL EQUIPMENT CORP.
2197          ;*MAYNARD, MASS. 01754
2198          ;*
2199          ;*PROGRAM BY ED CROWLEY
2200          ;*
2201          ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
2202          ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
2203          ;*
2204          000001 $TN=1
2205          .SBTTL OPERATIONAL SWITCH SETTINGS
2206          ;*
2207          ;*      SWITCH              USE
2208          ;*      -----              -----
2209          ;*      15              HALT ON ERROR
2210          ;*      14              LOOP ON TEST
2211          ;*      13              INHIBIT ERROR TYPEOUTS
2212          ;*      11              INHIBIT ITERATIONS
2213          ;*      9              LOOP ON ERROR
2214          ;*      8              LOOP ON TEST IN SWR<7:0>
2215          .SBTTL TRAP CATCHER
2216          ;*
2217          000000          .=0
2218          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
2219          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
2220          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
2221          000174          .=174
2222          000174 000000  DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
2223          000176 000000  SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
2224          .SBTTL STARTING ADDRESS(ES)
2225          000200 000137 026162  JMP      @#INPARX          ;;JUMP TO STARTING ADDRESS OF PROGRAM
2226          000204 000137 002156  JMP      @#BEGIN          ;;BEGIN EXECUTION WITH DEFAULT PARAMETERS
2227          000210 000137 026146  JMP      @#INPARC          ;;INPUT PARAMETERS - DEVICE SELECTION ONLY
2228          ;*
2229          .SBTTL ACT11 HOOKS
2230          ;*
2231          ;:*****
2232          ;HOOKS REQUIRED BY ACT11
2233          000214          $SVPC=.          ;SAVE PC
2234          000046          .=46
2235          000046 020750          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
2236          000052          .=52
2237          000052 120000          .WORD 120000          ;;2)SET LOC.52 TO 120000
2238          000214          .= $SVPC          ;; RESTORE PC
2239          .SBTTL APT PARAMETER BLOCK
2240          ;*
2241          ;:*****
2242          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
2243          ;:*****
2244          000214          .$X=.          ;;SAVE CURRENT LOCATION
2245          000024          .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
2246          000024 000200          200          ;;FOR APT START UP

```

```
2247          000044          . =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
2248 000044 000214          $APTHDR ;;POINT TO APT HEADER BLOCK
2249          000214          . =.$X    ;;RESET LOCATION COUNTER
2250          ;;*****
2251          ;;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
2252          ;;INTERFACE SPEC.
2253
2254 000214          $APTHD:
2255 000214 000000          $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
2256 000216 001232          $MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
2257 000220 000036          $TSTM:  .WORD 30.    ;;RUN TIM OF LONGEST TEST
2258 000222 000170          $PASTM: .WORD 120.   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
2259 000224 000170          $UNITM: .WORD 120.   ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
2260 000226 000052          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
2261
2262
2263          .SBTTL BASIC DEFINITIONS
2264
2265          ;;INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
2266          001100          STACK= 1100
2267          .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
2268          .EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL
2269
2270          ;;*MISCELLANEOUS DEFINITIONS
2271          000011          HT= 11      ;;CODE FOR HORIZONTAL TAB
2272          000012          LF= 12      ;;CODE FOR LINE FEED
2273          000015          CR= 15      ;;CODE FOR CARRIAGE RETURN
2274          000200          CRLF= 200   ;;CODE FOR CARRIAGE RETURN-LINE FEED
2275          177776          PS= 177776  ;;PROCESSOR STATUS WORD
2276          .EQUIV PS,PSW
2277          177774          STKLMT= 177774 ;;STACK LIMIT REGISTER
2278          177772          PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
2279          177570          DSWR= 177570 ;;HARDWARE SWITCH REGISTER
2280          177570          DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
2281
2282          ;;*GENERAL PURPOSE REGISTER DEFINITIONS
2283          000000          R0= X0      ;;GENERAL REGISTER
2284          000001          R1= X1      ;;GENERAL REGISTER
2285          000002          R2= X2      ;;GENERAL REGISTER
2286          000003          R3= X3      ;;GENERAL REGISTER
2287          000004          R4= X4      ;;GENERAL REGISTER
2288          000005          R5= X5      ;;GENERAL REGISTER
2289          000006          R6= X6      ;;GENERAL REGISTER
2290          000007          R7= X7      ;;GENERAL REGISTER
2291          000006          SP= X6      ;;STACK POINTER
2292          000007          PC= X7      ;;PROGRAM COUNTER
2293
2294          ;;*PRIORITY LEVEL DEFINITIONS
2295          000000          PR0= 0      ;;PRIORITY LEVEL 0
2296          000040          PR1= 40     ;;PRIORITY LEVEL 1
2297          000100          PR2= 100    ;;PRIORITY LEVEL 2
2298          000140          PR3= 140    ;;PRIORITY LEVEL 3
2299          000200          PR4= 200    ;;PRIORITY LEVEL 4
2300          000240          PR5= 240    ;;PRIORITY LEVEL 5
2301          000300          PR6= 300    ;;PRIORITY LEVEL 6
2302          000340          PR7= 340    ;;PRIORITY LEVEL 7
```



```
2303
2304          ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
2305          SW15= 100000
2306          SW14= 40000
2307          SW13= 20000
2308          SW12= 10000
2309          SW11= 4000
2310          SW10= 2000
2311          SW09= 1000
2312          SW08= 400
2313          SW07= 200
2314          SW06= 100
2315          SW05= 40
2316          SW04= 20
2317          SW03= 10
2318          SW02= 4
2319          SW01= 2
2320          SW00= 1
2321          .EQUIV SW09,SW9
2322          .EQUIV SW08,SW8
2323          .EQUIV SW07,SW7
2324          .EQUIV SW06,SW6
2325          .EQUIV SW05,SW5
2326          .EQUIV SW04,SW4
2327          .EQUIV SW03,SW3
2328          .EQUIV SW02,SW2
2329          .EQUIV SW01,SW1
2330          .EQUIV SW00,SW0
2331
2332          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
2333          BIT15= 100000
2334          BIT14= 40000
2335          BIT13= 20000
2336          BIT12= 10000
2337          BIT11= 4000
2338          BIT10= 2000
2339          BIT09= 1000
2340          BIT08= 400
2341          BIT07= 200
2342          BIT06= 100
2343          BIT05= 40
2344          BIT04= 20
2345          BIT03= 10
2346          BIT02= 4
2347          BIT01= 2
2348          BIT00= 1
2349          .EQUIV BIT09,BIT9
2350          .EQUIV BIT08,BIT8
2351          .EQUIV BIT07,BIT7
2352          .EQUIV BIT06,BIT6
2353          .EQUIV BIT05,BIT5
2354          .EQUIV BIT04,BIT4
2355          .EQUIV BIT03,BIT3
2356          .EQUIV BIT02,BIT2
2357          .EQUIV BIT01,BIT1
2358          .EQUIV BIT00,BIT0
```

```
2359
2360      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
2361      000004      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
2362      000010      RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
2363      000014      TBITVEC=14        ;; "T" BIT
2364      000014      TRTVEC= 14        ;; TRACE TRAP
2365      000014      BPTVEC= 14        ;; BREAKPOINT TRAP (BPT)
2366      000020      IOTVEC= 20        ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
2367      000024      PWRVEC= 24        ;; POWER FAIL
2368      000030      EMTVEC= 30        ;; EMULATOR TRAP (EMT) **ERRGR**
2369      000034      TRAPVEC=34        ;; "TRAP" TRAP
2370      000060      TKVEC= 60         ;; TTY KEYBOARD VECTOR
2371      000064      TPVEC= 64         ;; TTY PRINTER VECTOR
2372      000240      PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
```

```

2373      .SBTTL COMMON TAGS
2374
2375      ;:*****
2376      ;:THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
2377      ;:USED IN THE PROGRAM.
2378
2379      001100      .SCTAG:      .=1100      ;:START OF COMMON TAGS
2380      001100      000000      .WORD      0      ;:CONTAINS THE TEST NUMBER
2381      001100      000000      .STSTNM: .BYTE 0      ;:CONTAINS ERROR FLAG
2382      001102      000      .SERFLG: .BYTE 0      ;:CONTAINS SUBTEST ITERATION COUNT
2383      001103      000      .SICNT:  .WORD 0      ;:CONTAINS SCOPE LOOP ADDRESS
2384      001104      000000      .SLPADR: .WORD 0      ;:CONTAINS SCOPE RETURN FOR ERRORS
2385      001106      000000      .SLPERR: .WORD 0      ;:CONTAINS TOTAL ERRORS DETECTED
2386      001110      000000      .SERTTL: .WORD 0      ;:CONTAINS ITEM CONTROL BYTE
2387      001112      000000      .SITEMB: .BYTE 0      ;:CONTAINS MAX. ERRORS PER TEST
2388      001114      000      .SERMAX: .BYTE 1      ;:CONTAINS PC OF LAST ERROR INSTRUCTION
2389      001115      001      .SERRPC: .WORD 0      ;:CONTAINS ADDRESS OF 'GOOD' DATA
2390      001116      000000      .SGDADR: .WORD 0      ;:CONTAINS ADDRESS OF 'BAD' DATA
2391      001120      000000      .SBDADR: .WORD 0      ;:CONTAINS 'GOOD' DATA
2392      001122      000000      .SGDDAT: .WORD 0      ;:CONTAINS 'BAD' DATA
2393      001124      000000      .SBDDAT: .WORD 0      ;:RESERVED--NOT TO BE USED
2394      001126      000000      .WORD      0
2395      001130      000000      .WORD      0
2396      001132      000000      .WORD      0
2397      001134      000      $AUTOB: .BYTE 0      ;:AUTOMATIC MODE INDICATOR
2398      001135      000      $INTAG: .BYTE 0      ;:INTERRUPT MODE INDICATOR
2399      001136      000000      .WORD      0
2400      001140      177570      $SWR:   .WORD      DSWR      ;:ADDRESS OF SWITCH REGISTER
2401      001142      177570      $DISPLAY: .WORD      DDISP      ;:ADDRESS OF DISPLAY REGISTER
2402      001144      177560      $TKS:   177560      ;:TTY KBD STATUS
2403      001146      177562      $TKB:   177562      ;:TTY KBD BUFFER
2404      001150      177564      $TPS:   177564      ;:TTY PRINTER STATUS REG. ADDRESS
2405      001152      177566      $TPB:   177566      ;:TTY PRINTER BUFFER REG. ADDRESS
2406      001154      000      $NULL:  .BYTE      0      ;:CONTAINS NULL CHARACTER FOR FILLS
2407      001155      002      $FILLS: .BYTE      2      ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
2408      001156      012      $FILLC: .BYTE     12      ;:INSERT FILL CHARS. AFTER A 'LINE FEED'
2409      001157      000      $TPFLG: .BYTE      0      ;:'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
2410      001160      000000      $REGAD: .WORD      0      ;:CONTAINS THE ADDRESS FROM
2411      ;:WHICH ($REGO) WAS OBTAINED
2412      001162      000000      $REG0:  .WORD      0      ;:CONTAINS (($REGAD)+0)
2413      001164      000000      $REG1:  .WORD      0      ;:CONTAINS (($REGAD)+2)
2414      001166      000000      $REG2:  .WORD      0      ;:CONTAINS (($REGAD)+4)
2415      001170      000000      $REG3:  .WORD      0      ;:CONTAINS (($REGAD)+6)
2416      001172      000000      $REG4:  .WORD      0      ;:CONTAINS (($REGAD)+10)
2417      001174      000000      $REG5:  .WORD      0      ;:CONTAINS (($REGAD)+12)
2418      001176      000000      $REG6:  .WORD      0      ;:CONTAINS (($REGAD)+14)
2419      001200      000000      $REG7:  .WORD      0      ;:CONTAINS (($REGAD)+16)
2420      001202      000000      $TMP0:  .WORD      0      ;:USER DEFINED
2421      001204      000000      $TMP1:  .WORD      0      ;:USER DEFINED
2422      001206      000000      $TMP2:  .WORD      0      ;:USER DEFINED
2423      001210      000000      $TMP3:  .WORD      0      ;:USER DEFINED
2424      001212      000000      $TMP4:  .WORD      0      ;:USER DEFINED
2425      001214      000000      $TMP5:  .WORD      0      ;:USER DEFINED
2426      001216      000000      $TMP6:  .WORD      0      ;:USER DEFINED
2427      001220      000000      $TMP7:  .WORD      0      ;:USER DEFINED
2428      001222      000000      $TIMES: 0      ;:MAX. NUMBER OF ITERATIONS

```

```

2429 001224 000000 $ESCAPE:0 ;:ESCAPE ON ERROR ADDRESS
2430 001226 077 $QUES: .ASCII /?/ ;:QUESTION MARK
2431 001227 015 $CRLF: .ASCII <15> ;:CARRIAGE RETURN
2432 001230 000012 $LF: .ASCIZ <12> ;:LINE FEED
2433 ;:*****
2434 .SBTTL APT MAILBOX-ETABLE
2435 ;:*****
2436 ;:*****
2437 .EVEN
2438 001232 $MAIL: ;:APT MAILBOX
2439 001232 000000 $MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
2440 001234 000000 $FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
2441 001236 000000 $TESTN: .WORD ATESTN ;:TEST NUMBER
2442 001240 000000 $PASS: .WORD APASS ;:PASS COUNT
2443 001242 000000 $DEVCT: .WORD ADEVCT ;:DEVICE COUNT
2444 001244 000000 $UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
2445 001246 000000 $MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS
2446 001250 000000 $MSGLG: .WORD AMSGLG ;:MESSAGE LENGTH
2447 001252 $ETABLE: ;:APT ENVIRONMENT TABLE
2448 001252 000 $ENV: .BYTE AENV ;:ENVIRONMENT BYTE
2449 001253 000 $ENVM: .BYTE AENVM ;:ENVIRONMENT MODE BITS
2450 001254 000000 $SWREG: .WORD ASWREG ;:APT SWITCH REGISTER
2451 001256 000000 $USWR: .WORD AUSWR ;:USER SWITCHES
2452 001260 000000 $CPUOP: .WORD ACPUOP ;:CPU TYPE,OPTIONS
2453 ;:BITS 15-11=CPU TYPE
2454 ;: 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2455 ;: 11/70=06,PDQ=07,Q=10
2456 ;:BIT 10=REAL TIME CLOCK
2457 ;:BIT 9=FLOATING POINT PROCESSOR
2458 ;:BIT 8=MEMORY MANAGEMENT
2459 001262 000 $MAMS1: .BYTE AMAMS1 ;:HIGH ADDRESS,M.S. BYTE
2460 001263 000 $MTYP1: .BYTE AMTYP1 ;:MEM. TYPE,BLK#1
2461 ;:MEM.TYPE BYTE -- (HIGH BYTE)
2462 ;: 900 NSEC CORE=001
2463 ;: 300 NSEC BIPOLAR=002
2464 ;: 500 NSEC MOS=003
2465 001264 000000 $MADR1: .WORD AMADR1 ;:HIGH ADDRESS,BLK#1
2466 ;:MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2467 001266 000 $MAMS2: .BYTE AMAMS2 ;:HIGH ADDRESS,M.S. BYTE
2468 001267 000 $MTYP2: .BYTE AMTYP2 ;:MEM.TYPE,BLK#2
2469 001270 000000 $MADR2: .WORD AMADR2 ;:MEM.LAST ADDRESS,BLK#2
2470 001272 000 $MAMS3: .BYTE AMAMS3 ;:HIGH ADDRESS,M.S.BYTE
2471 001273 000 $MTYP3: .BYTE AMTYP3 ;:MEM.TYPE,BLK#3
2472 001274 000000 $MADR3: .WORD AMADR3 ;:MEM.LAST ADDRESS,BLK#3
2473 001276 000 $MAMS4: .BYTE AMAMS4 ;:HIGH ADDRESS,M.S.BYTE
2474 001277 000 $MTYP4: .BYTE AMTYP4 ;:MEM.TYPE,BLK#4
2475 001300 000000 $MADR4: .WORD AMADR4 ;:MEM.LAST ADDRESS,BLK#4
2476 001302 000000 $VECT1: .WORD AVECT1 ;:INTERRUPT VECTOR#1,BUS PRIORITY#1
2477 001304 000000 $VECT2: .WORD AVECT2 ;:INTERRUPT VECTOR#2BUS PRIORITY#2
2478 001306 000000 $BASE: .WORD ABASE ;:BASE ADDRESS OF EQUIPMENT UNDER TFST
2479 001310 000000 $DEVN: .WORD ADEVN ;:DEVICE MAP
2480 001312 000000 $CDW1: .WORD ACDW1 ;:CONTROLLER DESCRIPTION WORD#1
2481 001314 000000 $CDW2: .WORD ACDW2 ;:CONTROLLER DESCRIPTION WORD#2
2482 001316 000000 $DDW0: .WORD ADDW0 ;:DEVICE DESCRIPTOR WORD#0
2483 001320 000000 $DDW1: .WORD ADDW1 ;:DEVICE DESCRIPTOR WORD#1
2484 001322 000000 $DDW2: .WORD ADDW2 ;:DEVICE DESCRIPTOR WORD#2

```


2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557

001356

001356 030360
001360 030427
001362 030506
001364 000000

001366 030524
001370 030562
001372 030660
001374 000000

001376 030702
001400 030562
001402 030660
001404 000000

001406 030760
001410 030562
001412 030660
001414 000000

001416 031016
001420 030562
001422 030660
001424 000000

001426 031053
001430 030562
001432 030660
001434 000000

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ;;POINTS TO THE ERROR MESSAGE
;* DH ;;POINTS TO THE DATA HEADER
;* DT ;;POINTS TO THE DATA
;* DF ;;POINTS TO THE DATA FORMAT

\$ERRTB:

;ERROR TABLE ITEM FOR ERROR MESSAGE 1

EM1 ;'DH11 REGISTER REFERENCE CAUSED TIMEOUT'
DH1 ;' (PC) (PS) (SP) TEST DEVADR REGADR ''
DT1 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 2

EM2 ;'SYSTEM CONTROL REGISTER ERROR'
DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
DT2 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 3

EM3 ;'DH11 MASTER CLEAR FAILED TO CLR SPECIFIED REG''
DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
DT2 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 4

EM4 ;'LINE PARAMETER REGISTER ERROR''
DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
DT2 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 5

EM5 ;'BREAK CONTROL REGISTER ERROR''
DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
DT2 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR MESSAGE 6

EM6 ;'SILO STATUS REGISTER ERROR''
DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
DT2 ;\$ERRPC,\$TMPO,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4
0 ;PRINT ALL OCTAL

```
2558
2559 ;ERROR TABLE ITEM FOR ERROR MESSAGE 7
2560
2561 001436 031106 EM7 ;'CURRENT ADDRESS REGISTER ERROR - LINE #XX''
2562 001440 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2563 001442 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2564 001444 000000 0 ;PRINT ALL OCTAL
2565
2566 ;ERROR TABLE ITEM FOR ERROR MESSAGE 10
2567
2568 001446 031160 EM10 ;'BYTE COUNTER REGISTER ERROR - LINE #XX''
2569 001450 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2570 001452 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2571 001454 000000 0 ;PRINT ALL OCTAL
2572
2573 ;ERROR TABLE ITEM FOR ERROR MESSAGE 11
2574
2575 001456 031227 EM11 ;'UNEXPECTED DH11 RCVR INTERRUPT''
2576 001460 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2577 001462 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2578 001464 000000 0 ;PRINT ALL OCTAL
2579
2580 ;ERROR TABLE ITEM FOR ERROR MESSAGE 12
2581
2582 001466 031266 EM12 ;'UNEXPECTED DH11 XMITTR INTERRUPT''
2583 001470 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2584 001472 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2585 001474 000000 0 ;PRINT ALL OCTAL
2586
2587 ;ERROR TABLE ITEM FOR ERROR MESSAGE 13
2588
2589 001476 031327 EM13 ;'CHAR AVAILABLE FAILED TO GENERATE RCVR INTERRUPT''
2590 001500 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2591 001502 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2592 001504 000000 0 ;PRINT ALL OCTAL
2593
2594 ;ERROR TABLE ITEM FOR ERROR MESSAGE 14
2595
2596 001506 031410 EM14 ;'TRANSMITTER NPR LOGIC ERROR - LINE # ''
2597 001510 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2598 001512 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2599 001514 000000 0 ;PRINT ALL OCTAL
2600
2601 ;ERROR TABLE ITEM FOR ERROR MESSAGE 15
2602
2603 001516 031457 EM15 ;'XMITTR FAILED TO INTERRUPT - LINE # ''
2604 001520 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2605 001522 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2606 001524 000000 0 ;PRINT ALL OCTAL
2607
2608 ;ERROR TABLE ITEM FOR ERROR MESSAGE 16
2609
2610 001526 031525 EM16 ;'RCVR FAILED TO INTERRUPT''
2611 001530 030562 DH2 ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ''
2612 001532 030660 DT2 ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2613 001534 000000 0 ;PRINT ALL OCTAL
```

```
2614
2615 ;ERROR TABLE ITEM FOR ERROR MESSAGE 17
2616
2617 001536 031556 EM17 ;"TRANSMITTER TIMING ERROR - LINE # ""
2618 001540 031622 DH6 ;" (PC) (PS) (SP) TEST DEVADR SPEED TIMEB TIMEC""
2619 001542 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2620 001544 000000 0 ;PRINT ALL OCTAL
2621
2622 ;ERROR TABLE ITEM FOR ERROR MESSAGE 20
2623
2624 001546 031720 EM20 ;RECEIVER TIMING ERROR - LINE # ""
2625 001550 031622 DH6 ;" (PC) (PS) (SP) TEST DEVADR SPEED TIMEB TIMEC""
2626 001552 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2627 001554 000000 0 ;PRINT ALL OCTAL
2628
2629 ;ERROR TABLE ITEM FOR ERROR MESSAGE 21
2630
2631 001556 031761 EM21 ;"RCVR FAILED TO INTERRUPT - LINE # ""
2632 001560 030562 DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
2633 001562 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2634 001564 000000 0 ;PRINT ALL OCTAL
2635
2636 ;ERROR TABLE ITEM FOR ERROR MESSAGE 22
2637
2638 001566 032025 EM22 ;"CHAR AVAIL FAILED TO SET ON TIME - LINE # ""
2639 001570 030562 DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
2640 001572 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2641 001574 000000 0 ;PRINT ALL OCTAL
2642
2643 ;ERROR TABLE ITEM FOR ERROR MESSAGE 23
2644
2645 001576 032101 EM23 ;"BASIC DATA TEST ERROR - LINE # ""
2646 001600 032142 DH7 ;" (PC) (PS) (SP) TEST DEVADR CHRLNG WAS S/B ""
2647 001602 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2648 001604 000000 0 ;PRINT ALL OCTAL
2649
2650 ;ERROR TABLE ITEM FOR ERROR MESSAGE 24
2651
2652
2653 001606 032237 EM24 ;"AUTO ECHO TEST ERROR - LINE # ""
2654 001610 030562 DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
2655 001612 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2656 001614 000000 0 ;PRINT ALL OCTAL
2657
2658 ;ERROR TABLE ITEM FOR ERROR MESSAGE 25
2659
2660 001616 032277 EM25 ;"BREAK BIT TEST ERROR - LINE # ""
2661 001620 030562 DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
2662 001622 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2663 001624 000000 0 ;PRINT ALL OCTAL
2664
2665 ;ERROR TABLE ITEM FOR ERROR MESSAGE 26
2666
2667 001626 032337 EM26 ;"HALF-DUPLEX TEST ERROR - LINE # ""
2668 001630 030562 DH2 ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B ""
2669 001632 030660 DT2 ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
```



```
2670 001634 000000          0          ;PRINT ALL OCTAL
2671
2672          ;ERROR TABLE ITEM FOR ERROR MESSAGE 27
2673
2674 001636 032401          EM27          ;"UNEXPECTED BUS ERROR TRAP"
2675 001640 032433          DH3           ; (PC) (PS) (SP) TEST TRPPC TRPPS
2676 001642 032512          DT3           ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2
2677 001644 000000          0           ;PRINT ALL OCTAL
2678
2679          ;ERROR TABLE ITEM FOR ERROR MESSAGE 30
2680
2681 001646 032530          EM30          ;"UNEXPECTED RSVD INSTR TRAP"
2682 001650 032433          DH3           ; (PC) (PS) (SP) TEST TRPPC TRPPS
2683 001652 032512          DT3           ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2
2684 001654 000000          0           ;PRINT ALL OCTAL
2685
2686          ;ERROR TABLE ITEM FOR ERROR MESSAGE 31
2687
2688 001656 032563          EM31          ;"AUTO ECHO DATA COMPARE ERROR - LINE # "
2689 001660 032633          DH4           ; (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "
2690 001662 030660          DT2           ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2691 001664 000000          0           ;PRINT ALL OCTAL
2692
2693          ;ERROR TABLE ITEM FOR ERROR MESSAGE 32
2694
2695 001666 032730          EM32          ;"AUTO ECHO TEST TIMEOUT - LINE # "
2696 001670 032772          DH5           ;" (PC) (LPRG) TEST"
2697 001672 033020          DT4           ;$ERRPC,$TMPO,$TMP2
2698 001674 000000          0           ;PRINT ALL OCTAL
2699
2700          ;ERROR TABLE ITEM FOR ERROR MESSAGE 33
2701
2702
2703 001676 033030          EM33          ;"PARITY LOGIC TEST ERROR - LINE # "
2704 001700 030562          DH2           ;" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B"
2705 001702 030660          DT2           ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2706 001704 000000          0           ;PRINT ALL OCTAL
2707
2708          ;ERROR TABLE ITEM FOR ERROR MESSAGE 34
2709
2710 001706 033073          EM34          ;"MULTI-LINE PARITY DATA TEST ERROR - LINE # - SUBTEST # "
2711 001710 032633          DH4           ;" (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "
2712 001712 030660          DT2           ;$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2713 001714 000000          0           ;PRINT ALL OCTAL
2714
2715          ;ERROR TABLE ITEM FOR ERROR MESSAGE 35
2716
2717 001716 033166          EM35          ;"MULTI-LINE PARITY DATA TEST TIMEOUT"
2718 001720 033232          DH14          ;" (PC) (LPRG) LINACTION "
2719 001722 033262          DT6           ;"$ERRPC,$TMPO,$TMP3"
2720 001724 000000          0           ;PRINT ALL OCTAL
2721
2722          ;ERROR TABLE ITEM FOR ERROR MESSAGE 36
2723
2724 001726 033272          EM36          ;CHAR AVAILABLE TIMEOUT"
2725 001730 032772          DH5           ;" (PC) (LPRG) TEST"
```

```
2726 001732 033020          DT4          :$ERRPC,$TMP0,$TMP2
2727 001734 000000          0          :PRINT ALL OCTAL
2728
2729          :ERROR TABLE ITEM FOR ERROR MESSAGE 37
2730
2731 001736 033321          EM37         :'"DATA COMPARE ERROR - LINE # "'
2732 001740 032633          DH4         :'" (PC) (PS) (SP) TEST WASADR SBADR WAS S/B "'
2733 001742 030660          DT2         :$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2734 001744 000000          0          :PRINT ALL OCTAL
2735
2736          :ERROR TABLE ITEM FOR ERROR MESSAGE 40
2737
2738 001746 033357          EM40         :'"BUFFER ACTIVE REG ERROR - LINE # "'
2739 001750 030562          DH2         :'" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "'
2740 001752 030660          DT2         :$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2741 001754 000000          0          :PRINT ALL OCTAL
2742
2743          :ERROR TABLE ITEM FOR ERROR MESSAGE 41
2744
2745 001756 033422          EM41         :'"RCVR FALSE INTERRUPT"'
2746 001760 032772          DH5         :'" (PC) (LPRG) TEST"'
2747 001762 033020          DT4         :$ERRPC,$TMP0,$TMP2
2748 001764 000000          0          :PRINT ALL OCTAL
2749
2750          :ERROR TABLE ITEM FOR ERROR MESSAGE 42
2751
2752 001766 033447          EM42         :'"SILO OVERFLOW ERROR"'
2753 001770 032772          DH5         :'" (PC) (LPRG) TEST"'
2754 001772 033020          DT4         :$ERRPC,$TMP0,$TMP2
2755 001774 000000          0          :PRINT ALL OCTAL
2756
2757          :ERROR TABLE ITEM FOR ERROR MESSAGE 43
2758
2759 001776 033473          EM43         :'"SILO OVERFLOW FAILED TO GENERATE RCVR INTERRUPT"'
2760 002000 030562          DH2         :'" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "'
2761 002002 030660          DT2         :$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2762 002004 000000          0          :PRINT ALL OCTAL
2763
2764          :ERROR TABLE ITEM FOR ERROR MESSAGE 44
2765
2766 002006 033553          EM44         :'"NON EX MEMORY FAILED TO GENERATE XMITTR INTERRUPT"'
2767 002010 030562          DH2         :'" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "'
2768 002012 030660          DT2         :$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2769 002014 000000          0          :PRINT ALL OCTAL
2770
2771          :ERROR TABLE ITEM FOR ERROR MESSAGE 45
2772
2773 002016 033635          EM45         :'"XMIT DONE FAILED TO CENERATE XMITTR INTERRUPT"'
2774 002020 030562          DH2         :'" (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "'
2775 002022 030660          DT2         :$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2776 002024 000000          0          :PRINT ALL OCTAL
2777
2778          :ERROR TABLE ITEM FOR ERROR MESSAGE 46
2779
2780 002026 033713          EM46         :'"CURRENT ADDRESS MEMORY PATTERNS TEST ERROR - LINE # "'
2781 002030 034001          DH10        :'" (PC) LINEWR PATTRN TEST DEVADR REGADR WAS S/B "'
```

```
2782 002032 034076          DT5          ;$ERRPC,$TMP0,$TMP1,$REG0,$REG1,$REG2,$REG3,$REG4
2783 002034 000000          0          ;PRINT ALL OCTAL
2784
2785          ;ERROR TABLE ITEM FOR ERROR MESSAGE 47
2786
2787 002036 034120          EM47          ;'BYTE COUNT MEMORY PATTERNS TEST ERROR - LINE # ''
2788 002040 034001          DH10          ; (PC) LINEWR PATTRN TEST DEVADR REGADR WAS S/B''
2789 002042 034076          DT5          ;$ERRPC,$TMP0,$TMP1,$REG0,$REG1,$REG2,$REG3,$REG4
2790 002044 000000          0          ;PRINT ALL OCTAL
2791
2792          ;ERROR TABLE ITEM FOR ERROR MESSAGE 50
2793
2794 002046 034201          EM50          ;'TEST TIMEOUT WAITING FOR XMIT DONE - LINE # '
2795 002050 030562          DH2          ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B''
2796 002052 030660          DT2          ;'$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4''
2797 002054 000000          0          ;PRINT ALL OCTAL
2798
2799          ;ERROR TABLE ITEM FOR ERROR MESSAGE 51
2800
2801 002056 034257          EM51          ;'NPR LOGIC TEST 2 ERROR''
2802 002060 034306          DH11          ;' (PC) LINACT LINCHK TEST DEVADR REGADR WAS S/B''
2803 002062 034076          DT5          ;$ERRPC,$TMP0,$TMP1,$REG0,$REG1,$REG2,$REG3,$REG4''
2804 002064 000000          0          ;PRINT ALL OCTAL
2805
2806          ;ERROR TABLE ITEM FOR ERROR MESSAGE 52
2807
2808 002066 034403          EM52          ;'BASIC DATA COMPARE ERROR''
2809 002070 030562          DH2          ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B''
2810 002072 030660          DT2          ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4''
2811 002074 000000          0          ;PRINT ALL OCTAL
2812
2813          ;ERROR TABLE ITEM FOR ERROR MESSAGE 53
2814
2815 002076 034201          EM50          ;'TEST TIMEOUT WAITING FOR XMIT DONE - LINE # ''
2816 002100 034434          DH12          ;' (PC) SPEED (SP) TEST DEVADR REGADR WAS S/B''
2817 002102 030660          DT2          ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2818 002104 000000          0          ;PRINT ALL OCTAL
2819
2820          ;ERROR TABLE ITEM FOR ERROR MESSAGE 54
2821
2822 002106 032025          EM22          ;'CHAR AVAIL FAILED TO SET ON TIME - LINE # ''
2823 002110 034434          DH12          ;' (PC) SPEED (SP) TEST DEVADR REGADR WAS S/B''
2824 002112 030660          DT2          ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2825 002114 000000          0          ;PRINT ALL OCTAL
2826
2827          ;ERROR TABLE ITEM FOR ERROR MESSAGE 55
2828
2829 002116 032025          EM22          ;'CHAR AVAIL FAILED TO SET ON TIME - LINE # ''
2830 002120 034531          DH13          ;' (PC) (PS) (SP) TEST DEVADR CHRLNG SCRNAS SCRS/B
2831 002122 030660          DT2          ;$ERRPC,$TMP0,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4
2832 002124 000000          0          ;PRINT ALL OCTAL
2833
2834          ;ERROR TABLE ITEM FOR ERROR MESSAGE 56
2835
2836 002126 034630          EM56          ;'OVERRUN BIT FAILED TO SET - LINE # ''
2837 002130 030562          DH2          ;' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B''
```

```
2838 002132 030660          DT2          :$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4''
2839 002134 000000          0          :PRINT ALL OCTAL
2840
2841 ;ERROR TABLE ITEM FOR ERROR MESSAGE 57
2842
2843 002136 034675          FM57          :''STORAGE OVERFLOW BIT FAILED - LINE # ''
2844 002140 030562          DH2          :'' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B''
2845 002142 030660          DT2          :$ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4''
2846 002144 000000          0          :PRINT ALL OCTAL
2847
2848 ;ERROR TABLE ITEM FOR ERROR MESSAGE 60
2849 002146 036140          MSG4          :MODEM CONTROL ERROR.. RUN DZDHK
2850 002150 034744          DH15          :'' (PC) DEVADR LINE ''
2851 002152 033020          DT4          :$ERRPC,$TMPO,$TMP2
2852 002154 000000          0
2853
```

```

2854 002156 004737 027244 BEGIN: JSR PC,DCACHE ;DISABLE CACHE ;:++D
2855 002162 005000 CLR RO ;INIT RO TO INDICATE DEFAULT PARAMETERS
2856 002164 005037 030346 BEGINA: CLR TITFLG ;INIT TITLE MESSAGE FLAG
2857 .SBTTL INITIALIZE THE COMMON TAGS
2858 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2859 002170 012706 001100 MOV #SCMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
2860 002174 005026 CLR (R6)+ ;:CLEAR MEMORY LOCATION
2861 002176 022706 001140 CMP #SWR,R6 ;:DONE?
2862 002202 001374 BNE -6 ;:LOOP BACK IF NO
2863 002204 012706 001100 MOV #STACK,SP ;:SETUP THE STACK POINTER
2864 ;;INITIALIZE A FEW VECTORS
2865 002210 012737 021004 000020 MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
2866 002216 012737 000340 000022 MOV #340,@IOTVEC+2 ;:LEVEL 7
2867 002224 012737 021274 000030 MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
2868 002232 012737 000340 000032 MOV #340,@EMTVEC+2 ;:LEVEL 7
2869 002240 012737 024060 000034 MOV #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
2870 002246 012737 000340 000036 MOV #340,@TRAPVEC+2 ;:LEVEL 7
2871 002254 012737 024142 000024 MOV #SPURDN,@PURVEC ;:POWER FAILURE VECTOR
2872 002262 012737 000340 000026 MOV #340,@PURVEC+2 ;:LEVEL 7
2873 002270 013737 020716 020710 MOV SENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
2874 002276 005037 001222 CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
2875 002302 005037 001224 CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
2876 002306 112737 000001 001115 MOVB #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
2877 002314 012737 002314 001106 MOV #,$LPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
2878 002322 012737 002322 001110 MOV #,$LPERR ;:SETUP THE ERROR LOOP ADDRESS
2879 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2880 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2881 002330 013746 000004 MOV @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
2882 002334 012737 002370 000004 MOV #64$,@ERRVEC ;:SET UP ERROR VECTOR
2883 002342 012737 177570 001140 MOV #DSWR,SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
2884 002350 012737 177570 001142 MOV #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
2885 002356 022777 177777 176554 CMP #-1,@SWR ;:TRY TO REFERENCE HARDWARE SWR
2886 002364 001012 BNE 66$ ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
2887 ;:AND THE HARDWARE SWR IS NOT = -1
2888 002366 000403 BR 65$ ;:BRANCH IF NO TIMEOUT
2889 002370 012716 002376 64$: MOV #65$,(SP) ;:SET UP FOR TRAP RETURN
2890 002374 000002 RTI
2891 002376 012737 000176 001140 65$: MOV #SWREG,SWR ;:POINT TO SOFTWARE SWR
2892 002404 012737 000174 001142 MOV #DISPREG,DISPLAY
2893 002412 012637 000004 66$: MOV (SP)+,@ERRVEC ;:RESTORE ERROR VECTOR
2894
2895 002416 005037 001240 CLR $PASS ;:CLEAR PASS COUNT
2896 002422 132737 000200 001253 BITB #APTSIZE,$ENVM ;:TEST USER SIZE UNDER APT
2897 002430 001403 BEQ 67$ ;:YES,USE NON-APT SWITCH
2898 002432 012737 001254 001140 MOV #SSWREG,SWR ;:NO,USE APT SWITCH REGISTER
2899 002440 67$:
2900 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2901 002440 005737 000042 TST @#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
2902 002444 001012 BNE 68$ ;:BRANCH IF YES
2903 002446 123727 001252 000001 CMPB $ENV,#1 ;:ARE WE RUNNING UNDER APT?
2904 002454 001406 BEQ 68$ ;:BRANCH IF YES
2905 002456 023727 001140 000176 CMP SWR,#SWREG ;:SOFTWARE SWITCH REG SELECTED?
2906 002464 001005 BNE 69$ ;:BRANCH IF NO
2907 002466 104406 GTSWR ;:GET SOFT-SWR SETTINGS
2908 002470 000403 BR 69$
2909 002472 112737 000001 001134 68$: MOVB #1,$AUTOB ;:SET AUTO-MODE INDICATOR

```

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78 08:05 PAGE 71

GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0069
SEQ 0068

2910 002500

698:

CZDMM-D-0
CZDHMD.P11MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78 08:05 PAGE 72

GET VALUE FOR SOFTWARE SWITCH REGISTER

SEQ 0070
SEQ 0069

```

2911 002500 012737 026640 000004 START1: MOV #BUSER,ERRVEC ;SET UP THE BUS ERROR VECTOR
2912 002506 012737 000340 000006 MOV #340,ERRVEC+2
2913 002514 012737 026714 000010 MOV #RESERR,RESVEC ;SET UP THE RSVD INSTR VECTOR
2914 002522 012737 000340 000012 MOV #340,RESVEC+2
2915 002530 005737 030346 TST TITFLG ;HAVE WE TYPED TITLE ONCE ?
2916 002534 001012 BNE 1$ ;BR IF YES
2917 002536 104401 TYPE ;GO TYPE PROGRAM TITLE
2918 002540 035000 TITLE
2919 002542 005137 030346 COM TITFLG ;SET FLAG - TYPE TITLE ONLY ONCE PER LOAD
2920 002546 032777 000001 176364 BIT #BIT0,@SWR ;DO WE WANT TO AUTOSIZE?
2921 002554 001002 BNE 1$ ;BRANCH IF NOT.
2922 002556 004737 025140 JSR PC,AUTOSZ ;GO AUTOSIZE.
2923 002562 005737 030000 1$: TST VCFLG ;START AT 200 ??
2924 002566 001413 BEQ 11$ ;BR IF NOT
2925 002570 032777 000001 176342 BIT #BIT0,@SWR ;ARE PARAMETERS TO BE INPUT MANUALLY?
2926 002576 001003 BNE 9$ ;BRANCH IF YES.
2927 002600 013700 030306 MOV ADRVEC,R0 ;OTHERWISE, GET ADDRESSES BETWEEN VECTORS FROM AUTOSIZER
2928 002604 000402 BR 10$
2929 002606 004737 026110 9$: JSR PC,INPARA ;GO ASK FOR PARAMETERS
2930 002612 005037 030000 10$: CLR VCFLG ;RE INIT VECTOR FLAG
2931 002616 005700 11$: TST R0 ;USE DEFAULT PARAMETERS ?
2932 002620 001407 BEQ START2 ;BR IF YES
2933 002622 022700 177777 CMP #-1,R0 ;CHANGE DH SELECT PARAM ONLY ?
2934 002626 001002 BNE 2$ ;BR IF NOT
2935 002630 000137 026270 JMP INPAR3 ;GO ASK FOR SELECT PARAM.
2936 002634 000137 026200 2$: JMP INPAR ;GO ASK FOR ALL PARAMETERS
2937
2938 002640 012737 027676 030326 START2: MOV #DHADTB-2,ADPTR ;GET POINTER TO ADDRESS TABLE
2939 002646 012737 027736 030330 MOV #DHSVCTB-2,VCPTR ;GET POINTER TO VECTOR TABLE
2940 002654 012737 030000 030332 MOV #BRLVL-2,BRPTR ;GET POINTER TO BR LEVEL TABLE
2941 002662 012737 177777 030320 MOV #-1,DHNUM ;START WITH DH #00
2942 002670 012737 000001 027306 MOV #1,SELMSK ;SET UP DH11 BIT TEST MARKER
2943
2944 002676 005237 030320 RESTRT: INC DHNUM ;GENERATE DH11 DEV NUMBER
2945 002702 062737 000002 030326 ADD #2,ADPTR ;UPDATE TABLE POINTERS
2946 002710 062737 000002 030330 ADD #2,VCPTR
2947 002716 062737 000002 030332 ADD #2,BRPTR
2948 002724 033737 027306 027310 BIT SELMSK,DHSEL ;TEST FOR SELECTED DH11
2949 002732 001004 BNE RSTRTA ;BR IF SELECTED FOR TEST
2950 002734 006337 027306 REST1: ASL SELMSK ;SHIFT MARKER TO TEST NEXT DH11
2951 002740 001737 BEQ START2 ;BR IF 16 TESTED - START OVER
2952 002742 000755 BR RESTRT ;GO TEST IF THIS ONE SELECTED
2953 002744 017737 025356 027302 RSTRTA: MOV @ADPTR,DHADR ;SET UP DH11 ADDRESS
2954 002752 017737 025352 027304 MOV @VCPTR,DHVCT ;SET UP THE DH11 VECTOR ENTRY
2955 002760 017737 025346 030316 MOV @BRPTR,DHRLVL ;GET BR LEVEL VALUES
2956 002766 004537 024636 JSR R5,SUNUM ;GO SET DH NUMBER IN THE MESSAGE BUFFER
2957 002772 030320 DHNUM
2958 002774 035057 TITLE2+20
2959 002776 104401 TYPE ;GO PRINT "TESTING DH11 #XX"
2960 003000 035037 TITLE2
2961 003002 004737 024330 JSR PC,LDTBF1 ;GO LOAD XMITTR OUTPUT BUFFER WITH
2962 ;BINARY COUNT PATTERN
2963 003006 012737 003006 001106 MOV #.,$LPADR ;INIT SCOPE LOOP RETURN

```

2964
2965
2966
2967 003014 000004
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005 003016 010102
3006 003020 010205
3007 003022 062705 000020
3008 003026 013746 000004
3009 003032 012737 003060 000004
3010 003040 162702 000002
3011
3012 003044 062702 000002
3013 003050 020205
3014 003052 001412
3015 003054 005712
3016 003056 000772
3017
3018 003060 004737 024352
3019 003064 022626

:*TEST 1 CHECK SSYN RESPONSE FROM ALL DH11 REGISTERS

TST1: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST ATTEMPTS TO REFERENCE EACH OF THE EIGHT REGISTERS IN THE DH11 SELECTED FOR TEST USING ITS ASSIGNED UNIBUS ADDRESS. IF ANY ADDRESS FAILS TO RESPOND A BUS ERROR TRAP VECTORS THE TEST TO THE ERROR SET-UP AND CALL ROUTINE. AFTER THE ERROR IS TYPED THE TEST WILL TEST THE NEXT DH11 ADDRESS IN SEQUENCE UNTIL ALL EIGHT ARE TESTED.

ERRORS:

1.) ERROR 1 REPORTS THAT THE REGISTER WHOSE ADDRESS IS IN R2 FAILED TO RESPOND WITH 'SSYN' WHEN REFERENCED.

SYNC: (NONE)

DEBUG:

- 1.) PROBLEM IS MOST LIKELY THE M7277 MODULE.
- 2.) IF ALL EIGHT REGISTERS FAIL TO RESPOND, MAKE SURE THAT YOU CONFIGURED THE PROGRAM PROPERLY BEFORE STARTING. IF YOU DID, CHECK THE SETTINGS OF THE ADDRESS SELECT JUMPERS ON THE M7277 MODULE.
- 3.) IF ONE OR MORE RESPONDED PROPERLY, SET UP AN ERROR SCOPE LOOP AND BACKTRACK THROUGH THE LOGIC STARTING WITH THE KEY LOGIC SIGNALS LISTED BELOW.

KEY LOGIC:

	M7277	SH3	SSYN H	CE2
			DEVICE RESPONDING L	E72-6
		SH4	DEVICE SELECTED H	E09-11
%				
	MOV	R1,R2		;COPY IT INTO R2
	MOV	R2,R5		;ALSO R5
	ADD	#20,R5		;R5 WILL TELL US WHEN WE'VE TESTED ALL 8
	MOV	ERRVEC,-(SP)		;SAVE BUS ERROR VECTOR
	MOV	#3\$,ERRVEC		;GO TO 3\$ IF REG FAILS TO RESPOND
	SUB	#2,R2		;SO WE START WITH FIRST REG
1\$:	ADD	#2,R2		;POINT TO A DH11 REGISTER
	CMP	R2,R5		;TESTED ALL EIGHT ??
	BEQ	4\$;BR IF YES
2\$:	TST	(R2)		;ACCESS DH11 REG ADDR
	BR	1\$;BR WHEN ALL 8 ARE DONE
3\$:	JSR	PC,SUER1		;GO SET UP ERROR INFO
	CMP	(SP)+,(SP)+		;FIX SP BECAUSE OF TRAP

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052) 10-MAR-78
09-MAR-78 15:32 T1

08:05 PAGE 74
CHECK SSYN RESPONSE FROM ALL DH11 REGISTERS

SEQ 0072
SEQ 0071

```

3020 003066 012737 003054 001110      MOV    #2$, $LPERR      ;SET UP ERRCR LOOP RETURN
3021 003074 104001                      ERROR  1                ;DH11 REGISTER FAILED TO RESPOND TO MSYN
3022                                     BR     1$                ;GO TEST NEXT ONE
3023 003076 000762                      MOV    (SP)+,ERRVEC     ;RESTORE BUS ERROR VECTOR
3024 003100 012637 000004      4$:

```

3025
3026
3027
3028 003104 000004
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074 003106 012705 027320
3075 003112 010125
3076 003114 010125
3077 003116 010125
3078 003120 010125
3079 003122 062745 000016
3080 003126 062745 000014

*TEST 2 TEST THAT 'MASTER CLR' CAN CLEAR THE 'SCR','LPR','BKR',AND 'SSR' REGS

TST2: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST SETS ALL WRITEABLE BITS IN THE TARGET REGISTER (SCR, LPR, BKR, AND SSR) THEN SETS BIT11 IN THE 'SCR' (MASTER CLEAR) AND CHECKS THAT IT INDEED CLEARED ALL BITS IN THE TARGET REGISTER. IT PERFORMS THIS SEQUENCE FOR ALL TARGET REGISTERS. IF A REGISTER FAILS TO CLEAR PROPERLY, THE ERROR IS REPORTED, AND THEN THE ROUTINE TESTS THE NEXT REGISTER IN SEQUENCE.

ERRORS:

- 1.) ERROR 3 REPORTS THAT THE REGISTER WHOSE ADDRESS IS IN R2 FAILED TO CLEAR WHEN MASTER CLEAR WAS ACTIVATED.

SYNC:

M7289 SH6 SCR 11 H (MASTER CLEAR) FK2

DEBUG:

- 1.) IF THE ERROR REPORTS INDICATE THAT ALL REGISTERS ARE FAILING, ESTABLISH AN ERROR SCOPE LOOP AND PROCEED TO BACKTRACK THROUGH THE KEY LOGIC SIGNALS LISTED BELOW.
- 2.) IF ONLY ONE REGISTER FAILS WITH ALL SET BITS -- THEN LET TESTS 03-10 RUN -- THEY WILL PROBABLY GIVE BETTER ISOLATION. USE ONE OF THESE TESTS TO DEBUG THE FAULT.

KEY LOGIC:

M7277 SH3 INIT A L FR2
INIT B L FM2
INIT A H EF2
INIT B H FV2
SH4 LOAD SCR HIGH BYTE H CP1
M7289 SH6 SCR 11 H (MASTER CLEAR) FK2
M7278 SH3 BUFF DATA 11 H AA1

X

MOV #MSTCLR,R5 ;GET POINTER TO ADDRESS TABLE
MOV R1,(R5)+ ;SET UP THE TEST ADDRESS TABLE SO THAT
MOV R1,(R5)+ ;IT CONTAINS THE ADDRESSES OF THE
MOV R1,(R5)+ ;SCR,LPR,BKR, AND SSR REGISTERS
MOV R1,(R5)+
ADD #SSR,-(R5) ;GENERATE SSR ADDRESS
ADD #BKR,-(R5) ;GENERATE BKR ADDRESS

```
3081 003132 062745 000004      ADD    #LPR,-(R5)      ;GENERATE LPR ADDRESS
3082 003136 005745              TST    -(R5)          ;POINT R5 TO FIRST ADDR ENTRY (SCR)
3083 003140 005004              CLR    R4             ;RESULT S/B 000000 AFTER MASTER CLEAR
3084
3085 003142 012502              1$:   MOV    (R5)+,R2    ;GET REG ADDRESS
3086 003144 022705 027330      CMP    #MSTCLR+10,R5 ;DONE ALL FOUR REGS ??
3087 003150 001415              BEQ    TST3           ;;BR IF YES
3088 003152 012712 177777      2$:   MOV    #-1,(R2)    ;SET 1'S IN REGISTER
3089 003156 052711 004000      BIS    #BIT11,(R1)   ;ISSUE MASTER CLEAR
3090 003162 011203              MOV    (R2),R3       ;GET CONTENT OF REGISTER
3091 003164 001766              BEQ    1$            ;;BR IF IT'S ALL ZEROES
3092
3093 003166 004737 024412      JSR    PC,SUER2      ;GO SET UP ERROR INFO
3094 003172 012737 003152 001110  MOV    #2$, $LPERR   ;SET UP ERROR LOOP RETURN
3095 003200 104003              ERROR  3             ;MASTER CLR FAILED TO CLR SEL. REG.
3096 003202 000757              BR     1$            ;GO TEST NEXT REGISTER
```

3097
3098
3099
3100 003204 000004
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152

: *TEST 3 TEST "SCR" REG R/W BITS CAN SET/CLR (NORMAL MODE)
: *****
TST3: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT EACH R/W BIT IN THE "SCR" REGISTER CAN BE INDIVIDUALLY SET AND CLEARED IN NORMAL MODE (MAINT BIT = 0) A BIT MASK (RGMSK1: 131177) IS USED TO DEFINE THE R/W BITS (ALL BUT BITS 14, 11, 10, 8, AND 7). THE TEST IS REPEATED ELEVEN TIMES WITH A DIFFERENT BIT SELECTED FOR EACH TEST. R5 CONTAINS THE BIT CURRENTLY BEING TESTED. IF AN ERROR IS DETECTED, IT IS REPORTED AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE UNTIL ALL HAVE BEEN TESTED.

ERRORS:

1.) ERROR 2 IS CALLED TO REPORT A FAILURE TO SET PROPERLY AND AGAIN TO REPORT A FAILURE TO CLEAR PROPERLY.

SYNC:

1.) SET FAILURE M7277 SH4 LOAD SSR LOW BYTE H CR1
2.) CLR FAILURE M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG:

1.) IF ALL BITS FAIL - SUSPECT THE "LOAD SCR" SIGNALS ON THE M7277 SH4.
2.) IF ONLY ONE OR TWO BITS FAIL - SUSPECT EITHER THE "SCR" REGISTER FLOPS ON THE M7289 SH6, THE BUS RECEIVERS ON THE M7278 SH3 AND SH4, OR THE MULTIPLEXORS AND BUS DRIVERS ON THE M7289 SH5-8.

KEY LOGIC:

M7278 SH3 BUFF DATA <15:08> H
SH4 BUFF DATA <07:00> H

M7277 SH4 LOAD SCR LOW BYTE H CT2
LOAD SCR HIGH BYTE H CP1
DATA TO BUS H EN2
DATA SOURCE <A,B,C> H DU1,DU2,DT2

M7289 SH5 BUF DATA TO BUS B H E05-12
BUS DATA <15:12> L
SH6 BUS DATA <11:08> L
SCR <15:00> H
SH7 BUF DATA TO BUS A H E05-8
BUS DATA <07:04> L
SH8 BUS DATA <03:00> L

%

TEST "SCR" REG R/W BITS CAN SET/CLR (NORMAL MODE)

```
3153 003206 012737 003236 001110      MOV    #4$, $LPERR      ;SET UP ERROR LOOP RETURN
3154 003214 010102                    MOV    R1,R2            ;GET REGISTER ADDRESS
3155 003216 012705 000001            MOV    #1,R5            ;SET UP TO START WITH BIT00
3156
3157 003222 030537 027662            1$:   BIT    R5,RGMSK1    ;SHALL WE TEST THIS BIT ?
3158 003226 001003                    BNE    4$              ;BR IF YES
3159 003230 006305                    2$:   ASL    R5          ;SHIFT TO TST NEXT BIT
3160 003232
3161 003232 001430                    BEQ    TST4            ;;<BR IF DONE ALL R/W BITS>
3162 003234 000772                    BR     1$              ;GO TEST NEXT BIT
3163
3164 003236 010504                    4$:   MOV    R5,R4        ;RESULT S/B IN R4
3165 003240 005012                    CLR    (R2)           ;INIT REG BEING TESTED
3166 003242 112761 000000 000016    MOVB   #0,SSR(R1)     ;SCOPE SYNC
3167 003250 010512                    MOV    R5,(R2)        ;SET THE BIT
3168 003252 011203                    MOV    (R2),R3        ;GET THE WAS DATA
3169 003254 020403                    CMP    R4,R3          ;RESULT = S/B DATA ??
3170 003256 001403                    BEQ    5$              ;BR IF YES
3171
3172 003260 004737 024412            JSR    PC,SUER2       ;GO SET UP ERROR INFO
3173 003264 104002                    ERROR  2              ;SELECTED BIT FAILED TO SET IN SCR
3174
3175 003266 005004                    5$:   CLR    R4          ;SET UP TO CLEAR THE BIT S/B=000000
3176 003270 112761 000000 000017    MOVB   #0,SSR+1(R1)  ;SCOPE SYNC
3177 003276 040512                    BIC   R5,(R2)        ;CLR THE SELECTED BIT
3178 003300 011203                    MOV    (R2),R3        ;GET THE WAS DATA
3179 003302 001403                    BEQ    6$              ;BR IF IT CLEARED
3180
3181 003304 004737 024412            JSR    PC,SUER2       ;GO SET UP THE ERROR INFO
3182 003310 104002                    ERROR  2              ;SELECTED BIT FAILED TO CLEAR IN SCR
3183 003312 000746                    6$:   BR     2$          ;GO SELECT NEXT BIT
```

3184
3185
3186
3187 003314 000004
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221 003316 010102
3222 003320 012705 000001
3223 003324 030537 027664
3224 003330 001003
3225 003332 006305
3226 003334 001420
3227 003336 000772
3228
3229 003340 005004
3230 003342 005012
3231 003344 112761 000000 000016
3232 003352 010512
3233 003354 011203
3234 003356 001765
3235
3236 003360 004737 024412
3237 003364 012737 003340 001110
3238 003372 104002
3239 003374 000756

: *TEST 4 TEST "SCR" REG. READ ONLY BITS (NORMAL MODE)
: *****

TST4: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE "SCR" REGISTER READ ONLY BITS CAN NOT BE SET OR CLEARED IN NORMAL MODE. A BIT MASK (RGMSK2: 046600) IS USED TO DEFINE THE READ ONLY BITSS (14,11,10,8, AND 7). THE TEST IS REPEATED FIVE TIMES, ONCE FOR EACH BIT TO BE TESTED, AND ANY ERRORS DETECTED ARE REPORTED. AFTER THE ERROR REPORT THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE UNTIL ALL BITS HAVE BEEN TESTED.

ERRORS:

1.) ERROR 2 IS CALLED TO REPORT ANY READ ONLY BIT THAT FAILED TO RESPOND PROPERLY.

SYNC:

M7277 SH4 LOAD SSR LOW BYTE H CR1

DEBUG:

SAME AS FOR TEST 03

KEY LOGIC:

SAME AS FOR TEST 03

%
MOV R1,R2 ;MAKE IT THE REG. ADDR ALSO
MOV #1,R5 ;INIT BIT TEST MARKER
1\$: BIT R5,RGMSK2 ;IS IT A READ ONLY BIT ??
BNE 3\$;BR IF IT IS - GO TEST IT
2\$: ASL R5 ;SHIFT BIT MARKER
BEQ TST5 ;;BR IF DONE ALL BITS
BR 1\$;GO TEST THIS BIT
3\$: CLR R4 ;RESULT S/B = 000000
CLR (R2) ;INIT REG BEING TESTED
MOVB #0,SSR(R1) ;SCOPE SYNC
MOV R5,(R2) ;ATTEMPT TO SET A READ ONLY BIT
MOV (R2),R3 ;GET THE WAS DATA
BEQ 2\$;BR IF THE BIT DIDN'T SET
JSR PC,SUER2 ;GO SET UP ERROR INFO
MOV #3\$,\$LPERR ;SET UP ERROR LOOP RETURN ADDR
ERROR 2 ;READ ONLY BIT SET IN "SCR"
BR 2\$;CONTINUE WITH NEXT BIT

3240
3241
3242
3243 003376 000004
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295

*TEST 5 TEST "SCR" REG. BITS THAT CAN BE SET/CLR IN MAINT. MODE

TST5: SCOPE
.REM 2
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE "SCR" REGISTER READ-ONLY BITS (14, 10, AND 07) CAN BE SET/CLR IN MAINT. MODE (SCRO9=1) ONLY. A BIT MASK (RGMSK6: 042200) IS USED TO DEFINE THE BITS TO TEST. THE TEST PERFORMS THE FOLLOWING SEQUENCE:

1. SELECT A BIT TO TEST
2. SET MAINT. MODE
3. SET THE BIT AND VERIFY THAT IT SET
4. CLEAR THE MAINT. MODE BIT
5. ATTEMPT TO CLEAR THE TEST BIT
6. TEST TO SEE THAT IT DID NOT CLEAR
7. SET MAINT. MODE
8. CLEAR THE TEST BIT AND VERIFY THAT IT CLEARED
9. REPEAT 1-7 UNTIL ALL BITS HAVE BEEN TESTED

ANY ERRORS DETECTED ARE REPORTED AND THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE UNTIL ALL BITS HAVE BEEN TESTED.

ERRORS:

- 1.) ERROR 2 IS CALLED AT THREE DIFFERENT POINTS TO REPORT ONE OF THE THREE POSSIBLE FAILURE MODES DESCRIBED IN 3, 6, AND 7 IN THE ABSTRACT.

SYNC:

- 1.) STEP 3 FAILURE TO SET WITH MAINT. MODE SET

M7277 SH4 LOAD SSR LOW BYTE H CR1

- 2.) STEP 5 FAILURE TO REMAIN SET WITH MAINT MODE NOT SET

M7277 SH4 LOAD SSR HIGH BYTE H CP2

- 3.) STEP 8 FAILURE TO CLEAR WITH MAINT. MODE SET

M7277 SH4 LOAD LPR H EP2

DEBUG:

- 1.) ASSUMING THE PREVIOUS TESTS RAN SUCCESSFULLY THE FAULT IS MOST LIKELY THE M7289 MODULE.

KEY LOGIC:

```

3296
3297
3298
3299
3300
3301 003400 012737 003430 001110
3302 003406 010102
3303 003410 012705 000001
3304 003414 030537 027674
3305 003420 001003
3306 003422 006305
3307 003424 001457
3308 003426 000772
3309
3310 003430 010504
3311 003432 052704 001000
3312 003436 005012
3313 003440 052712 001000
3314 003444 112761 000000 000016
3315 003452 050512
3316 003454 011203
3317 003456 020304
3318 003460 001404
3319
3320 003462 004737 024412
3321 003466 104002
3322 003470 000754
3323
3324 003472 042712 001000
3325 003476 042704 001000
3326 003502 112761 000000 000017
3327 003510 040512
3328 003512 011203
3329 003514 020304
3330 003516 001404
3331
3332 003520 004737 024412
3333 003524 104002
3334 003526 000735
3335
3336 003530 012704 001000
3337 003534 050412
3338 003536 012761 000000 000004
3339 003544 040512
3340 003546 011203
3341 003550
3342 003552
3343
3344 003554 004737 024412
3345 003560 104002
3346 003562 000717

```

SAME AS TEST 03 WITH THE FOLLOWING ADDITION

```

M7289 SH4 74121 ONE-SHOTS E35-6, E23-6
X
MOV #3$, $LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;MAKE IT REG ADDR TOO
MOV #1,R5 ;INIT BIT TEST MARKER
1$: BIT R5,RGMSK6 ;IS IT A READ ONLY BIT ??
BNE 3$ ;BR IF YES - TEST IT
2$: ASL R5 ;SHIFT THE BIT MARKER
BEQ TST6 ;;BR IF DONE ALL SELECTED BITS
BR 1$ ;GO TEST FOR THIS BIT

3$: MOV R5,R4 ;SET UP S/B DATA
BIS #BIT09,R4 ;PUT IN THE MAINT. BIT
CLR (R2) ;INIT REG BEING TESTED
BIS #BIT09,(R2) ;TURN ON MAINT. MODE
MOVB #0,SSR(R1) ;SCOPE SYNC
BIS R5,(R2) ;SET THE SELECTED BIT
MOV (R2),R3 ;GET THE WAS DATA
CMP R3,R4 ;DID SELECTED BIT GET SET ??
BEQ 4$ ;BR IF IT DID

JSR PC,SUER2 ;GO SET UP ERROR INFO
ERROR 2 ;SELECTED BIT FAILED TO SET IN MAINT MODE
BR 2$ ;GO TEST NEXT BIT

4$: BIC #BIT09,(R2) ;TURN OFF MAINT. MODE
BIC #BIT09,R4 ;CLR MAINT BIT IN S/B DATA
MOVB #0,SSR+1(R1) ;SCOPE SYNC
BIC R5,(R2) ;ATTEMPT TO CLR SELECTED BIT
MOV (R2),R3 ;GET THE WAS DATA
CMP R3,R4 ;DID BIT GET CLEARED ??
BEQ 5$ ;BR IF IT DIDN'T

JSR PC,SUER2 ;GO SET UP ERROR INFO
ERROR 2 ;SELECTED BIT GOT CLEARED WITH MAINT MODE OFF
BR 2$ ;GO TEST NEXT BIT

5$: MOV #BIT09,R4 ;SET UP S/B DATA
BIS R4,(R2) ;SET MAINT. MODE
MOV #0,LPR(R1) ;SCOPE SYNC
BIC R5,(R2) ;NOW CLR SELECTED BIT
MOV (R2),R3 ;GET THE WAS DATA
CMP R3,R4 ;DID BIT GET CLEARED OK ??
BEQ 2$ ;BR IF YES

JSR PC,SUER2 ;GO SET UP ERROR INFO
ERROR 2 ;FAILED TO CLR SELECTED BIT IN MAINT MODE
BR 2$ ;GO SELECT NEXT BIT FOR TEST

```


3347
3348
3349
3350 003564 000004

: *TEST 6 TEST THAT ALL R/W BITS IN 'LPR' CAN BE SET/CLR
: *****
TST6: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT ALL R/W BITS IN THE 'LPR' REGISTER CAN BE SET AND CLEARED INDIVIDUALLY. A BIT MASK (RGMSK3: 177767) IS USED TO DEFINE THE BITS TO BE TESTED (ALL BUT BIT03). THE TEST SEQUENCE IS AS FOLLOWS:

- 1. SELECT A BIT TO TEST
- 2. SET THE BIT AND VERIFY IT SET
- 3. CLEAR THE BIT AND VERIFY IT CLEARED
- 4. REPEAT 1 THRU 3 UNTIL ALL BITS TESTED

ANY ERRORS DETECTED ARE REPORTED AND AFTER THE ERROR, THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE.

ERRORS:

- 1.) ERROR 4 IS CALLED TO REPORT BOTH FAIL TO SET AND FAIL TO CLEAR FAULTS.

SYNC:

- 1.) FAIL TO SET: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) FAIL TO CLEAR: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG:

- 1.) IF ALL BITS FAIL THE PROBLEM IS MOST LIKELY THE M7277 MODULE (LPR LOAD SIGNALS)
- 2.) IF NOT THEN IT IS PROBABLY AN 'LPR' REGISTER CHIP OR BAD OUTPUT DATA MUX CHIP, BOTH ON THE M7278 MODULE.

KEY LOGIC:

M7277 SH4 LOAD LPR H EP2
M7278 SH5 LPR <15:12> L (E52)
SH6 LPR <11:08> L (E37)
SH7 LPR <07:04> L (E59)
SH8 LPR <03:00> L (E61)
SH5,6,7,8 OUTPUT MUX CHIPS (74151'S PIN 2)

3399 %
3400 003566 012737 003622 001110
3401 003574 010102
3402 003576 062702 000004

MOV #38,\$LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;COPY IT IN R2
ADD #LPR,R2 ;GENERATE REGADR IN R2

CZDMM-D-0
CZDHMD.P11MACY11 30A(1052)
09-MAR-78 15:3210-MAR-78
T6

08:05 PAGE 83

TEST THAT ALL R/W BITS IN "LPR" CAN BE SET/CLR

SEQ 0081
SEQ 0080

3403	003602	012705	000001		MOV	#1,R5		;INIT BIT TEST MARKER
3404	003606	030537	027666	1\$:	BIT	R5,RGMSK3		;TEST THIS BIT ??
3405	003612	001003			BNE	3\$;BR IF YES
3406	003614	006305		2\$:	ASL	R5		;SHIFT THE MARKER
3407	003616	001430			BEQ	TST7		;BR IF DONE ALL BITS
3408	003620	000772			BR	1\$;GO TEST NXT BIT
3409								
3410	003622	010504		3\$:	MOV	R5,R4		;SET UP S/B DATA
3411	003624	005012			CLR	(R2)		;INIT REG BEING TESTED
3412	003626	112761	000000	000016	MOVB	#0,SSR(R1)		;SCOPE SYNC
3413	003634	010512			MOV	R5,(R2)		;SET LPR BIT
3414	003636	011203			MOV	(R2),R3		;GET THE WAS DATA
3415	003640	020304			CMP	R3,R4		;DID IT SET
3416	003642	001403			BEQ	4\$;BR IF IT SET PROPERLY
3417								
3418	003644	004737	024412		JSR	PC,SUER2		;GO SET UP ERROR INFO
3419	003650	104004			ERROR	4		;LPR BIT FAILED TO SET PROPERLY
3420								
3421	003652	005004		4\$:	CLR	R4		;GET READY TO CLEAR SELECTED BIT
3422	003654	112761	000000	000017	MOVB	#0,SSR+1(R1)		;SCOPE SYNC
3423	003662	040512			BIC	R5,(R2)		;CLEAR THE BIT
3424	003664	011203			MOV	(R2),R3		;GET THE WAS DATA
3425	003666	001752			BEQ	2\$;BR IF BIT CLEARED PROPERLY
3426								
3427	003670	004737	024412		JSR	PC,SUER2		;GO SET UP ERROR INFO
3428	003674	104004			ERROR	4		;LPR BIT FAILED TO CLEAR PROPERLY
3429	003676	000746			BR	2\$;GO SELECT NEXT BIT

3430
3431
3432
3433 003700 000004

;*TEST 7 TEST THAT ALL R/W BITS IN 'BKR' CAN BE SET/CLR

TST7: SCOPE
.REM %
TEST ABSTRACT:

3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483

THIS TEST VERIFIES THAT ALL BITS IN THE BREAK CONTROL REGISTER CAN BE SET AND CLEARED INDIVIDUALLY. IT USES A BIT MASK (RGMSK4: 177777) TO DEFINE THE R/W BITS (ALL 16.). R5 ALWAYS CONTAINS THE BIT CURRENTLY SELECTED FOR TEST. THE TEST SEQUENCE IS AS FOLLOWS:

- 1. SELECT A BIT TO TEST
- 2. SET THE BIT AND VERIFY THAT IT SET PROPERLY
- 3. CLEAR THE BIT AND VERIFY THAT IT CLEARED PROPERLY
- 4. REPEAT 1 THRU 4 UNTIL ALL BITS HAVE BEEN TESTED.

ANY ERROR DETECTED IS REPORTED AND THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE.

ERRORS:

- 1.) ERROR 5 IS CALLED TO REPORT BOTH FAIL TO SET PROPERLY AND FAIL TO CLEAR PROPERLY FAULTS.

SYNC:

- 1.) FAIL TO SET: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) FAIL .J CLR: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG:

- 1.) THE ONLY DIFFERENCES IN THE DATA PATH HERE AND THAT FOT THE PREVIOUS TESTS ARE THE ACTUAL REGISTER CHIPS AND THE INPUT SELECTED ON THE OUTPUT DATA MULTIPLEXORS.
- 2.) IF ALL BITS FAIL THE PROBLEM IS MOST LIKELY THE M7277.
- 3.) IF ONLY ONE OR TWO FAIL THE PROBLEM IS MOST LIKELY THE M7278.

KEY LOGIC:

M7277 SH4 LOAD BCR H FU1
DATA TO BUS H EN2
DATA SOURCE (A,B,C) H DU1,DU2,DT2

M7278 SH5 - SH8 74175 REGISTER CHIPS (E51,E38,E67,E60)
SH5 - SH8 74151'S MUX CHIPS INPUT PIN 13

3484 003702 012737 003736 001110
3485 003710 010102

X
MOV #3\$, \$LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;GENERATE 'BKR' ADDRESS IN R2

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
T7

08:05 PAGE 85

TEST THAT ALL R/W BITS IN 'BKR' CAN BE SET/CLR

SEQ 0083
SEQ 0082

3486	003712	062702	000014		ADD	#BKR,R2	
3487	003716	012705	000001		MOV	#1,R5	;INIT BIT TEST MARKER
3488	003722	030537	027670	1\$:	BIT	R5,RGMSK4	;TEST THIS BIT ??
3489	003726	001003			BNE	3\$;BR IF YES
3490	003730	006305		2\$:	ASL	R5	;SHIFT BIT MARKER
3491	003732	001430			BEQ	TST10	;BR IF ALL BITS TESTED
3492	003734	000772			BR	1\$;GO TEST THE BIT
3493							
3494	003736	010504		3\$:	MOV	R5,R4	;SET UP S/B DATA
3495	003740	005012			CLR	(R2)	;INIT REG BEING TESTED
3496	003742	112761	000000 000016		MOVB	#0,SSR(R1)	;SCOPE SYNC
3497	003750	050512			BIS	R5,(R2)	;SET THE SELECTED BIT IN 'BKR'
3498	003752	011203			MOV	(R2),R3	;GET THE WAS DATA
3499	003754	020304			CMP	R3,R4	;DID BIT SET OK
3500	003756	001403			BEQ	4\$;BR IF YES
3501							
3502	003760	004737	024412		JSR	PC,SUER2	;GO SET UP ERROR INFO
3503	003764	104005			ERROR	5	;BKR BIT FAILED TO SET PROPERLY
3504							
3505	003766	005004		4\$:	CLR	R4	;SET UP S/B DATA
3506	003770	112761	000000 000017		MOVB	#0,SSR+1(R1)	;SCOPE SYNC
3507	003776	040512			BIC	R5,(R2)	;CLEAR BKR BIT
3508	004000	011203			MOV	(R2),R3	;GET THE BKR WAS DATA
3509	004002	001752			BEQ	2\$;BR IF BKR BIT CLEARED OK
3510							
3511	004004	004737	024412		JSR	PC,SUER2	;GO SET UP ERROR INFO
3512	004010	104005			ERROR	5	;BKR BIT FAILED TO CLR PROPERLY
3513	004012	000746			BR	2\$;GO SELECT NEXT BIT

3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569

004014 000004

:TEST 10 TEST THAT ALL R/W BITS IN "SSR" CAN BE SET/CLR

TST10: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT ALL R/W BITS IN THE SILO STATUS REGISTER (SSR) CAN BE SET AND CLEARED INDIVIDUALLY. IT USES A BIT MASK (RGMSK5: 100077) TO DEFINE THE R/W BITS (15,5,4,3,2,1, AND 0). R5 ALWAYS CONTAINS THE BIT CURRENTLY SELECTED FOR TEST. THE TEST SEQUENCE IS AS FOLLOWS:

- 1. SELECT A BIT TO TEST
- 2. SET THE BIT AND VERIFY THAT IT SET PROPERLY
- 3. CLEAR THE BIT AND VERIFY THAT IT CLEARED PROPERLY
- 4. REPEAT 1 THRU 3 UNTIL ALL BITS ARE TESTED

ANY ERRORS DETECTED ARE REPORTED AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE.

ERRORS:

- 1.) ERROR 6 IS CALLED TO REPOORT BOTH FAIL TC SET PROPERLY AND FAIL TO CLEAR PROPERLY FAULTS.

SYNC:

- 1.) FAIL TO SET: M7277 SH4 LOAD LPR H EP2
- 2.) FAIL TO CLR: M7277 SH4 LOAD BCR H FU1

DEBUG:

- 1.) THE ONLY DIFFERENCES BETWEEN THEE DATA PATHS USED BY THIS TEST AND THAT USED BY THE PREVIOUS TESTS ARE THE ACTUAL "SSR" REGISTER CHIPS AND THE INPUT PIN SELECTED ON THE OUTPUT DATA MULTIPLEXORS.
- 2.) IF ALL BITS FAIL IT IS MOST LIKELY THE M7277
- 3.) IF BITS <13:08> FAIL IT IS MOST LIKELY THE M7279
- 4.) IF JUST ONE OR TWO BITS FAIL IT IS MOST LIKELY THE M7278

KEY LOGIC:

M7277 SH4 LOAD SSR LOW BYTE H CR1
LOAD SSR HIGH BYTE H CP2
DATA TO BUS H EN2
DATA SOURCE (A,B,C) H DU1,DU2,DT2

M7279 SH2 SSR <13:08> H (E20 AND E24)

M7278 SH5 - SH8 REGISTER CHIPS E53,E68, OR E69 (74175'S)
OUTPUT MUX CHIPS - (74151'S PIN 12)

X

```
3570 004016 012737 004052 001110      MOV    #3$, $LPERR      ;SET UP THE ERROR LOOP RETURN
3571 004024 010102                    MOV    R1,R2            ;GENERATE "SSR" ADDRESS IN R2
3572 004026 062702 000016              ADD    #SSR,R2
3573 004032 012705 000001              MOV    #1,R5            ;INIT BIT TEST MARKER
3574 004036 030537 027672              1$:   BIT    R5,RGMSK5     ;TEST THIS BIT ??
3575 004042 001003                    BNE    3$               ;BR IF YES
3576 004044 006305              2$:   ASL    R5            ;SHIFT BIT MARKER
3577 004046 001435                    BEQ    TST11            ;:BR IF ALL BITS TESTED
3578 004050 000772                    BR     1$               ;GO TEST THE BIT
3579
3580 004052 010504              3$:   MOV    R5,R4            ;SET UP S/B DATA
3581 004054 005012                    CLR    (R2)            ;INIT REG BEING TESTED
3582 004056 012761 000000 000004      MOV    #0,LPR(R1)      ;SCOPE SYNC
3583 004064 050512                    BIS    R5,(R2)         ;SET THE SELECTED BIT IN "SSR"
3584 004066 011203                    MOV    (R2),R3         ;GET THE WAS DATA
3585 004070 02703 077700              BIC    #77700,R3       ;CLEAR OUT DON'T CARE BITS
3586 004074 020304                    CMP    R3,R4            ;DID BIT SET OK
3587 004076 001403                    BEQ    4$               ;BR IF YES
3588
3589 004100 004737 024412              JSR    PC,SUER2        ;GO SET UP ERROR INFO
3590 004104 104006                    ERROR  6                ;SSR BIT FAILED TO SET PROPERLY
3591
3592 004106 005004              4$:   CLR    R4            ;SET UP S/B DATA
3593 004110 012761 000000 000014      MOV    #0,BKR(R1)     ;SCOPE SYNC
3594 004116 040512                    BIC    R5,(R2)         ;CLEAR SSR BIT
3595 004120 011203                    MOV    (R2),R3         ;GET THE SSR WAS DATA
3596 004122 042703 077700              BIC    #77700,R3       ;CLEAR JUNK BITS
3597 004126 020304                    CMP    R3,R4            ;DID THE SSR BIT GET CLEARED ??
3598 004130 001745                    BEQ    2$               ;BR IF SSR BIT CLEARED OK
3599
3600 004132 004737 024412              JSR    PC,SUER2        ;GO SET UP ERROR INFO
3601 004136 104006                    ERROR  6                ;SSR BIT FAILED TO CLR PROPERLY
3602 004140 000741                    BR     2$               ;GO SELECT NEXT BIT
```

3603
3604
3605
3606 004142 000004
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647 004144 012737 004200 001110
3648 004152 010102
3649 004154 062702 000004
3650 004160 012705 000001
3651 004164 030537 027666
3652 004170 001003
3653 004172 006305
3654 004174 001436
3655 004176 000772
3656
3657 004200 013704 027666
3658 004204 005012

```
*****  
;*TEST 11 TEST THAT CLR/SET OF BIT 'N' IN 'LPR' DOES NOT CLEAR ANY OTHER BITS  
*****  
TST11: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT SETTING AND CLEARING EACH R/W BIT IN THE 'LPR' REGISTER DOES NOT DISTURB (CLEAR) ANY OTHER BIT IN THE REGISTER. A BIT MASK (RGMSK3: 177767) IS USED TO DEFINE THE R/W BITS (ALL BUT BIT 03). R5 ALWAYS CONTAINS THE BIT CURRENTLY SELECTED FOR TEST. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A BIT TO TEST
2. SET ALL THE WRITABLE BITS
3. CLEAR THE SELECTED BIT - VERIFY IT CLEARED PROPERLY
4. SET THE SELECTED BIT - VERIFY IT SET PROPERLY
5. REPEAT 1 THRU 4 UNTIL ALL BITS ARE TESTED

ANY ERRORS DETECTED ARE REPORTED AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE .

ERRORS:

- 1.) ERROR 4 IS CALLED TO REPORT BOTH FAIL TO CLEAR PROPERLY AND FAIL TO SET PROPERLY FAULTS.

SYNC:

- 1.) FAIL TO CLR: M7277 LOAD SSR LOW BYTE H CR1
- 2.) FAIL TO SET: M7277 LOAD SSR HIGH BYTE H CP2

DEBUG:

- 1.) PROBLEMS DETECTED BY THIS TEST INDICATE ADJACENT BIT INTERFERENCE CAUSED BY CROSS TALK OR NOISE. PROBLEM IS MOST LIKELY THE M7278.

KEY LOGIC: (SAME AS FOR TEST 6)

%

```
MOV #3$, $LPERR ;SET UP THE ERROR LOOP RETURN  
MOV R1,R2 ;SET UP THE REG ADDR  
ADD #LPR,R2  
MOV #1,R5 ;INIT BIT TEST MASK  
1$: BIT R5, RGMSK3 ;TEST THIS BIT ??  
BNE 3$ ;BR IF YES  
2$: ASL R5 ;SHIFT THE BIT TEST MASK  
BEQ TST12 ;;BR IF TESTED ALL BITS  
BR 1$ ;GO TEST THIS BIT  
3$: MOV RGMSK3,R4 ;SET UP S/B DATA  
CLR (R2) ;INIT REG BEING TESTED
```

```
3659 004206 112761 000000 000016      MOVB    #0,SSR(R1)      ;SCOPE SYNC
3660 004214 040504                      BIC     R5,R4           ;CLR BIT 'N'
3661 004216 013712 027666              MOV     RGMSK3,(R2)    ;SET ALL R/W BITS IN LPR
3662 004222 040512                      BIC     R5,(R2)        ;CLEAR BIT 'N' IN LPR
3663 004224 011203                      MOV     (R2),R3        ;GET THE WAS DATA
3664 004226 020304                      CMP     R3,R4          ;DID IT CLEAR OK ?
3665 004230 001404                      BEQ     4$             ;BR IF YES
3666
3667 004232 004737 024412              JSR     PC,SUER2       ;GO SET UP ERROR INFO
3668 004236 104004                      ERROR   4              ;BIT 'N' FAILED TO CLR PROPERLY
3669 004240 000754                      BR      2$             ;GO TEST NEXT BIT
3670
3671 004242 050504                      4$:  BIS     R5,R4          ;SET BIT 'N' IN S/B DATA
3672 004244 112761 000000 000017      MOVB    #0,SSR+1(R1)  ;SCOPE SYNC
3673 004252 050512                      BIS     R5,(R2)        ;SET BIT 'N' IN LPR
3674 004254 -011203                      MOV     (R2),R3        ;GET THE WAS DATA
3675 004256 020304                      CMP     R3,R4          ;DID BIT 'N' SET PROPERLY ?
3676 004260 001744                      BEQ     2$             ;BR IF YES
3677
3678 004262 004737 024412              JSR     PC,SUER2       ;GO SET UP ERROR INFO
3679 004266 104004                      ERROR   4              ;BIT 'N' FAILED TO SET PROPERLY
3680 004270 000740                      BR      2$             ;GO SELECT NEXT BIT
```


3681
3682
3683
3684 004272 000004
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725 004274 012737 004330 001110
3726 004302 010102
3727 004304 062702 000014
3728 004310 012705 000001
3729 004314 030537 027670
3730 004320 001003
3731 004322 006305
3732 004324 001436
3733 004326 000772
3734
3735 004330 013704 027670
3736 004334 040504

```
*****  
*TEST 12 TEST THAT CLR/SET OF BIT 'N' IN 'BKR' DOES NOT CLEAR ANY OTHER BITS  
*****  
TST12: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT CLEARING AND SETTING EACH R/W BIT IN THE BREAK CONTROL REGISTER INDIVIDUALLY DOES NOT DISTURB ANY OF THE OTHER BITS. A BIT MASK (RGMSK4: 17777) IS USED TO DEFINE THE R/W BITS (ALL 16.). R5 ALWAYS CONTAINS THE BIT CURRENTLY SELECTED FOR TEST. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A BIT TO TEST
2. SET ALL WRITABLE BITS IN THE 'BKR'
3. CLEAR THE SELECTED BIT AND VERIFY THAT IT CLEARED PROPERLY
4. SET THE SELECTED BIT AND VERIFY THAT IT SET PROPERLY
5. REPEAT 1 THRU 4 UNTIL ALL BITS HAVE BEEN TESTED

ANY ERROR DETECTED IS REORTRD AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE.

ERRORS:

- 1.) ERROR 5 IS CALLED TO REPORT BOTH CLEAR AND SET FAULTS.

SYNC:

- 1.) FAIL TO CLR: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) FAIL TO SET: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG:

- 1.) LIKE THE PREVIOUS TEST, FAILURES HERE INDICATE ADJACENT BIT INTERFERENCE CAUSED BY CROSS TALK OR NOISE. THE FAULT IS MOST LIKELY THE M7278 MODULE.

KEY LOGIC: (SAME AS FOR TEST 7)

```
%  
MOV #3$, $LPERR ;SET UP THE ERROR LOOP RETURN  
MOV R1,R2 ;SET UP THE REG ADDR  
ADD #BKR,R2  
MOV #1,R5 ;INIT BIT TEST MASK  
1$: BIT R5, RGMSK4 ;TEST THIS BIT ??  
BNE 3$ ;BR IF YES  
2$: ASL R5 ;SHIFT THE BIT TEST MASK  
BEQ TST13 ;;BR IF TESTED ALL BITS  
BR 1$ ;GO TEST THIS BIT  
3$: MOV RGMSK4,R4 ;SET UP S/B DATA  
BIC R5,R4 ;CLR BIT 'N'
```

TEST THAT CLR/SET OF BIT 'N' IN 'BKR' DOES NOT CLEAR ANY OTHER BITS

```
3737 004336 005012 CLR (R2) ;INIT REG BEING TESTED
3738 004340 013712 027670 MOV RGMSK4,(R2) ;SET ALL R/W BITS IN BKR
3739 004344 112761 000000 000016 MOVB #0,SSR(R1) ;SCOPE SYNC
3740 004352 040512 BIC R5,(R2) ;CLEAR BIT 'N' IN BKR
3741 004354 011203 MOV (R2),R3 ;GET THE WAS DATA
3742 004356 020304 CMP R3,R4 ;DID IT CLEAR OK ?
3743 004360 001404 BEQ 4$ ;BR IF YES
3744
3745 004362 004737 024412 JSR PC,SUER2 ;GO SET UP ERROR INFO
3746 004366 104005 ERROR 5 ;BIT 'N' FAILED TO CLR PROPERLY
3747 004370 000754 BR 2$ ;GO TEST NEXT BIT
3748
3749 004372 050504 4$: BIS R5,R4 ;SET BIT 'N' IN S/B DATA
3750 004374 112761 000000 000017 MOVB #0,SSR+1(R1) ;SCOPE SYNC
3751 004402 050512 BIS R5,(R2) ;SET BIT 'N' IN BKR
3752 004404 011203 MOV (R2),R3 ;GET THE WAS DATA
3753 004406 020304 CMP R3,R4 ;DID BIT 'N' SET PROPERLY ?
3754 004410 001744 BEQ 2$ ;BR IF YES
3755
3756 004412 004737 024412 JSR PC,SUER2 ;GO SET UP ERROR INFO
3757 004416 104005 ERROR 5 ;BIT 'N' FAILED TO SET PROPERLY
3758 004420 000740 BR 2$ ;GO SELECT NEXT BIT
```

3759
3760
3761
3762 004422 C 0004
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803 004424 012737 004460 001110
3804 004432 010102
3805 004434 062702 000016
3806 004440 012705 000001
3807 004444 030537 027672
3808 004450 001003
3809 004452 006305
3810 004454 001442
3811 004456 000772
3812
3813 004460 013704 027672
3814 004464 040504

```
::*****  
:*TEST 13 TEST THAT CLR/SET OF BIT 'N' IN 'SSR' DOES NOT CLEAR ANY OTHER BITS  
:*****  
TST13: SCOPE  
.REM X  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT CLEARING AND SETTING EACH R/W BIT IN THE SILO STATUS REGISTER INDIVIDUALLY DOES NOT DISTURB ANY OF THE OTHER BITS. A BIT MASK (RGMSK5: 100077) IS USED TO DEFINE THE R/W BITS (15,5,4,3,2,1, AND 0). R5 ALWAYS CONTAINS THE BIT CURRENTLY SELECTED FOR TEST. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A BIT TO TEST
2. SET ALL WRITABLE BITS IN THE 'SSR'
3. CLEAR THE SELECTED BIT AND VERIFY THAT IT CLEARED PROPERLY
4. SET THE SELECTED BIT AND VERIFY THAT IT SET PROPERLY
5. REPEAT 1 THRU 4 UNTIL ALL BITS HAVE BEEN TESTED

ANY ERROR DETECTED IS REORTRD AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE.

ERRORS:

- 1.) ERROR 6 IS CALLED TO REPORT BOTH CLEAR AND SET FAULTS.

SYNC:

- 1.) FAIL TO CLR: M7277 SH4 LOAD LPR H EP2
- 2.) FAIL TO SET: M7277 SH4 LOAD BCR H FU1

DEBUG:

- 1.) LIKE THE PREVIOUS TEST, FAILURES HERE INDICATE ADJACENT BIT INTERFERENCE CAUSED BY CROSS TALK OR NOISE. THE FAULT IS MOST LIKELY THE M7278 MODULE.

KEY LOGIC: (SAME AS FOR TEST 10)

```
X  
MOV #3$, $LPERR ;SET UP THE ERROR LOOP RETURN  
MOV R1,R2 ;SET UP THE REG ADDR  
ADD #SSR,R2  
MOV #1,R5 ;INIT BIT TEST MASK  
1$: BIT R5, RGMSK5 ;TEST THIS BIT ??  
BNE 3$ ;BR IF YES  
2$: ASL R5 ;SHIFT THE BIT TEST MASK  
BEQ TST14 ;;BR IF TESTED ALL BITS  
BR 1$ ;GO TEST THIS BIT  
3$: MOV RGMSK5,R4 ;SET UP S/B DATA  
BIC R5,R4 ;CLR BIT 'N'
```

3815	004466	005012			CLR	(R2)	: INIT REG BEING TESTED
3816	004470	013712	027672		MOV	RGMSK5,(R2)	: SET ALL R/W BITS IN SSR
3817	004474	012761	000000	000004	MOV	#0,LPR(R1)	: SCOPE SYNC
3818	004502	040512			BIC	R5,(R2)	: CLEAR BIT 'N' IN SSR
3819	004504	011203			MOV	(R2),R3	: GET THE WAS DATA
3820	004506	042703	077700		BIC	#77700,R3	: CLEAR JUNK BITS
3821	004512	020304			CMP	R3,R4	: DID IT CLEAR OK ?
3822	004514	001404			BEQ	4\$: BR IF YES
3823							
3824	004516	004737	024412		JSR	PC,SUER2	: GO SET UP ERROR INFO
3825	004522	104006			ERROR	6	: BIT 'N' FAILED TO CLR PROPERLY
3826	004524	000752			BR	2\$: GO TEST NEXT BIT
3827							
3828	004526	050504			BIS	R5,R4	: SET BIT 'N' IN S/B DATA
3829	004530	012761	000000	000014	MOV	#0,BKR(R1)	: SCOPE SYNC
3830	004536	050512			BIS	R5,(R2)	: SET BIT 'N' IN SSR
3831	004540	011203			MOV	(R2),R3	: GET THE WAS DATA
3832	004542	042703	077700		BIC	#77700,R3	: CLEAR JUNK BITS
3833	004546	020304			CMP	R3,R4	: DID BIT 'N' SET PROPERLY ?
3834	004550	001740			BEQ	2\$: BR IF YES
3835							
3836	004552	004737	024412		JSR	PC,SUER2	: GO SET UP ERROR INFO
3837	004556	104006			ERROR	6	: BIT 'N' FAILED TO SET PROPERLY
3838	004560	000734			BR	2\$: GO SELECT NEXT BIT

3839
3840
3841
3842 004562 000004
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894

: *TEST 14 "CAR" MEMORY ADDRESSING TEST
: *****
TST14: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT EACH LOCATION IN THE CURRENT ADDRESS MEMORY CAN BE UNIQUELY ADDRESSED. IT WRITES THE PATTERN SHOWN BELOW INTO THE MEMORY AND THEN READS BACK EACH LOCATION TO VERIFY THAT IT WAS WRITTEN CORRECTLY. SINCE THE MEMORY LOGIC IS PARTITIONED INTO FOUR 16 X 4 READ/WRITE MEMORY CHIPS, THE PATTERN RESULTS IN THE LINE NUMBER (00 - 17(8)) BEING WRITTEN AS DATA INTO TO EACH LOCATION (00 - 17(8)) IN EACH CHIP. THAT IS: LOC00 = 00, LOC 01 = 01,LOC 17 = 17.

MEMORY PATTERN:	LOCATION	CONTENTS	(BOTH OCTAL)
	00	000000	
	01	010421	
	02	021042	
	03	031463	
	04	042104	
	05	052525	
	06	063146	
	07	073567	
	10	104210	
	11	114631	
	12	125252	
	13	135673	
	14	146314	
	15	156735	
	16	167356	
	17	177777	

ANY ERRORS DETECTED ARE REPORTED AND THEN THE TEST RESUMES CHECKING THEE NEXT LOCATION IN SEQUENCE UNTIL ALL 16. HAVE BEEN CHECKED.

NOTE: THIS TEST ALWAYS CHECKS ALL 16. LINES REGARDLESS OF HOW THE LINE SELECTION PARAMETER WAS INITIALLY SET UP.

ERRORS:

1.) ERROR 7 IS CALLED TO REPORT ANY LINES (LOCATIONS) THAT FAIL. THE FAILING LINE # IS INCLUDED AS PART OF THE ERROR HEADER MESSAGE.

SYNC:

1.) WRITE SYNC: M7277 SH4 LOAD SSR LOW BYTE H CR1
2.) READ SYNC: M7277 SH4 LOAD SSR HIGH BYTE H CP2

3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950

DEBUG:

- 1.) ANALYZE THE ERROR REPORTS CAREFULLY ASKING THE FOLLOWING QUESTIONS:
 - A. DOES THE FAULT AFFECT ONLY ONE LINE ?
 - B. DOES THE FAULT AFFECT ONLY ONE 4-BIT DATA GROUP ?
IE <15:12>, <11:08>, <07:04>, OR <03:00>
 - C. DOES ANY DATA AT ALL APPEAR TO BE WRITTEN ?
- 2.) IF "A" IS TRUE THEN SUSPECT AN ADDRESSING PROBLEM IN THE MEMORY ADDRESS MUX.
- 3.) IF "B" IS TRUE THEN SUSPECT A DATA MUX, UP-COUNTER, MEMORY, OR INVERTER CHIP PROBLEM.
- 4.) IF "C" IS TRUE SUSPECT A MEMORY WRITE TIMING PROBLEM.
- 5.) IN MOST CASES THE FAULT IS MOST LIKELY THE M7277 OR M7278.

KEY LOGIC:

M7277	SH4	LOAD CA H	E58-13
		DATA TO BUS H	EN2
		DATA SOURCE (A,B,C)	DU1,DU2,DT2
	SH5	MEMADD SOURCE SEL H	E55-8
		CA MEM WRITE ENAB L	E50-1
		BUF ADDRS TO BUS H	E33-1 (SHD BE LOW)
		74157 MUX CHIPS E33,E27,E20 BITS<17:08>	
		74193 COUNTER CHIPS E19,E26,E32 BITS<17:08>	
		7489 MEMORY CHIPS E18,E25,E31 BITS<17:08>	
		7404 INVERTER CHIPS E30,E24,E17 BITS<17:08>	
	SH5	74157 MUX CHIP E48	
		74157 DATA MUX CHIPS E13,E06 BITS<07:00>	
		74193 COUNTER CHIPS E12,E05 BITS<07:00>	
		7489 MEMORY CHIPS E11,E04 BITS<07:00>	
M7278	SH5 THRU SH8	74151 DATA MUX OUTPUT CHIPS (PIN 1 INPUT)	
		MOV R1,R2	:COPY IT IN R2
		ADD #CAR,R2	:SET UP REGADR IN R2
		MOV LINSSEL,\$TMP7	:SAVE LINE SELECT PARAMETER
		MOV #-1,LINSSEL	:DO ALL LINES FOR THIS TEST
	1\$:	JSR PC,SELINE	:GO SELECT A LINE NO.
		BR 3\$::BR IF DONE ALL SELECTED LINES
		TSTB LINE	:DOING LINE 00 ?
		BNE 2\$:BR IF NOT
		CLR R4	:INIT TEST DATA
	2\$:	BISB LINE,(R1)	:SELECT A LINE
		MOVB #0,SSR(R1)	:SCOPE SYNC
		MOV R4,(R2)	:LOAD THE CAR REG.

004564	010102		
004566	062702	000006	
004572	013737	027312	001220
004600	012737	177777	027312
004606	004737	024544	
004612	000415		
004614	105737	030322	
004620	001001		
004622	005004		
004624	153711	030322	
004630	112761	000000	000016
004636	010412		

3951	004640	062704	010421		ADD	#10421,R4	:GENERATE NEW DATA	
3952	004644	000760			BR	1\$:GO DO NEXT LINE	
3953								
3954	004646	004737	024544	3\$:	JSR	PC,SELINE	:GO SELECT A LINE NO.	
3955	004652	000434			BR	7\$::BR IF CHECKED ALL LINES	
3956	004654	105737	030322		TSTB	LINE	:DOING LINE 00 ?	
3957	004660	001001			BNE	4\$:BR IF NOT	
3958	004662	005004			CLR	R4	:INIT S/B DATA	
3959								
3960	004664	153711	030322	4\$:	BISB	LINE,(R1)	:SELECT A LINE	
3961	004670	112761	000000	000017	MOVB	#0,SSR+1(R1)	:SCOPE SYNC	
3962	004676	011203			MOV	(R2),R3	:GET CONTENTS OF CAR	
3963	004700	020304			CMP	R3,R4	:WAS DATA OK ?	
3964	004702	001412			BEQ	5\$:BR IF YES	
3965								
3966	004704	004737	024412		JSR	PC,SUER2	:GO SET UP ERROR INFO	
3967	004710	004537	024636		JSR	R5,SUNUM	:SET UP LINE NO. IN MSG BUFFER	
3968	004714	030322			LINE			
3969	004716	031155			EM7+47			
3970	004720	012737	004736	001110	MOV	#6\$,SLPERR	:SET UP ERROR LOOP RETURN	
3971	004726	104007			ERROR	7	:CAR ADDRESSING ERROR	
3972								
3973	004730	062704	010421	5\$:	ADD	#10421,R4	:GENERATE NEW S/B DATA	
3974	004734	000744			BR	3\$:GO CHECK NEXT LINE	
3975								
3976	004736	005037	030322	6\$:	CLR	LINE	:RESTART AT LINE 00 IF LOOPING	
3977	004742	000721			BR	1\$:GO RESTART	
3978								
3979	004744	013737	001220	027312	7\$:	MOV	\$TMP7,LINSEL	:RESTORE LINE SELECT PARAMETER

3980
3981
3982
3983 004752 000004
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035

*TEST 15 'BCR' MEMORY ADDRESSING TEST

TST15: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT EACH LOCATION IN THE BYTE COUNT MEMORY CAN BE UNIQUELY ADDRESSED. IT WRITES THE PATTERN SHOWN BELOW INTO THE MEMORY AND THEN READS BACK EACH LOCATION TO VERIFY THAT IT WAS WRITTEN CORRECTLY. SINCE THE MEMORY LOGIC IS PARTITIONED INTO FOUR 16 X 4 READ/WRITE MEMORY CHIPS, THE PATTERN RESULTS IN THE LINE NUMBER (00 - 17(8)) BEING WRITTEN AS DATA INTO TO EACH LOCATION (00 - 17(8)) IN EACH CHIP. THAT IS: LOC00 = 00, LOC 01 = 01,LOC 17 = 17.

MEMORY PATTERN:	LOCATION	CONTENTS	(BOTH GCTAL)
	00	000000	
	01	010421	
	02	021042	
	03	031463	
	04	042104	
	05	052525	
	06	063146	
	07	073567	
	10	104210	
	11	114631	
	12	125252	
	13	135673	
	14	146314	
	15	156735	
	16	167356	
	17	177777	

ANY ERRORS DETECTED ARE REPORTED AND THEN THE TEST RESUMES CHECKING THEE NEXT LOCATION IN SEQUENCE UNTIL ALL 16. HAVE BEEN CHECKED.

NOTE: THIS TEST ALWAYS CHECKS ALL 16. LINES REGARDLESS OF HOW THE LINE SELECTION PARAMETER WAS INITIALLY SET UP.

ERRORS:

1.) ERROR 10 IS CALLED TO REPORT ANY LINES (LOCATIONS) THAT FAIL. THE FAILING LINE # IS INCLUDED AS PART OF THE ERROR HEADER MESSAGE.

SYNC:

1.) WRITE SYNC: M7277 SH4 LOAD SSR LOW BYTE H CR1
2.) READ SYNC: M7277 SH4 LOAD SSR HIGH BYTE H CP2

4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091

DEBUG:

1.) ANALYZE THE ERROR REPORTS CAREFULLY ASKING THE FOLLOWING QUESTIONS:

- A. DOES THE FAULT AFFECT ONLY ONE LINE ?
- B. DOES THE FAULT AFFECT ONLY ONE 4-BIT DATA GROUP ?
IE <15:12>, <11:08>, <07:04>, OR <03:00>
- C. DOES ANY DATA AT ALL APPEAR TO BE WRITTEN ?

2.) IF "A" IS TRUE THEN SUSPECT AN ADDRESSING PROBLEM IN THE MEMORY ADDRESS MUX.

3.) IF "B" IS TRUE THEN SUSPECT A DATA MUX, UP-COUNTER, MEMORY, OR INVERTER CHIP PROBLEM.

4.) IF "C" IS TRUE SUSPECT A MEMORY WRITE TIMING PROBLEM.

5.) IN MOST CASES THE FAULT IS MOST LIKELY THE M7277 OR M7278.

KEY LOGIC:

```

M7277 SH4 LOAD BC H FU2
DATA TO BUS H EN2
DATA SOURCE (A,B,C) DU1,DU2,DT2

SH5 MEMADD SOURCE SEL H E55-8
BC MEM WRITE ENAB L E57-4
BUF ADDRS TO BUS H E33-1 (SHD BE LOW)

M7278 SH3 BITS<15:08>
74157 INPUT MUX CHIPS E18,E19
74193 UP COUNTER CHIPS E27,E26
7489 MEMORY CHIPS E33,E34
7404 INVERTER CHIPS E41,E42

SH4 BITS<07:00>
74157 INPUT MUX CHIPS E16,E17
74193 UP COUNTER CHIPS E24,E25
7489 MEMORY CHIPS E31,E32
7404 INVERTER CHIPS E39,E40

SH5 THRU SH8 74151 DATA MUX OUTPUT CHIPS (PIN 1 INPUT)

```

```

X
MOV R1,R2 ;COPY IT IN R2
ADD #BCR,R2 ;SET UP REGADR IN R2
MOV LINSSEL,$TAP7 ;SAVE LINE SELECT PARAMETER
MOV #-1,LINSSEL ;DO ALL LINES FOR THIS TEST
1$: JSR PC,SELINE ;GO SELECT A LINE NO.
BR 3$ ;;BR IF DONE ALL SELECTED LINES
TSTB LINE ;DOING LINE 00 ?
BNE 2$ ;BR IF NOT
CLR R4 ;INIT TEST DATA

2$: BISB LINE,(R1) ;SELECT A LINE
MOVB #0,SSR(R1) ;SCOPE SYNC

```

```

004754 010102
004756 062702 000010
004762 013737 027312 001220
004770 012737 177777 027312
004776 004737 024544
005002 000415
005004 105737 030322
005010 001001
005012 005004
005014 153711 030322
005020 112761 000000 000016

```

CZDMM-D-0
CZDHMD.P11MACY11 30A(1052)
09-MAR-78 15:3210-MAR-78
T1508:05 PAGE 99
'BCR' MEMORY ADDRESSING TESTSEQ 0097
SEQ 0096

4092	005026	010412				MOV	R4,(R2)	;LOAD THE BCR REG.
4093	005030	062704	010421			ADD	#10421,R4	;GENERATE NEW DATA
4094	005034	000760				BR	1\$;GO DO NEXT LINE
4095								
4096	005036	004737	024544		3\$:	JSR	PC,SELINE	;GO SELECT A LINE NO.
4097	005042	000434				BR	7\$;BR IF CHECKED ALL LINES
4098	005044	105737	030322			TSTB	LINE	;DOING LINE 00 ?
4099	005050	001001				BNE	4\$;BR IF NOT
4100	005052	005004				CLR	R4	;INIT S/B DATA
4101								
4102	005054	153711	030322		4\$:	BISB	LINE,(R1)	;SELECT A LINE
4103	005060	112761	000000	000017		MOVB	#0,SSR+1(R1)	;SCOPE SYNC
4104	005066	011203				MOV	(R2),R3	;GET CONTENTS OF BCR
4105	005070	020304				CMP	R3,R4	;WAS DATA OK ?
4106	005072	001412				BEQ	5\$;BR IF YES
4107								
4108	005074	004737	024412			JSR	PC,SUER2	;GO SET UP ERROR INFO
4109	005100	004537	024636			JSR	R5,SUNUM	;GO SET UP LINE NO. IN MSG BUFFER
4110	005104	030322				LINE		
4111	005106	031224				EM10+44		
4112	005110	012737	005126	001110		MOV	#6\$,\$LPERR	;SET UP ERROR LOOP RETURN
4113	005116	104010				ERROR	10	;BCR ADDRESSING ERROR
4114								
4115	005120	062704	010421		5\$:	ADD	#10421,R4	;GENERATE NEW S/B DATA
4116	005124	000744				BR	3\$;GO CHECK NEXT LINE
4117								
4118	005126	005037	030322		6\$:	CLR	LINE	;RESTART AT LINE 00 IF LOOPING
4119	005132	000721				BR	1\$;GO RESTART
4120								
4121	005134	013737	001220	027312	7\$:	MOV	\$TMP7,LINSEL	;RESTORE THE LINE SELECT PARAMETER

4122
4123
4124
4125 005142 000004
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164 005144 010102
4165 005146 062702 000006
4166 005152 004737 024544
4167 005156 000443
4168 005160 012704 177777
4169 005164 153711 030322
4170 005170 004537 024636
4171 005174 030322
4172 005176 031155
4173
4174 005200 112761 000000 000016 2\$:
4175 005206 010412
4176 005210 011203
4177 005212 020403

```
*****  
: *TEST 16 "CAR" REGISTER TEST - ALL 1'S / ALL 0'S - ALL LINES  
: *****  
TST16: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THE ABILITY TO SET AND CLEAR ALL BITS IN ALL THE SELECTED LOCATIONS (LINES) OF THE CURRENT ADDRESS MEMORY. IT USES THE CONFIGURATION PARAMETER (LINSEL:) TO DEFINE WHICH LINES TO TEST. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A LINE # TO TEST
2. LOAD THE SELECTED LOCATION WITH 177777
3. READ IT BACK TO VERIFY ALL BITS SET
4. LOAD THE SELECTED LOCATION WITH 000000
5. READ IT BACK TO VERIFY ALL BITS CLEARED
6. REPEAT STEPS 1 THRU 5 UNTIL ALL SELECTED LINES ARE TESTED.

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES WITH THE NEXT LINE # IN SEQUENCE AS DEFINED BY "LINSEL".

ERRORS:

1.) ERROR 7 IS CALLED TO REPORT ALL DATA COMPARE ERRORS

SYNC:

- 1.) WRITE 1'S: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) WRITE 0'S: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 14)

KEY LOGIC: (REFER TO TEST 14)

```
%  
1$: MOV R1,R2 ;COPY IT INTO R2  
ADD #CAR,R2 ;R2 GETS CAR ADDRESS  
JSR PC,SELINE ;GO SELECT A LINE NO.  
BR TST17 ;;BR IF DONE ALL SELECTED LINES  
MOV #-1,R4 ;RESULT IN CAR S/B = 177777  
BISB LINE,(R1) ;SELECT A LINE NO.  
JSR R5,SUNUM ;GO SET UP LINE NO. IN MSG BUFFER  
LINE  
EM7+47  
2$: MOVB #0,SSR(R1) ;SCOPE SYNC  
MOV R4,(R2) ;LOAD A CAR WITH ALL ONES  
MOV (R2),R3 ;GET THE WAS DATA FROM THE CAR  
CMP R4,R3 ;DID IT CONTAIN ALL ONES ??
```

```
4178 005214 001406      BEQ      3$          ;;BR IF ALL 1'S
4179
4180 005216 004737 024412      JSR      PC,SUER2    ;GO SET UP ERROR INFO
4181 005222 012737 005200 001110     MOV      #2$, $LPERR ;SET UP ERROR LOOP RETURN
4182 005230 104007      ERROR    7          ;FAILED TO SET ALL 1'S IN SELECTED CAR
4183
4184 005232 005004      CLR      R4          ;RESULT IN CAR S/B = 000000
4185 005234 112761 000000 000017     3$:     MOVB    #0,SSR+1(R1) ;SCOPE SYNC
4186 005242 010412      MOV      R4,(R2)     ;CLEAR SELECTED CAR
4187 005244 011203      MOV      (R2),R3     ;GET THE WAS DATA
4188 005246 001741      BEQ      1$          ;BR IF CAR GOT CLEARED
4189
4190 005250 004737 024412      JSR      PC,SUER2    ;GO SET UP FOR ERROR CALL
4191 005254 012737 005232 001110     MOV      #3$, $LPERR ;SET UP ERROR LOOP RETURN
4192 005262 104007      ERROR    7          ;FAILED TO CLR ALL BITS IN SELECTED CAR
4193 005264 000732      BR       1$          ;GO TEST NEXT LINE
```

4194
4195
4196
4197 005266 000004
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236 005270 010102
4237 005272 062702 000010
4238 005276 004737 024544
4239 005302 000443
4240 005304 012704 177777
4241 005310 153711 030322
4242 005314 004537 024636
4243 005320 030322
4244 005322 031224
4245
4246 005324 112761 000000 000016 2\$:
4247 005332 010412
4248 005334 011203
4249 005336 020403

```
*****  
: *TEST 17 'BCR' REGISTER TEST - ALL 1'S / ALL 0'S - ALL LINES  
: *****  
TST17: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THE ABILITY TO SET AND CLEAR ALL BITS IN ALL THE SELECTED LOCATIONS (LINES) OF THE BYTE COUNT MEMORY. IT USES THE CONFIGURATION PARAMETER (LINSEL:) TO DEFINE WHICH LINES TO TEST. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A LINE # TO TEST
2. LOAD THE SELECTED LOCATION WITH 177777
3. READ IT BACK TO VERIFY ALL BITS SET
4. LOAD THE SELECTED LOCATION WITH 000000
5. READ IT BACK TO VERIFY ALL BITS CLEARED
6. REPEAT STEPS 1 THRU 5 UNTIL ALL SELECTED LINES ARE TESTED.

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES WITH THE NEXT LINE # IN SEQUENCE AS DEFINED BY 'LINSEL'.

ERRORS:

1.) ERROR 10 IS CALLED TO REPORT ALL DATA COMPARE ERRORS

SYNC:

- 1.) WRITE 1'S: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) WRITE 0'S: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 15)

KEY LOGIC: (REFER TO TEST 15)

```
%  
MOV R1,R2 ;COPY IT INTO R2  
ADD #BCR,R2 ;R2 GETS BCR ADDRESS  
1$: JSR PC,SELINE ;GO SELECT A LINE NO.  
BR TST20 ;:BR IF DONE ALL SELECTED LINES  
MOV #-1,R4 ;RESULT IN BCR S/B = 177777  
BISB LINE,(R1) ;SELECT A LINE NO.  
JSR R5,SUNUM ;GO SET UP LINE NO. IN MSG BUFFER  
LINE  
EM10+44  
2$: MOVB #0,SSR(R1) ;SCOPE SYNC  
MOV R4,(R2) ;LOAD A BCR WITH ALL ONES  
MOV (R2),R3 ;GET THE WAS DATA FROM THE BCR  
CMP R4,R3 ;DID IT CONTAIN ALL ONES ??
```

4250	005340	001406			BEQ	3\$::BR IF ALL 1'S
4251								
4252	005342	004737	024412		JSR	PC,SUER2		:GO SET UP ERROR INFO
4253	005346	012737	005324	001110	MOV	#2\$,\$LPERR		:SET UP ERROR LOOP RETURN
4254	005354	104010			ERROR	10		:FAILED TO SET ALL 1'S IN SELECTED BCR
4255								
4256	005356	005004			CLR	R4		:RESULT IN BCR S/B = 000000
4257	005360	112761	000000	000017	MOVB	#0,SSR+1(R1)		:SCOPE SYNC
4258	005366	010412			MOV	R4,(R2)		:CLEAR SELECTED BCR
4259	005370	011203			MOV	(R2),R3		:GET THE WAS DATA
4260	005372	001741			BEQ	1\$:BR IF BCR GOT CLEARED
4261								
4262	005374	004737	024412		JSR	PC,SUER2		:GO SET UP FOR ERROR CALL
4263	005400	012737	005356	001110	MOV	#3\$,\$LPERR		:SET UP ERROR LOOP RETURN
4264	005406	104010			ERROR	10		:FAILED TO CLR ALL BITS IN SELECTED BCR
4265	005410	000732			BR	1\$:GO TEST NEXT LINE

4266
4267
4268
4269 005412 000004
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312 005414 010102
4313 005416 062702 000006
4314 005422 012705 027330
4315 005426 012537 001204
4316 005432 001472
4317 005434 004737 024544
4318 005440 000772
4319 005442 113737 030322 030324
4320
4321 005450 105037 001210

```
*****  
;+TEST 20 "CAR" MEMORY PATTERNS TEST / O'S DISTURB  
*****  
TST20: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT WHEN A TEST PATTERN IS WRITTEN INTO LOCATION 'N' OF THE 'CAR' MEMORY, IT DOES NOT DISTURB ANY BITS IN ANY OTHER LOCATIONS. THERE ARE THREE TEST PATTERNS USED* (177777, 125252, 052525) FOR EACH LOCATION SELECTED BY THE CONFIGURATION PARAMETER 'LINSEL'. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A TEST PATTERN
2. SELECT A LINE # TO TEST
3. CLEAR ALL 16. LOCATIONS IN THE MEMORY
4. WRITE THE TEST PATTERN INTO THE SELECTED LOCATION
5. VERIFY THAT THE PATTERN WAS WRITTEN CORRECTLY AND THAT NO OTHER LOCATIONS WERE DISTURBED.
6. REPEAT 2 THRU 5 UNTIL ALL SELECTED LINES TESTED
7. REPEAT 1 THRU 6 UNTIL ALL THREE PATTERNS TESTED

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES WITH CHECKING THE NEXT LINE IN SEQUENCE.

ERRORS:

- 1.) ERROR 46 IS CALLED TO REPORT ANY ERROR DETECTED. THE INFORMATION PRINTED INCLUDES THE LINE # WRITTEN, THE LINE # BEING CHECKED, AND THE PATTERN USED.

SYNC*

- 1.) WRITE LINE: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) READ CHECK: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 14)

KEY LOGIC: (REFER TO TEST 14)

```
%  
MOV R1,R2 ;SET UP REGADR  
ADD #CAR,R2  
MOV #PATR1,R5 ;SET UP POINTER TO DATA PATTERNS  
1$: MOV (R5)+,$TMP1 ;GET A DATA TEST PATTERN  
BEQ TST21 ;;BR IF DONE THREE PATTERNS  
11$: JSR PC,SELINE ;GO SELECT A LINE TO TEST  
BR 1$ ;BR IF DONE ALL SELECTED LINES  
MOV B LINE,LINSEL ;SAVE THE LINE NO. FOR ERROR LOOPING  
2$: CLRB $TMP3 ;INIT LINE COUNTER
```

```
4322 005454 113711 001210 3$: MOVB $TMP3,(R1) ;SELECT A LINE TO CLEAR
4323 005460 005012 CLR (R2) ;CLR CAR FOR THAT LINE
4324 005462 105237 001210 INCB $TMP3 ;GENERATE NEW LINE NO.
4325 005466 123727 001210 000020 CMPB $TMP3,#20 ;DONE CLEARING ALL LINES ?
4326 005474 001367 BNE 3$ ;BR IF NOT
4327
4328 005476 113711 030324 MOVB LINEA,(R1) ;SET LINE SELECT BITS
4329 005502 112761 000000 000016 MOVB #0,SSR(R1) ;SCOPE SYNC
4330 005510 013712 001204 MOV $TMP1,(R2) ;LOAD CAR WITH TEST PATTERN
4331
4332 005514 105037 001206 CLRB $TMP2 ;INIT A LINE COUNTER
4333 005520 013704 001204 4$: MOV $TMP1,R4 ;SET UP S/B DATA
4334 005524 113711 001206 MOVB $TMP2,(R1) ;SET LINE SELECT IN SCR
4335 005530 112761 000000 000017 MOVB #0,SSR+1(R1) ;SCOPE SYNC
4336 005536 011203 MOV (R2),R3 ;GET WAS DATA
4337 005540 123737 001206 030322 CMPB $TMP2,LINE ;IS THIS THE LINE WITH THE TEST PATTERN
4338 005546 001401 BEQ 5$ ;BR IF IT IS
4339 005550 005004 CLR R4 ;MAKE S/B DATA = 000000
4340 005552 020304 5$: CMP R3,R4 ;CORRECT DATA IN CAR ?
4341 005554 001412 BEQ 6$ ;BR IF YES
4342
4343 005556 004737 024500 JSR PC,SUER4 ;GO SET UP ERROR IN FO
4344 005562 004537 024636 JSR R5,SUNUM ;GO SET UP LINE NO. IN MSG BUFFER
4345 005566 001206 $TMP2
4346 005570 033776 EM46+63
4347 005572 012737 005450 001110 MOV #2$,$LPERR ;SET UP ERROR LOOP RETURN
4348 005600 104046 ERROR 46 ;INCORRECT DATA READ FROM CAR
4349
4350 005602 105237 001206 6$: INCB $TMP2 ;GENERATE NEXT LINE NO.
4351 005606 122737 000020 001206 CMPB #20,$TMP2 ;DONE ALL LINES ?
4352 005614 001707 BEQ 11$ ;BR IF YES
4353 005616 000740 BR 4$ ;GO CHECK NEXT LINE
```


4354
4355
4356
4357 005620 000004
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4399
4400 005622 010102
4401 005624 062702 000010
4402 005630 012705 027330
4403 005634 012537 001204
4404 005640 001472
4405 005642 004737 024544
4406 005646 000772
4407 005650 113737 030322 030324
4408
4409 005656 105037 001210

: *TEST 21 'BCR' MEMORY PATTERNS TEST / O'S DISTURB
: *****
TST21: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT WHEN A TEST PATTERN IS WRITTEN INTO LOCATION 'N' OF THE 'BCR' MEMORY, IT DOES NOT DISTURB ANY BITS IN ANY OTHER LOCATIONS. THERE ARE THREE TEST PATTERNS USED* (177777, 125252, 052525) FOR EACH LOCATION SELECTED BY THE CONFIGURATION PARAMETER 'LINSEL'. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A TEST PATTERN
2. SELECT A LINE # TO TEST
3. CLEAR ALL 16. LOCATIONS IN THE MEMORY
4. WRITE THE TEST PATTERN INTO THE SELECTED LOCATION
5. VERIFY THAT THE PATTERN WAS WRITTEN CORRECTLY AND THAT NO OTHER LOCATIONS WERE DISTURBED.
6. REPEAT 2 THRU 5 UNTIL ALL SELECTED LINES TESTED
7. REPEAT 1 THRU 6 UNTIL ALL THREE PATTERNS TESTED

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES WITH CHECKING THE NEXT LINE IN SEQUENCE.

ERRORS:

- 1.) ERROR 47 IS CALLED TO REPORT ANY ERROR DETECTED. THE INFORMATION PRINTED INCLUDES THE LINE # WRITTEN, THE LINE # BEING CHECKED, AND THE PATTERN USED.

SYNC*

- 1.) WRITE LINE: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) READ CHECK: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 15)

KEY LOGIC: (REFER TO TEST 15)

%

```
MOV R1,R2 ;SET UP REGADR
ADD #BCR,R2
MOV #PATRMA,R5 ;SET UP POINTER TO DATA PATTERNS
1$: MOV (R5)+,$STMP1 ;GET A DATA TEST PATTERN
BEQ TST22 ;:BR IF DONE THREE PATTERNS
11$: JSR PC,SELINEL ;GO SELECT A LINE TO TEST
BR 1$ ;:BR IF SELECTED ALL LINES
MOVBL LINE,LINEL ;SAVE THE LINE NO. FOR ERROR LOOP
2$: CLRB $STMP3 ;INIT LINE COUNTER
```

```
4410 005662 113711 001210      3$:  MOVB  $TMP3,(R1)      ;SELECT A LINE TO CLEAR
4411 005666 005012                CLR  (R2)              ;CLR BCR FOR THAT LINE
4412 005670 105237 001210      INCB  $TMP3            ;GENERATE NEW LINE NO.
4413 005674 123727 001210 000020  CMPB  $TMP3,#20        ;DONE CLEARING ALL LINES ?
4414 005702 001367                BNE   3$              ;BR IF NOT
4415
4416 005704 113711 030324      MOVB  LINEA,(R1)       ;SET LINE SELECT BITS
4417 005710 112761 000000 000016  MOVB  #0,SSR(R1)      ;SCOPE SYNC
4418 005716 013712 001204      MOV   $TMP1,(R2)      ;LOAD BCR WITH TEST PATTERN
4419
4420 005722 105037 001206      CLRB  $TMP2           ;INIT A LINE COUNTER
4421 005726 013704 001204      4$:  MOV   $TMP1,R4       ;SET UP S/B DATA
4422 005732 113711 001206      MOVB  $TMP2,(R1)      ;SELECT A LINE TO CHECK
4423 005736 112761 000000 000017  MOVB  #0,SSR+1(R1)    ;SCOPE SYNC
4424 005744 011203                MOV   (R2),R3         ;GET WAS DATA
4425 005746 123737 001206 030322  CMPB  $TMP2,LINE      ;IS THIS THE LINE WITH THE TEST PATTERN
4426 005754 001401                BEQ   5$              ;BR IF IT IS
4427 005756 005004                CLR  R4               ;MAKE S/B DATA = 000000
4428 005760 020304      5$:  CMP   R3,R4          ;CORRECT DATA IN BCR ?
4429 005762 001412                BEQ   6$              ;BR IF YES
4430
4431 005764 004737 024500      JSR   PC,SUER4        ;GO SET UP ERROR IN FO
4432 005770 004537 024636      JSR   R5,SUNUM        ;GO SET UP LINE NO.IN MSG BUFFER
4433 005774 001206                $TMP2
4434 005776 034176                EM47+56
4435 006000 012737 005656 001110  MOV   #2$,$LPERR      ;SET UP ERROR LOOP RETURN
4436 006006 104047                ERROR 47              ;INCORRECT DATA READ FROM BCR
4437
4438 006010 105237 001206      6$:  INCB  $TMP2           ;GENERATE NEXT LINE NO.
4439 006014 122737 000020 001206  CMPB  #20,$TMP2       ;DONE ALL LINES ?
4440 006022 001707                BEQ   11$             ;BR IF YES
4441 006024 000740                BR    4$              ;GO CHECK NEXT LINE
```

4442
4443
4444
4445 006026 000004
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485 006030 010102
4486 006032 062702 000006
4487 006036 012705 177777
4488 006042 010537 001204
4489 006046 004737 024544
4490 006052 000465
4491 006054 113737 030322 030324
4492
4493 006062 105037 001210
4494 006066 113711 0C1210
4495 006072 010512
4496 006074 105237 001210
4497 006100 123727 001210 000020

: *TEST 22 'CAR' MEMORY PATTERNS TEST / 1'S DISTURB
: *****
TST22: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT WHEN ALL ZEROS ARE WRITTEN INTO LINE 'N'
IN THE 'CAR' MEMORY, IT DOES NOT CLEAR ANY BITS IN ANY OTHER LOCATIONS.
ONLY THE LINES SELECTED BY 'LINSEL' ARE TESTED. THE TEST SEQUENCE IS
AS FOLLOWS:

1. SELECT A LINE TO TEST
2. SET ALL ONES (177777) INTO ALL MEMORY LOCATIONS
3. CLEAR THE SELECTED LINE
4. VERIFY THAT ONLY THE SELECTED LINE WAS CLEARED
AND ALL OTHER LINES STILL CONTAIN 177777
5. REPEAT STEPS 1 THRU 4 UNTIL ALL SELECTED LINES ARE TESTED

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES CHECKING THE NEXT
LINE IN SEQUENCE.

ERRORS:

- 1.) ERROR 46 IS CALLED TO REPORT ALL ERRORS. THE INFORMATION PRINTED
INCLUDES THE LINE # WRITTEN, THE LINE # BEING CHECKED, AND THE
PATTERN USED.

SYNC:

- 1.) WRITE LINE: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) CHECK LINE: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 14)

KEY LOGIC: (REFER TO TEST 14)

X
MOV R1,R2 ;SET UP REGADR
ADD #CAR,R2
MOV #-1,R5 ;TEST PATERRN IN R5 = 177777
MOV R5,\$TMP1 ;SAVE FOR ERROR REPORTING
1\$: JSR PC,SELINE ;GO SELECT A LINE TO TEST
BR TST23 ;:BR IF DONE ALL LINES
MOV LINE,LINEA ;SAVE THE LINE NO. FOR ERROR LOOP
2\$: CLRB \$TMP3 ;INIT LINE COUNTER
3\$: MOVB \$TMP3,(R1) ;SELECT A LINE TO CLEAR
MOV R5,(R2) ;LOAD CAR WITH 177777
INCB \$TMP3 ;GENERATE NEW LINE NO.
CMPB \$TMP3,#20 ;DONE SETTING ALL LINES TO 177777 ?

```

4498 006106 001367          BNE      3$          ;BR IF NOT
4499
4500 006110 113711 030324    MOVB     LINEA,(R1)   ;SET LINE SELECT IN SCR
4501 006114 112761 000000 000016    MOVB     #0,SSR(R1)  ;SCOPE SYNC
4502 006122 005012          CLR      (R2)        ;CLEAR THE CAR UNDER TEST
4503
4504 006124 105037 001206          CLRB     $TMP2       ;INIT A LINE COUNTER
4505 006130 005004          CLR      R4          ;MAKE S/B DATA = 000000
4506 006132 113711 001206    4$:     MOVB     $TMP2,(R1) ;SELECT A LINE TO CHECK
4507 006136 112761 000000 000017    MOVB     #0,SSR+1(R1) ;SCOPE SYNC
4508 006144 011203          MOV      (R2),R3     ;GET WAS DATA
4509 006146 123737 001206 030322    CMPB     $TMP2,LINE  ;IS THIS THE LINE WITH THE TEST PATTERN
4510 006154 001401          BEQ      5$          ;BR IF IT IS
4511 006156 010504          MOV      R5,R4       ;MAKE S/B DATA = 177777
4512 006160 020304    5$:     CMP      R3,R4     ;CORRECT DATA IN CAR ?
4513 006162 001412          BEQ      6$          ;BR IF YES
4514
4515 006164 004737 024500    JSR      PC,SUER4    ;GO SET UP ERROR IN FO
4516 006170 004537 024636    JSR      R5,SUNUM    ;GO SET UP LINE NO. IN MSG BUFFER
4517 006174 001206          $TMP2
4518 006176 033776          EM46+63
4519 006200 012737 006062 001110    MOV      #2$,$LPERR  ;SET UP ERROR LOOP RETURN
4520 006206 104046          ERROR     46         ;INCORRECT DATA READ FROM CAR
4521
4522 006210 105237 001206    6$:     INCB     $TMP2     ;GENERATE NEXT LINE NO.
4523 006214 122737 000020 001206    CMPB     #20,$TMP2   ;DONE ALL LINES ?
4524 006222 001711          BEQ      1$          ;BR IF YES
4525 006224 000741          BR      4$          ;GO CHECK NEXT LINE

```

4526
4527
4528
4529 006226 000004
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569 006230 010102
4570 006232 062702 000010
4571 006236 012705 177777
4572 006242 010537 001204
4573 006246 004737 024544
4574 006252 000465
4575 006254 113737 030322 030324
4576
4577 006262 105037 001210
4578 006266 113711 001210
4579 006272 010512
4580 006274 105237 001210
4581 006300 123727 001210 000020

```
*****  
: *TEST 23 'BCR' MEMORY PATTERNS TEST / 1'S DISTURB  
: *****  
TST23: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT WHEN ALL ZEROS ARE WRITTEN INTO LINE 'N'
IN THE 'BCR' MEMORY, IT DOES NOT CLEAR ANY BITS IN ANY OTHER LOCATIONS.
ONLY THE LINES SELECTED BY 'LINSEL' ARE TESTED. THE TEST SEQUENCE IS
AS FOLLOWS:

1. SELECT A LINE TO TEST
2. SET ALL ONES (177777) INTO ALL MEMORY LOCATIONS
3. CLEAR THE SELECTED LINE
4. VERIFY THAT ONLY THE SELECTED LINE WAS CLEARED
AND ALL OTHER LINES STILL CONTAIN 177777
5. REPEAT STEPS 1 THRU 4 UNTIL ALL SELECTED LINES ARE TESTED

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES CHECKING THE NEXT
LINE IN SEQUENCE.

ERRORS:

- 1.) ERROR 47 IS CALLED TO REPORT ALL ERRORS. THE INFORMATION PRINTED
INCLUDES THE LINE # WRITTEN, THE LINE # BEING CHECKED, AND THE
PATTERN USED.

SYNC:

- 1.) WRITE LINE: M7277 SH4 LOAD SSR LOW BYTE H CR1
- 2.) CHECK LINE: M7277 SH4 LOAD SSR HIGH BYTE H CP2

DEBUG: (REFER TO TEST 15)

KEY LOGIC: (REFER TO TEST 15)

```
%  
MOV R1,R2 ;SET UP REGADR  
ADD #BCR,R2  
MOV #-1,R5 ;TEST PATERRN IN R5 = 177777  
MOV R5,$TMP1 ;SAVE IT FOR ERROR REPORTING  
1$: JSR PC,SELINE ;GO SELECT A LINE TO TEST  
BR TST24 ;;BR IF DONE ALL LINES  
MOVB LINE,LINEA ;SAVE THE LINE NO.  
2$: CLRB $TMP3 ;INIT LINE COUNTER  
3$: MOVB $TMP3,(R1) ;SELECT A LINE TO INJT  
MOV R5,(R2) ;LOAD BCR WITH 177777  
INCB $TMP3 ;GENERATE NEW LINE NO.  
4581: CMPB $TMP3,#20 ;DONE SETTING ALL LINES TO 177777 ?
```

CZDMM-D-0
CZDHMD.P11MACY11 30A(1052) 10-MAR-78
09-MAR-78 15:32 T2308:05 PAGE 111
'BCR' MEMORY PATTERNS TEST / 1'S DISTURBSEQ 0109
SEQ 0108

```

4582 006306 001367      BNE      3$      ;BR IF NOT
4583
4584 006310 113711 030324      MOVB     LINEA,(R1) ;SET LINE SELECT BITS
4585 006314 112761 000000 000016      MOVB     #0,SSR(R1) ;SCOPE SYNC
4586 006322 005012      CLR      (R2)      ;CLEAR THE BCR UNDER TEST
4587
4588 006324 105037 001206      CLRB     $TMP2     ;INIT A LINE COUNTER
4589 006330 005004      CLR      R4        ;MAKE S/B DATA = 000000
4590 006332 113711 001206      4$:     MOVB     $TMP2,(R1) ;SELECT A LINE TO CHECK
4591 006336 112761 000000 000017      MOVB     #0,SSR+1(R1) ;SCOPE SYNC
4592 006344 011203      MOV      (R2),R3   ;GET WAS DATA
4593 006346 123737 001206 030322      CMPB     $TMP2,LINE ;IS THIS THE LINE WITH THE TEST PATTERN
4594 006354 001401      BEQ      5$        ;BR IF IT IS
4595 006356 010504      MOV      R5,R4     ;MAKE S/B DATA = 177777
4596 006360 020304      5$:     CMP      R3,R4     ;CORRECT DATA IN BCR ?
4597 006362 001412      BEQ      6$        ;BR IF YES
4598
4599 006364 004737 024500      JSR      PC,SUER4   ;GO SET UP ERROR IN FO
4600 006370 004537 024636      JSR      R5,SUNUM   ;GO SET UP LINE NO. IN MSG BUFFER
4601 006374 001206      $TMP2
4602 006376 034176      EM47+56
4603 006400 012737 006262 001110      MOV      #2$,$LPERR ;SET UP ERROR LOOP RETURN
4604 006406 104047      ERROR    47        ;INCORRECT DATA READ FROM BCR
4605
4606 006410 105237 001206      6$:     INCB     $TMP2     ;GENERATE NEXT LINE NO.
4607 006414 122737 000020 001206      CMPB     #20,$TMP2 ;DONE ALL LINES ?
4608 006422 001711      BEQ      1$        ;BR IF YES
4609 006424 000741      BR      4$        ;GO CHECK NEXT LINE

```

4610
4611
4612
4613 006426 000004
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665

: *TEST 24 TEST THAT "CAR" MEMORY EXT BITS SET/CLR PROPERLY
: *****
TST24: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE "EXT MEM" BITS (CAR<17:16> CAN BE SET AND CLEARED IN ALL "CAR" MEMORY LOCATIONS. IT WRITES THE BINARY TEST PATTERNS (11, 01, AND 10) INTO BITS<17:16> TO CHECK EVERY MEMORY LOCATION. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A TEST PATTERN TO USE
2. CLEAR ALL 18 BITS IN ALL 16 LOCATIONS
3. SELECT A LINE TO TEST
4. WRITE THE TEST PATTERN INTO <17:16> OF THE SELECTED LOCATION
5. READ CHECK ALL LOCATIONS TO VERIFY THAT ONLY THE SELECTED LOCATION CONTAINS THE PATTERN
6. REPEAT STEPS 3 THRU 5 UNTIL ALL SELECTED LINES TESTED
7. REPEAT STEPS 1 THRU 6 UNTIL ALL PATTERNS USED

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES CHECKING THE NEXT LINE IN SEQUENCE.

- NOTES: 1.) BITS<05:04> IN THE "SCR" ARE USED TO WRITE THE EXT MEM BITS
2.) BITS<07:06> IN THE "SSR" ARE USED TO CHECK BITS<17:16>

ERRORS:

1.) ERROR 7 IS CALLED TO REPORT ALL ERRORS

SYNC:

- 1.) WRITE CAR: M7277 SH4 LOAD LPR H EP2
- 2.) READ CAR: M7277 SH4 LOAD BCR H FU2

DEBUG:

- 1.) ASSUMING THAT THE PREVIOUS "CAR" MEMORY TESTS RAN ERROR FREE, THE PROBLEM IS EITHER THE M7277 OR THE M7278
- 2.) SET UP SCOPE ERROR LOOP AND START BACKTRACKING THROUGH THE LOGIC STARTING WITH THE KEY SIGNALS BELOW.

KEY LOGIC:

M7277	SH5	SCR05 H	CD2
		SCR04 H	CE1
		SSR07 H	CF1

```

4666                                     SSR06 H          CH1
4667
4668                                     M7278  SH7    74151 MUX CHIPS E66 AND E58 (INPUT PIN 12)
4669
4670                                     NOTE:  THER MAY BE A PRINT ERROR ON SH7 OF THE M7278. THE
4671                                     SIGNALS INTO THE MUX CHIPS E66 AND E58 COME FROM THE
4672                                     M7277 SH5 RATHER FROM M7279 SH3.
4673
4674                                     X
4675 006430 010102                       MOV    R1,R2          ;SET UP REGADR
4676 006432 062702 000016                 ADD    #SSR,R2
4677 006436 012705 027340                 MOV    #PATRNB,R5   ;SET UP POINTER TO DATA PATTERNS
4678 006442 012537 001204                 1$:   MOV    (R5)+,$TMP1 ;GET THE PATTERNS
4679 006446 012537 001206                 MOV    (R5)+,$TMP2
4680 006452 001505                       BEQ    TST25         ;;BR IF DONE ALL PATTERNS
4681
4682 006454 105037 001210                 2$:   CLRB   $TMP3      ;INIT A LINE COUNTER
4683 006460 142711 000017                 3$:   BICB   #17,(R1)    ;INIT LINE SELECT BITS IN "SCR"
4684 006464 153711 001210                 BISB   $TMP3,(R1)   ;SELECT A LINE IN SCR
4685 006470 142711 000060                 BICB   #60,(R1)    ;SO WE CLEAR ALL THE MEM EXT BITS
4686 006474 005061 000006                 CLR    CAR(R1) ;CLEAR A CAR
4687 006500 105237 001210                 INCB   $TMP3        ;GENERATE NXT LINE NO.
4688 006504 122737 000020 001210         CMPB   #20,$TMP3    ;CLEARED THE WHOLE THING ?
4689 006512 001362                       BNE    3$           ;BR IF NOT
4690
4691 006514 004737 024544                 4$:   JSR    PC,SELINE  ;GO SELECT A LINE NO.
4692 006520 000750                       BR     1$           ;BR IF DONE ALL LINES
4693 006522 153711 030322                 BISB   LINE,(R1)   ;SET UP LINE SELECT BITS
4694 006526 153711 001204                 BISB   $TMP1,(R1) ;SET UP MEM EXT BIT PATTERN
4695 006532 012761 000000 000004         MOV    #0,LPR(R1) ;SCOPE SYNC
4696 006540 012761 000000 000006         MOV    #0,CAR(R1) ;WRITE EXT BITS IN THIS LOCATION
4697
4698 006546 105037 001212                 CLRB   $TMP4        ;INIT A LINE COUNTER
4699 006552 013704 001206                 5$:   MOV    $TMP2,R4    ;SET UP S/B DATA
4700 006556 142711 000017                 BICB   #17,(R1)    ;INIT SELECT BITS IN "SCR"
4701 006562 153711 001212                 BISB   $TMP4,(R1)  ;SET SELECT BITS IN SCR
4702 006566 012761 000000 000014         MOV    #0,BKR(R1) ;SCOPE SYNC
4703 006574 016103 000006                 MOV    CAR(R1),R3  ;READ THE SELECTED "CAR"
4704 006600 011203                       MOV    (R2),R3     ;GET THE WAS DATA
4705 006602 042703 177477                 BIC    #177477,R3  ;CLEAR JUNK BITS
4706 006606 123737 030322 001212         CMPB   LINE,$TMP4  ;LINE UNDER TEST ??
4707 006614 001401                       BEQ    6$           ;BR IF YES
4708 006616 005004                       CLR    R4           ;MAKE S/B DATA = 000000
4709 006620 020304                 6$:   CMP    R3,R4      ;WERE MEM EXT BITS CORRECT ?
4710 006622 001412                       BEQ    7$           ;BR IF YES
4711
4712 006624 004737 024412                 JSR    PC,SUER2    ;GO SET UP ERROR INFO
4713 006630 004537 024636                 JSR    R5,SUNUM   ;GO SET LINE NO. IN MSG BUFFER
4714 006634 001212                       $TMP4
4715 006636 031155                       EM7+47
4716 006640 012737 006454 001110         MOV    #2$,$LPERR ;SET UP ERROR LOOP RETURN
4717 006646 104007                       ERROR  7           ;MEM EXT BITS READ INCORRECTLY
4718
4719 006650 105237 001212                 7$:   INCB   $TMP4        ;GENERATE NXT LINE NO.
4720 006654 122737 000020 001212         CMPB   #20,$TMP4  ;DONE ALL LINES
4721 006662 001674                       BEQ    2$          ;BR IF YES

```


CZDMM-D-0 MACY11 30A(1052) 10-MAR-78 08:05 PAGE 114 H 9 SEQ 0112
CZDHMD.P11 09-MAR-78 15:32 T24 TEST THAT "CAR" MEMORY EXT BITS SET/CLR PROPERLY SEQ 0111
4722 006664 000732 BR 5\$;GO CHECK NEXT LINE

4723
4724
4725
4726 006666 000004
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772 006670 012737 006746 001110
4773 006676 010102
4774 006700 013703 027304
4775 006704 012723 006776
4776 006710 113723 030316
4777 006714 105723
4778 006716 012723 007020

*TEST 25 TEST INTR. ENAB. BITS - INTR. CONDITION DISABLED

TST25: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT NO TRANSMITTER OR RECEIVER INTERRUPT OCCURS WHEN THE ENABLE BIT IS SET WITH OUT THE INTERRUPTING CONDITION ACTIVE. A BIT MASK (INTMSK: 030100) IS USED TO DEFINE THE I.E. BITS. IN THE "SCR" (BITS 13, 12, AND 06). THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE XMIT AND RCVR VECTORS
2. SELECT AN I.E. BIT TO TEST
3. INIT THE SP AND LOCK OUT INTERRUPTS
4. SET THE SELECTED BIT IN THE "SCR"
5. CLEAR THE PSW TO ALLOW INTRs
6. IF NO INTR: REPEAT 2 THRU 5 UNTIL ALL BITS TESTED
7. IF INTR: REPORT ERROR AND CONTINUE WITH NEXT BIT TO TEST

ALL ERRORS ARE REPORTED AND THEN THE TEST RESUMES WITH THE NEXT BIT IN SEQUENCE .

ERRORS:

- 1.) ERROR 11 IS CALLED TO REPORT RCVR INTR FAULTS
- 2.) ERROR 12 IS CALLED TO REPORT XMITTR INTR FAULTS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

- 1.) PROBLEM IS MOST LIKELY THE M7289 MODULE IF THIS IS THE FIRST TEST TO FAIL.
- 2.) SET UP SCOPE ERROR LOOP AND BACKTRACK THROUGH THE LOGIC STARTING WITH THE KEY LOGIC BELOW.

KEY LOGIC:

M7289 SH6 XMIT INT REQ H FM1
RCV INT REQ H DP1

```

%
MOV #3$, $LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;MAKE IT REGADR TOO
MOV DHVCT,R3 ;GET FIRST VECTOR ADDRESS
MOV #4$, (R3)+ ;GO TO 3$ IF RCVR INTRs
MOVB DHRLVL, (R3)+
TSTB (R3)+ ;UPDATE POINTER
MOV #5$, (R3)+ ;GO TO 5$ IF XMITTR INTRs

```

```
4779 006722 113713 030317      MOVB   DHTLVL,(R3)
4780 006726 012705 000001      MOV    #1,R5          ;INIT BIT TEST MARKER
4781 006732 030537 027676      1$:   BIT    R5,INTMSK  ;TEST THIS BIT ??
4782 006736 001003              BNE    3$            ;BR IF YES
4783 006740 006305      2$:   ASL    R5          ;SHIFT THE MARKER
4784 006742 001437      BEQ    6$            ;BR IF TESTED ALL REQUIRED BITS
4785 006744 000772      BR     1$            ;GO TEST FOR THIS ONE
4786
4787 006746 012706 001100      3$:   MOV    #STACK,SP  ;RESET SP FOR ERROR LOOPING
4788 006752 004737 027164      JSR   PC,CHPS2       ;GO LOCK OUT INTR
4789 006756 012711 004000      MOV   #BIT11,(R1)    ;CLEAR THE DH11 INTERFACE
4790 006762 010504      MOV   R5,R4          ;SET UP S/B DATA
4791 006764 050511      BIS   R5,(R1)        ;SET THE TEST I.E. BIT
4792 006766 004737 027150      JSR   PC,CHPS1       ;GO CLEAR PSW
4793 006772 000240      NOP
4794 006774 000761      BR    2$            ;WAIT A BIT TO ALLOW INTR
4795
4796 006776 004737 027200      4$:   JSR   PC,SAPS      ;SAVE THE ERROR PSW
4797 007002 011103      MOV   (R1),R3        ;GET THE WAS DATA
4798 007004 004737 024416      JSR   PC,SUER2A      ;GO SET UP ERROR INFO
4799 007010 104011      ERROR 11             ;DH11 RCVR SHOULD NOT HAVE INTERRUPTED
4800 007012 012716 006740      MOV   #2$,(SP)      ;SET UP TO RETURN
4801 007016 000002      RTI                  ;RETURN TO TEST NEXT BIT
4802
4803 007020 004737 027200      5$:   JSR   PC,SAPS      ;SAVE THE ERROR PSW
4804 007024 011103      MOV   (R1),R3        ;GET THE WAS DATA
4805 007026 004737 024416      JSR   PC,SUER2A      ;GO SET UP ERROR INFO
4806 007032 104012      ERROR 12             ;XMITTER SHOULD NOT HAVE INTERRUPTED
4807 007034 012716 006740      MOV   #2$,(SP)      ;SET UP TO RETURN
4808 007040 000002      RTI                  ;RETURN TO TEST NEXT BIT
4809
4810 007042 012706 001100      6$:   MOV   #STACK,SP  ;RESET THE SP JUST IN CASE
4811 007046 004737 026770      JSR   PC,RESTRP     ;GO RESTORE TRAP CATCHER IN VECTOR
```

4812
4813
4814
4815 007052 000004
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867

*TEST 26 TEST CHAR. AVAIL. I.E. WITH INTR. CONDITION ACTIVE

TST26: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST USES MAINT. MODE (SCR09=1) TO SET THE CHAR AVAIL BIT (SCR7) TO GENERATE A RCVR INTERRUPT THROUGH THE PROPER VECTOR. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE VECTORS
2. ISSUE DH11 'MASTER CLR', RESET THE SP, AND LOCK OUT INTR
3. PRINT THE DH11 TO GENERATE A RCVR INTR
4. CLEAR THE PSW TO ALLOW INTR
5. REPORT NO RCVR INTR OR FALSE XMITTR INTR.

ALL ERRORS ARE REPORTED AND THEN THE TEST RESETS THE VECTOR AND SP AND CONTINUES TO THE NEXT TEST IN THE PROGRAM.

ERRORS:

- 1.) ERROR 13 IS CALLED TO REPORT RCVR INTR FAULTS
- 2.) ERROR 12 IS CALLED TO REPORT XMITTR INTR FAULTS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

- 1.) IF NO RCVR INTR OCCURRED THE PROBLEM IS EITHER SECTION 'A' OF THE M7821 OR THE M7289 - SH6.
- 2.) IF A FALSE XMIT INTR OCCURRED THE PROBLEM IS MOST LIKELY THE M7821 GENERATING AN INCORRECT VECTOR ADDRESS.

KEY LOGIC:

M7289 SH6 E31-12
M7821 SEC 'A' BUS A BR L U2
BUS SACK L T2
BUS BG IN H B1
'A' MASTER L N1
VECTOR BIT 02 H D2

NOTE: REMEMBER THAT PROBLEMS IN THIS AREA COULD BE CAUSED BY ANY DEVICE IN THE SYSTEM INCLUDING THE 'CPU'. SYSTEM RE-CONFIGURATION TO ISOLATE THE FAULTY SUB-SYSTEM MAY BE REQUIRED.

MOV #1\$, \$LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;MAKE IT REGADR TOO

007054 012737 007112 001110
007062 010102

```
4868 007064 013703 027304      MOV      DHVCT,R3          ;GET FIRST VECTOR ADDR
4869 007070 012723 007222      MOV      #3$, (R3)+       ;GO TO 3$ IF RCVR INTR
4870 007074 113723 030316      MOVB     DHRLVL, (R3)+     ;
4871 007100 105723              TSTB     (R3)+            ;UPDATE POINTER
4872 007102 012723 007176      MOV      #2$, (R3)+       ;GO TO 3$ ON XMITTR INTR
4873 007106 113713 030317      MOVB     DHTLVL, (R3)     ;
4874 007112 012711 004000      MOV      #BIT11, (R1)     ;CLR THE DH11
4875 007116 012706 001100      MOV      #STACK, SP      ;RESET THE SP FOR ERROR LOOPS
4876 007122 004737 027164      JSR      PC, CHPS2        ;GO LOCK OUT INTR
4877 007126 012711 001000      MOV      #BIT09, (R1)     ;SET MAINT MODE BIT
4878 007132 052711 000100      BIS      #BIT06, (R1)     ;SET CHAR AVAILABLE I.E. BIT
4879 007136 052711 000200      BIS      #BIT07, (R1)     ;SET THE CHAR AVAIL BIT TO FORCE INTR
4880 007142 004737 027150      JSR      PC, CHPS1        ;GO CLEAR PSW
4881 007146 000240              NOP                       ;GIVE IT A LITTLE TIME
4882
4883 007150 004737 027200      JSR      PC, SAPS         ;SAVE THE ERROR PSW
4884 007154 011103              MOV      (R1), R3         ;GET THE WAS DATA
4885 007156 005011              CLR      (R1)            ;CLEAR OUT THE SCR
4886 007160 005011              CLR      (R1)
4887 007162 012704 001300      MOV      #1300, R4        ;SET UP S/B DATA
4888 007166 004737 024416      JSR      PC, SUER2A       ;GO SET UP ERROR INFO
4889 007172 104013              ERROR    13              ;TIMED OUT AWAITING CHAR AVAIL INTR
4890 007174 000412              BR       3$              ;GO EXIT TEST
4891
4892 007176 004737 027200      JSR      PC, SAPS         ;SAVE THE ERROR PSW
4893 007202 011103              MOV      (R1), R3         ;GET WAS DATA
4894 007204 012704 001300      MOV      #1300, R4        ;SET UP S/B DATA
4895 007210 005011              CLR      (R1)            ;CLR OUT SCR REG
4896 007212 005011              CLR      (R1)
4897 007214 004737 024416      JSR      PC, SUER2A       ;GO SET UP ERROR INFO
4898 007220 104012              ERROR    12              ;UNEXPECTED XMITTR INTR
4899
4900 007222 012706 001100      MOV      #STACK, SP      ;RESET THE SP
4901 007226 004737 026770      JSR      PC, RESTRP       ;GO RESTORE TRAP CATCHER
```

4902
4903
4904
4905 007232 000004
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946 007234 012737 007272 001110
4947 007242 010102
4948 007244 013703 027304
4949 007250 012723 007402
4950 007254 113723 030316
4951 007260 105723
4952 007262 012723 007356
4953 007266 113713 030317
4954 007272 012711 004000
4955 007276 012706 001100
4956 007302 004737 027164
4957 007306 012711 001000

```
*****  
: *TEST 27 TEST SILO OVFLW. I.E. WITH INTR. CONDITION ACTIVE  
: *****  
TST27: SCOPE  
.REM X  
TEST ABSTRACT  
*****
```

THIS TEST USES MAINT. MODE (SCR09=1) TO ENABLE SILO FULL INTERRUPT. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP XMIT AND RCVR VECTORS
2. RESET THE DH11 AND S.P. - THE LOCK OUT INTR.
3. PRIME DH11 TO GENERATE SILO FULL INTR. - ALLOW INTR.
4. REPORT ERROR IF NO RCVR. INTR OCCURS OR A FALSE XMIT INTR. DOES OCCUR
5. AFTER REPORTING ANY ERRORS DETECTED RESET THE SP AND VECTORS THEN GO TO TEST 30

ERRORS:

1. ERROR 43 IS CALLED TO REPORT NO RCVR INTR OCCURRED
2. ERROR 12 IS CALLED TO REPORT FALSE TRANSMITTER INTR.

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF THE RECEIVER INTR FAILED TO INTERRUPT PROBLEM IS MOST LIKELY THE M7289 MODULE
2. IF A FALSE XMITTR INTR OCCURRED PROBLEM IS MOST LIKELY THE M7281 MODULE

KEY LOGIC:

M7289 SHG E35,E50, OR E31
SCR 14 H (STORAGE) DS1

M7821 'B' SECTION

```
X  
MOV #1$,SLPERR ;SET UP THE ERROR LOOP RETURN  
MOV R1,R2 ;MAKE IT REGADR TOO  
MOV DHVCT,R3 ;GET FIRST VECTOR ADDR  
MOV #3$, (R3)+ ;GO TO 3$ IF RCVR INTR  
MOVB DHRLVL, (R3)+  
TSTB (R3)+ ;UPDATE POINTER  
MOV #2$, (R3)+ ;GO TO 2$ ON XMITTR INTR  
MOVB DHTLVL, (R3)  
1$: MOV #BIT11, (R1) ;CLR THE DH11  
MOV #STACK, SP ;RESET THE SP FOR ERROR LOOPS  
JSR PC, CHPS2 ;GO LOCK OUT INTR  
MOV #BIT09, (R1) ;SET MAINT MODE BIT
```

4958	007312	052711	040000		BIS	#BIT14,(R1)	;SET SILO OVFLW I.E. BIT
4959	007316	052711	010000		BIS	#BIT12,(R1)	;SET THE SILO FULL BIT TO FORCE INTR
4960	007322	004737	027150		JSR	PC,CHPS1	;GO CLEAR PSW
4961	007326	000240			NOP		;GIVE IT A LITTLE TIME
4962							
4963	007330	004737	027200		JSR	PC,SAPS	;SAVE THE ERROR PSW
4964	007334	011103			MOV	(R1),R3	;GET THE WAS DATA
4965	007336	005011			CLR	(R1)	;CLEAR OUT THE SCR
4966	007340	005011			CLR	(R1)	
4967	007342	012704	051000		MOV	#51000,R4	;SET UP S/B DATA
4968	007346	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
4969	007352	104043			ERROR	43	;TIMED OUT AWAITING SILO OVFLW INTR
4970	007354	000412			BR	3\$;GO EXIT TEST
4971							
4972	007356	004737	027200	2\$:	JSR	PC,SAPS	;SAVE THE ERROR PSW
4973	007362	011103			MOV	(R1),R3	;GET WAS DATA
4974	007364	012704	051000		MOV	#51000,R4	;SET UP S/B DATA
4975	007370	005011			CLR	(R1)	;CLR OUT SCR REG
4976	007372	005011			CLR	(R1)	
4977	007374	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
4978	007400	104012			ERROR	12	;UNEXPECTED XMITTR INTR
4979							
4980	007402	012706	001100	3\$:	MOV	#STACK,SP	;RESET THE SP
4981	007406	004737	026770		JSR	PC,RESTRP	;GO RESTORE TRAP CATCHER

4982
4983
4984
4985 007412 000004
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026 007414 012737 007452 001110
5027 007422 010102
5028 007424 013703 027304
5029 007430 012723 007536
5030 007434 113723 030316
5031 007440 105723
5032 007442 012723 007562
5033 007446 113713 030317
5034 007452 012711 004000
5035 007456 012706 001100
5036 007462 004737 027164
5037 007466 012711 001000

: *TEST 30 TEST NON EX MEM I.E. WITH INTR. CONDITION ACTIVE
: *****
TST30. SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE NON-EX-MEM BIT (SCR10) CAN CAUSE A TRANSMITTER INTERRUPT VIA THE PROPER VECTOR. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP XMIT AND RCVR VECTORS
2. CLEAR THE DH11, RESET SP, AND LOCK OUT INTRS
3. PRIME DH11 TO GENERATE YMIT INTR IN MAINT. MODE
4. ALLOW INTRS.
5. REPORT ERROR IF NO XMIT INTR OCCURS OR IF A FALSE RCVR INTR OCCURS
6. REST SP AND VECTORS THEN GO TO TEST 31

ERRORS:

1. ERROR 44 IS CALLED IF NON-EX-MEM FAILS TO GENERATE XMIT INTR
2. ERROR 11 IS CALLED IF FALSE RCVR INTR OCCURS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF THE NON-EX-MEM INTERRUPT FAILS TO OCCUR PROBLEM IS MOST LIKELY THE M7289 MODULE
2. IF A FALSE RCVR INTR OCCURS PROBLEM IS MOST LIKELY THE M7289 OR THE M7281 MODULES.

KEY LOGIC:

M7289 SH6 SCR 10 H (NO EX MEM) FL1
E35, E41, OR E48

X

```
MOV #1$, $LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;MAKE IT REGADR TOO
MOV DHVCT,R3 ;GET FIRST VECTOR ADDR
MOV #2$, (R3)+ ;GO TO 2$ IF RCVR INTRS
MOVB DHRLVL, (R3)+
TSTB (R3)+ ;UPDATE POINTER
MOV #3$, (R3)+ ;GO TO 3$ ON XMITTR INTRS
MOVB DHTLVL, (R3)
1$: MOV #BIT11, (R1) ;CLR THE DH11
MOV #STACK, SP ;RESET THE SP FOR ERROR LOOPS
JSR PC, CHPS2 ;GO LOCK OUT INTRS
MOV #BIT09, (R1) ;SET MAINT MODE BIT
```


5038	007472	052711	020000		BIS	#BIT13,(R1)	;SET XMITTR I.E. BIT
5039	007476	052711	002000		BIS	#BIT10,(R1)	;SET THE NON EX MEM BIT TO FORCE INTR
5040	007502	004737	027150		JSR	PC,CHPS1	;GO CLEAR PSW
5041	007506	000240			NOP		;GIVE IT A LITTLE TIME
5042							
5043	007510	004737	027200		JSP	PC,SAPS	;SAVF THE ERROR PSW
5044	007514	011103			MOV	(R1),R3	;GET THE WAS DATA
5045	007516	005011			CLR	(R1)	;CLEAR OUT THE SCR
5046	007520	005011			CLR	(R1)	
5047	007522	012704	023000		MOV	#23000,R4	;SET UP S/B DATA
5048	007526	004737	024416		JSR	PC,SUFR2A	;GO SET UP ERROR INFO
5049	007532	104044			ERROR	44	;TIMED OUT AWAITING NON EX MEM INTR
5050	007534	000412			BR	3\$;GO EXIT TEST
5051							
5052	007536	004737	027200	2\$:	JSR	PC,SAPS	;SAVE THE ERROR PSW
5053	007542	011103			MOV	(R1),R3	;GET WAS DATA
5054	007544	012704	023000		MOV	#23000,R4	;SET UP S/B DATA
5055	007550	005011			CLR	(R1)	;CLR OUT SCR REG
5056	007552	005011			CLR	(R1)	
5057	007554	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
5058	007560	104011			ERROR	11	;UNEXPECTED RCVR INTR
5059							
5060	007562	012706	001100	3\$:	MOV	#STACK,SP	;RESET THE SP
5061	007566	004737	026770		JSR	PC,RESTRP	;GO RESTORE TRAP CATCHER

5062
5063
5064
5065 007572 000004
5066
5067
5068
5069
5070
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080

: *TEST 31 TEST XMITTR DONE I.E. WITH INTR. CONDITION ACTIVE
: *****

TST31: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT XMIT DONE (SCR15) CAN BE SET 'N MAINT.
MODE TO CAUSE A XMITTR INTR VIA THE PROPER VECTOR. THE TEST SEQUENCE
IS AS FOLLOWS:

1. SET UP XMIT AND RCVR VECTORS
2. CLEAR THE DH11, RESET SP, AND LOCK OUT INTRS
3. PRIME DH11 TO GENERATE 'XMIT DONE' INTR
4. CLEAR PSW TO ALLOW INTRS
5. REPOT ERROR IF XMITTR FAILS TO INTR OR A FALSE RCVR INTR OCCURS

ERRORS:

1. ERROR 45 IS CALLED TO REPORT 'XMIT DONE' INTR FAILED TO OCCUR
2. ERROR 11 IS CALLED TO REPORT FALSE RCVR INTRS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF NO XMIT INTR OCCURS PROBLEM IS MOST LIKELY THE M7289 MODULE
2. IF A FALSE RCVR INTR. OCCURS PROBLEM IS MOST LIKELY THE M7821 MODULE.

KEY LOGIC:

M7289 SH6 SCR 15 H (XMIT) FR2
E48, E50

5103 X
5104 007574 012737 007632 001110
5105 007602 010102
5106 007604 013703 027304
5107 007610 012723 007716
5108 007614 113723 030316
5109 007620 105723
5110 007622 012723 007742
5111 007626 113713 030317
5112 007632 012711 00400G
5113 007636 012706 001100
5114 007642 004737 027164
5115 007646 012711 001000
5116 007652 052711 020000
5117 007656 052711 100000

```

X
MOV #1$, $LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1, R2 ;MAKE IT REGADR TOO
MOV DHVCT, R3 ;GET FIRST VECTOR ADDR
MOV #2$, (R3)+ ;GO TO 2$ IF RCVR INTRS
MOV DTRLVL, (R3)+
TSTB (R3)+ ;UPDATE POINTER
MOV #3$, (R3)+ ;GO TO 3$ ON XMITTR INTRS
MOV DHTLVL, (R3)
1$: MOV #BIT11, (R1) ;CLR THE DH11
MOV #STACK, SP ;RESET THE SP FOR ERROR LOOPS
JSR PC, CHPS2 ;GO LOCK OUT INTRS
MOV #BIT09, (R1) ;SET MAINT MODE BIT
BIS #BIT13, (R1) ;SET XMIT DONE I.E. BIT
BIS #BIT15, (R1) ;SET THE XMITTR DONE BIT TO FORCE INTR

```

5118	007662	004737	027150		JSR	PC,CHPS1	:GO CLEAR PSW
5119	007666	000240			NOP		:GIVE IT A LITTLE TIME
5120							
5121	007670	004737	027200		JSR	PC,SAPS	:SAVE THE ERROR PSW
5122	007674	011103			MOV	(R1),R3	:GET THE WAS DATA
5123	007676	005011			CLR	(R1)	:CLEAR OUT THE SCR
5124	007700	005011			CLR	(R1)	
5125	007702	012704	121000		MOV	#121000,R4	:SET UP S/B DATA
5126	007706	004737	024416		JSR	PC,SUER2A	:GO SET UP ERROR INFO
5127	007712	104045			ERROR	45	:TIMED OUT AWAITING XMIT DONE INTR
5128	007714	000412			BR	3\$:GO EXIT TEST
5129							
5130	007716	004737	027200	2\$:	JSR	PC,SAPS	:SAVE THE ERROR PSW
5131	007722	011103			MOV	(R1),R3	:GET WAS DATA
5132	007724	012704	121000		MOV	#121000,R4	:SET UP S/B DATA
5133	007730	005011			CLR	(R1)	:CLR OUT SCR REG
5134	007732	005011			CLR	(R1)	
5135	007734	004737	024416		JSR	PC,SUER2A	:GO SET UP ERROR INFO
5136	007740	104011			ERROR	11	:UNEXPECTED RCVR INTR
5137							
5138	007742	012706	001100	3\$:	MOV	#STACK,SP	:RESET THE SP
5139	007746	004737	026770		JSR	PC,RESTRP	:GO RESTORE TRAP CATCHER

5140
5141
5142
5143 007752 000004
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195

```
*****  
: *TEST 32 BASIC TRANSMITTER "NPR" LOGIC TEST 1  
: *****  
TST32: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST TRANSMITS A SINGLE BYTE FROM LOCATION 0 ON ALL SELECTED LINES (AS SELECTED BY THE CONFIGURATION PARAMETER "LINSEL:") ONE AT A TIME. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE XMITTR VECTOR
2. SELECT A LINE # TO TEST
3. RESET THE SP
4. CLEAR ALL LOCATIONS IN "CAR" AND "BCR" MEMORIES
5. LOCK OUT INTRs AND CLEAR THE DH11
6. PRIME DH11 TO XMIT ONE CHAR FROM LOCATION 0
9600 BAUD, 5-BITS, 1 STOP BIT
7. ACTIVATE SELECTED XMITTR AND ENABLE XMIT DONE INTR
8. CLEAR PSW TO ALLOW INTR.
9. ACTIVATE TIMER TO WAIT FOR XMIT DONE INTR
10. IF TIMEOUT OCCURS REPORT ERROR AND RESTART AT STEP 2
11. IF XMIT INTERRUPT OCCURS CHECK THE FOLLOWING CONDITIONS AND REPORT ANY ERRORS:
 - A. XMIT DONE (SCR15=1) SET
 - B. "BAR" BIT GOT CLEARED
 - C. "CAR" REGISTER GOT INCREMENTED TO +1
 - D. "BCR" REGISTER GO INCREMENTED TO 0
12. REPEAT STEP 2 THRU 11 UNTIL ALL SELECTED LINES TESTED
13. AFTER TESTING ALL LINES CLEAR THE DH11, RESET THE VECTOR CLEAR PSW, RESET SP, AND GO TO TEST 33.

ERRORS:

1. ERROR 15 IS CALLED IF XMIT DONE FAILS TO INTR ON TIME
2. ERROR 14 IS CALLED IF XMIT DONE NOT SET
3. ERROR 14 IS CALLED IF "BAR" BIT FAILED TO CLEAR
4. ERROR 14 IS CALLED IF "CAR" NOT INCREMENTED PROPERLY
5. ERROR 14 IS CALLED IF "BCR" NOT INCREMENTED PROPERLY

ALL ERROR MESSAGE HEADERS INCLUDE THE LINE NO. OF THE FAILING LINE.

SYNC: M7277 SH3 INIT A H EF2

(NOTE: USE SR09=1 TO LOCK ON FAILING LINE AND SR13=1 TO INHIBIT ERROR PRINTOUT TO MINIMIZE SCOPE LOOP.)

DEBUG:

1. IF ALL LINES FAIL TO INTERRUPT ON TIME, SUSPECT LOSS OF 9600 BAUD

5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251

- CLOCK SIGNAL OR BYTE COUNT DECODER ON M7278 SH3 (XMIT FINISHED PULSE L)
- 2. IF GROUP OF 8 LINES <15:08> OR <07:00> FAIL TO INTERRUPT ON TIME, SUSPECT THE BUFFERED CLOCK SIGNALS (TOP AND BOT) ON THE M7288 SH3
- 3. IF ONLY ONE LINE FAILS TO INTERRUPT ON TIME, SUSPECT LOSS OF CLOCK IN LINE MULTIPLEXORS M7288 SH4-SH11.
- 4. IF LINE INTERRUPTED OK BUT 'SCR', 'BAR', 'CAR', OR 'BCR' WAS INCORRECT REFER TO KEY LOGIC SIGNALS BELOW.

KEY LOGIC:

XMIT DONE FAILED TO SET:

```

M7289 SH6 SCR15 H FR2
M7278 SH3 XMIT FINISHED PULSE L AR1

```

'BAR' BIT FAILED TO CLEAR:

```

M7278 SH3 CLR BAR <15:00> L
SH5 BAR <15:12> H E54, E55 (7474)
SH6 BAR <11:08> H E62, E63 (7474)
SH7 BAR <07:04> H E70, E71 (7474)
SH8 BAR <03:00> H E78, E79 (7474)

```

'CAR' REG NOT INCREMENTED:

```

M7277 SH6 END CYCLE PULSE DLY H FL2
SH5 AND SH6 'CAR' MEMORY LOGIC
M796 END CYCLE H P2

```

'BCR' REG NOT INCREMENTED:

```

M796 END CYCLE L N2
M7278 SH3 AND SH4 'BCR' MEMORY LOGIC

```

X

```

MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN
MOV DHVCT, R3 ;GET THE FIRST VECTOR ADDRESS
ADD #4, R3 ;POINT TO XMITTR ENTRY
MOV #4$, (R3)+ ;GO TO 4$ ON XMITTR INTR
MOVB DHTLVL, (R3)
1$: JSR PC, SELINE ;GO SELECT A LINE NO. TO TEST
BR 8$ ;BR IF TESTED ALL SELECTED LINES

2$: MOV #STACK, SP ;RESET SP FOR ERROR LOOPS
MOV R1, R2 ;SET UP REGADR
MOV #120000, R4 ;SET UP S/B DATA
BISB LINE, R4
JSR PC, CLCABC ;GO CLEAR CAR AND BCR MEMORIES
JSR PC, CHPS2 ;GO LOCK OUT INTR
MOV #BIT11, (R1) ;CLEAR THE DH11 INTERFACE
BISB LINE, (R1) ;SELECT A LINE NO.
MOV #-1, BCR(R1) ;SET BYTE COUNT TO -1
MOV #33500, LPR(R1) ;SET UP LINE PARAMETERS
BIS LINMSK, BAR(R1) ;ACTIVATE SELECTED LINE

```

```

001110
000004
010154
030317
024544
001100
010102
120000
030322
024716
027164
004000
030322
000010
000004
000012

```

```

007754 012737 010010
007762 013703 027304
007766 062703 000004
007772 012723 010154
007776 113713 030317
010002 004737 024544
010006 000552
010010 012706 001100
010014 010102
010016 012704 120000
010022 153704 030322
010026 004737 024716
010032 004737 027164
010036 012711 004000
010042 153711 030322
010046 012761 177777 000010
010054 012761 033500 000004
010062 053761 027314 000012

```

CZDMM-D-0
CZDHMD.P11MACY11 30A(1052) 10-MAR-78 08:05 PAGE 127
09-MAR-78 15:32 T32

BASIC TRANSMITTER 'NPR' LOGIC TEST 1

SEQ 0125
SEQ 0124

```

5252 010070 052711 020000      BIS      #BIT13,(R1)      ;ENABLE INTERRUPT ON XMIT DONE
5253 010074 004737 027150      JSR      PC,CHPS1      ;GO CLEAR PSW
5254
5255 010100 012737 000001 030350      MOV      #1,TIMEA      ;INIT TIMER A
5256 010106 005037 030352      CLR      TIMEB        ;INIT TIMER B
5257 010112 000240          3$:      NOP                    ;DO NOTHING WAIT
5258 010114 004737 027016      JSR      PC,TIMEIT     ;CALL TIMER
5259 010120 000774          BR      3$             ;TIMER ROUTINE WILL MOVE RETURN PC AROUND
5260
5261
5262 010122 004737 027200      JSR      PC,SAPS      ;SAVE THE ERROR PSW
5263 010126 011103      MOV      (R1),R3      ;GET THE WAS DATA
5264 010130 042703 000200      BIC      #BIT07,R3    ;WE'RE NOT INTERESTED IN THIS BIT
5265 010134 004737 024416      JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5266 010140 004537 024636      JSR      R5,SUNUM     ;GO SET LINE NO. IN ERROR MSG
5267 010144 030322      LINE
5268 010146 031522      EM15+43
5269 010150 104015      ERROR   15           ;TIMEOUT WHILE AWAITING XMIT INTR
5270 010152 000713      BR      1$           ;GO TEST NEXT LINE
5271
5272 010154 005711          4$:      TST      (R1)         ;DID XMIT DONE SET ??
5273 010156 100411      BMI     5$           ;BR IF YES
5274
5275 010160 004737 027200      JSR      PC,SAPS      ;SAVE THE ERROR PSW
5276 010164 011103      MOV      (R1),R3      ;GET THE WAS DATA
5277 010166 004737 024416      JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5278 010172 004737 010372      JSR      PC,9$        ;GO SET UP SOME ERROR STUFF
5279 010176 104014      ERROR   14           ;XMIT DONE FAILED TO SET
5280 010200 000700      BR      1$           ;GO TEST NEXT LINE
5281
5282 010202 016103 000012          5$:      MOV      BAR(R1),R3    ;GET WAS DATA FROM 'BAR'
5283 010206 001413      BEQ     6$           ;BR IF BAR BIT GOT CLEARED
5284
5285 010210 004737 027200      JSR      PC,SAPS      ;SAVE THE ERROR PSW
5286 010214 062702 000012      ADD     #BAR,R2       ;SET UP REGADR
5287 010220 005004      CLR     R4            ;SET UP S/B DATA
5288 010222 004737 024416      JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5289 010226 004737 010372      JSR      PC,9$        ;GO SET UP SOME ERROR STUFF
5290 010232 104014      ERROR   14           ;BAR BIT FAILED TO CLEAR
5291 010234 000662      BR      1$           ;GO TEST NEXT LINE
5292
5293 010236 016103 000006          6$:      MOV      CAR(R1),R3    ;GET THE WAS DATA FROM CAR
5294 010242 022703 000001      CMP     #1,R3        ;DID IT GET INCREMENTED ?
5295 010246 001414      BEQ     7$           ;BR IF YES
5296
5297 010250 004737 027200      JSR      PC,SAPS      ;SAVE THE ERROR PSW
5298 010254 012704 000001      MOV     #1,R4        ;SET UP S/B DATA
5299 010260 062702 000006      ADD     #CAR,R2       ;SET UP REGADR
5300 010264 004737 024416      JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5301 010270 004737 010372      JSR      PC,9$        ;GO SET UP SOME ERROR STUFF
5302 010274 104014      ERROR   14           ;CAR REG NOT INCREMENTED PROPERLY
5303 010276 000641      BR      1$           ;GO TEST NEXT LINE
5304
5305 010300 016103 000010          7$:      MOV      BCR(R1),R3    ;GET WAS DATA FROM BCR
5306 010304 001636      BEQ     1$           ;BR IF BCR GOT INCREMENTED TO 000000
5307

```

CZDMM-D-0
CZDHMG.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
T32

08:05 PAGE 128
BASIC TRANSMITTER 'NPR' LOGIC TEST 1

SEQ 0126
SEQ 0125

5308	010306	004737	027200		JSR	PC,SAPS	;SAVE THE ERROR PSW
5309	010312	005004			CLR	R4	;SET UP S/B DATA
5310	010314	062702	C00010		ADD	#BCR,R2	;SET UP REGADR
5311	010320	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
5312	010324	004737	010372		JSR	PC,9\$;GO SET UP SOME ERROR STUFF
5313	010330	104014			ERROR	14	;BCR REG NOT INCREMENTED PROPERLY
5314	010332	000623			BR	1\$;GO TEST NEXT LINE
5315							
5316	010334	012711	004000	8\$:	MOV	#BIT11,(R1)	;CLEAR THE DH11
5317	010340	013703	027304		MOV	DHVCT,R3	;GET THE VECTOR ADDR
5318	010344	062703	000004		ADD	#4,R3	;POINT TO XMIT VECTOR
5319	010350	010313			MOV	R3,(R3)	;RESTORE TRAP CATCHER
5320	010352	062723	000002		ADD	#2,(R3)+	
5321	010356	005013			CLR	(R3)	
5322	010360	004737	027150		JSR	PC,CHPS1	;GO CLEAR PSW
5323	010364	012706	001100		MOV	#STACK,SP	;RESET THE STACK POINTER
5324	010370	000405			BR	TST33	;GO TO NEXT TEST
5325							
5326	010372	004537	024636	9\$:	JSR	R5,SUNUM	;GO SET UP LINE NO. IN MSG.
5327	010376	030322			LINE		
5328	010400	031454			EM14+44		
5329	010402	000207			RTS	PC	;RETURN TO REPORT ERROR

5330
5331
5332
5333 010404 000004

```
*****
;*TEST 33      TRANSMITTR NPR LOGIC TEST 2
*****
TST33: SCOPE
.REM      %
TEST ABSTRACT:
*****
```

5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379

THIS TEST IS SIMILAR TO TEST 32 EXCEPT THAT ALL LOCATIONS IN THE "BCR" AND "CAR" MEMORIES ARE TESTED TO VERIFY THAT TRANSMISSION ON THE SELECTED LINE DID NOT DISTURB ANY UNSELECTED LOCATIONS IN THE MEMORIES. IF ALSO OPERATES IN "FLAG" MODE RATHER THAN USING INTERRUPTS. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A LINE # TO TEST (AS DEFINED BY "LINSEL:")
2. CLEAR BOTH THE "CAR" AND "BCR" MEMORIES
3. LOAD THE "BCR" MEMORY WITH ALL ONES (BYTE COUNT = -1)
4. ACTIVATE THE XMITTER ON THE SELECTED LINE
5. ACTIVATE TIMER TO WAIT FOR "XMIT DONE"
6. IF "XMIT DONE" FAILS TO SET ON TIME - REPORT ERROR AND REPEAT 1 THRU 5 UNTIL ALL SELECTED LINES TESTED
7. IF "XMIT DONE" SETS CHECK ALL LOCATIONS IN THE "BCR" MEMORY REPORT ANY UNSELECTED LINES NOT CONTAINING -1 AND THE SELECTED LINE IF IT DOES NOT CONTAIN 0
8. CHECK ALL LOCATIONS IN THE "CAR" MEMORY AND REPORT ANY UNSELECTED LOCATIONS NOT CONTAINING 0 AND THE SELECTED LINE IF IT DOES NOT CONTAIN +1.
9. REPEAT STEPS 1 THRU 8 UNTIL ALL SELECTED LINES TESTED.

ERRORS:

1. ERROR 50 CALLED IF XMIT DONE TIMEOUT ERROR DETECTED.
2. ERROR 51 CALLED IF "BCR" MEMORY ERROR DETECTED
3. ERROR 51 CALLED IF "CAR" MEMORY ERROR DETECTED

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. ASSUMING TEST 32 RAN ERROR FREE THE PROBLEM IS MOST LIKELY THE:
M7278 MODULE IF "BCR" ERRORS
M7277 MODULE IF "CAR" ERRORS

KEY LOGIC: (SAME AS TEST 32)

5380 010406 012737 010422 001110
5381 010414 004737 024544
5382 010420 000544
5383 010422 052711 004000
5384 010426 004737 024716
5385 010432 004737 024760

```
%
1$: MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN
JSR PC, SELINE ;GO SELECT A LINE TO TEST
BR TST34 ;:BR IF DONE ALL SELECTED LINES
2$: BIS #BIT11, (R1) ;CLEAR THE DH11
JSR PC, CLCABC ;GO CLEAR "CAR" AND "BCR" MEMORIES
JSR PC, LDBCR ;GO LOAD "BCR" MEMORY WITH ALL ONES
```


5386	010436	153711	030322		BISB	LINE,(R1)	;SELECT THE LINE
5387	010442	012761	033500	000004	MOV	#33500,LPR(R1)	;SET UP PARAMETERS
5388	010450	013761	027314	000012	MOV	LINMSK,BAR(R1)	;ACTIVATE XMIT ON SELECTED LINE
5389							
5390	010456	012737	000001	030350	MOV	#1,TIMEA	;INIT TIMER A
5391	010464	005037	030352		CLR	TIMEB	;INIT TIMER B
5392	010470	005711			TST	(R1)	;XMITTR DONE YET
5393	010472	100423			BMI	4\$;BR IF YES
5394	010474	004737	027016		JSR	PC,TIMEIT	;CALL THE TIMER
5395	010500	000773			BR	3\$;TIMER ROUTINE WILL MOVE RETURN PC ;AROUND THIS BRANCH IF TIME OUT OCCURS
5396							
5397							
5398	010502	004737	027200		JSR	PC,SAPS	;SAVE THE ERROR PSW
5399	010506	011103			MOV	(R1),R3	;GET THE WAS DATA
5400	010510	012704	100000		MOV	#BIT15,R4	;SET UP S/B DATA
5401	010514	153704	030322		BISB	LINE,R4	
5402	010520	010102			MOV	R1,R2	;MAKE REGADR = DEVADR
5403	010522	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
5404	010526	004537	024636		JSR	R5,SUNUM	;SET LINE NO. IN MSG
5405	010532	030322			LINE		
5406	010534	034254			EM50+53		
5407	010536	104050			ERROR	50	;TIMED OUT AWAITING XMIT DONE ON SEL LINE
5408	010540	000725			BR	1\$;GO TRY THE NEXT LINE
5409							
5410	010542	005037	001220		CLR	\$TMP7	;INIT A LINE COUNTER
5411	010546	113711	001220		MOV	\$TMP7,(R1)	;SELECT LINE NO. IN "SCR"
5412	010552	012704	177777		MOV	#-1,R4	;SET UP S/B DATA
5413	010556	016103	000010		MOV	BCR(R1),R3	;GET THE WAS BYTE COUNT
5414	010562	123737	030322	001220	CMPB	LINE,\$TMP7	;WAS THIS THE ACTIVE LINE ??
5415	010570	001001			BNE	6\$;BR IF NOT
5416	010572	005004			CLR	R4	;CHANGE S/B DATA TO 000000
5417	010574	020304			CMP	R3,R4	;WAS BYTE COUNT CORRECT ??
5418	010576	001416			BEQ	7\$;BR IF YES
5419							
5420	010600	005037	001202		CLR	\$TMP0	;SAVE THE ACTIVE LINE NO.
5421	010604	113737	030322	001202	MOV	LINE,\$TMP0	
5422	010612	013737	001220	001204	MOV	\$TMP7,\$TMP1	;SAVE THE LINE NO. BEING CHECKED
5423	010620	010102			MOV	R1,R2	;SET UP REGADR = BCR REG ADDR
5424	010622	062702	000010		ADD	#BCR,R2	
5425	010626	004737	024500		JSR	PC,SUER4	;GO SET UP ERROR INFO
5426	010632	104051			ERROR	51	;BYTE COUNT INCORRECT
5427							
5428	010634	005004			CLR	R4	;SET UP S/B DATA
5429	010636	016103	000006		MOV	CAR(R1),R3	;GET THE WAS DATA
5430	010642	123737	030322	001220	CMPB	LINE,\$TMP7	;IS THIS THE ACTIVE LINE
5431	010650	001001			BNE	8\$;BR IF NOT
5432	010652	005204			INC	R4	;BUMP THE CAR ADDRESS FOR ACTIVE LINE
5433	010654	020304			CMP	R3,R4	;CAR CONTENTS CORRECT ??
5434	010656	001416			BEQ	9\$;BR IF YES
5435							
5436	010660	005037	001202		CLR	\$TMP0	;SET UP ACT LINE NO.
5437	010664	113737	030322	001202	MOV	LINE,\$TMP0	
5438	010672	013737	001220	001204	MOV	\$TMP7,\$TMP1	;SAVE THE LINE NO. BEING CHECKED
5439	010700	010102			MOV	R1,R2	;SET UP REGADR
5440	010702	062702	000006		ADD	#CAR,R2	
5441	010706	004737	024500		JSR	PC,SUER4	;SET UP THE ERROR INFO

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052) 10-MAR-78
09-MAR-78 15:32 T33

08:05 PAGE 131
TRANSMITTR NPR LOGIC TEST 2

L 10

SEQ 0129
SEQ 0128

5442	010712	104051				ERROR	51		;CAR REG INCORRECT
5443									
5444	010714	005237	001220		9\$:	INC	\$TMP7		;GENERATE NEW LINE NO.
5445	010720	022737	000020	001220		CMP	#20,\$TMP7		;TESTED ALL LINES
5446	010726	001707				BEQ	5\$;BR IF NOT
5447	010730	000631				BR	1\$;GO SELECT NEXT ACTIVE LINE

5448
5449
5450
5451 010732 000004
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503

: *TEST 34 TEST THAT CHARACTER AVAILABLE CAN CAUSE RCVR INTERRUPT
: *****

TST34: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT WHEN "CHAR AVAIL" (BIT07 IN "SCR") SETS AS A RESULT OF XMITTING AND RECEIVING ONE CHARACTER (USING SILO MAINT MODE) IT CAUSES A RCVR INTR VIA THE PROPER VECTOR. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE RCVR VECTOR
2. LOCK OUT INTRS, RESET SP, AND CLEAR DH11
3. USE MAINT MODE TO LOAD A CHAR INTO THE SILO (DATA=125252)
4. CLEAR PSW TO ALLOW INTERRUPTS
5. ACTIVATE TIMER TO WAIT FOR INTR TO OCCUR
6. IF NO RCVR INTR OCCURS REPORT ERROR AND GO TO STEP 9
7. WHEN RCVR INTRS - CHECK SILO DATA FOR 125252 - IF NOT CORRECT REPORT ERROR AND GO TO STEP 9
8. CHECK THAT SILO FILL LEVEL=1 - IF NOT REPORT ERROR
9. RESET SP, CLEAR PSW, RESET VECTOR, AND GO TO TEST 35

ERRORS:

1. ERROR 13 IS CALLED TO REPORT RCVR TIMEOUT ERROR
2. ERROR 52 IS CALLED TO REPORT SILO DATA INCORRECT
3. ERROR 6 IS CALLED TO REPORT INCORRECT SILO FILL COUNT

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF NOT RCVR INTR OCCURS SUSPECT THE M7277, M7281, OR M7279 MODULES
2. IF SILO DATA OR FILL-ERRORS SUSPECT THE M7279 MODULE

KEY LOGIC:

M7279	SH1	SILO DATA MUX'S (74157'S) SSR15 H	CR1
	SH2	DATA READY L NRC 15 H SILO MEMORY LOAD SILO L 5.068 MHZ (CLOCK) SSR <13:00> H	DV1 DL1 (E13,E17,E8,E3) 3341'S DJ2 DN1 (E20,E24)
M7289	SH6	RCV INT REQ H	DP1
M7278	SH8	SSR <03:00> H	

SH7 SSR <07:00> H

```

5504
5505
5506
5507
5508 010734 012737 010756 001110      MOV      #1$,SLPERR      ;SET UP THE ERROR LOOP RETURN
5509 010742 013703 027304                MOV      DHVCT,R3      ;GET FIRST VECTOR ADDR
5510 010746 012723 011042                MOV      #3$, (R3)+    ;GO TO 3$ ON RCVR INTERRUPT
5511 010752 113713 030316                MOVVB   DHRLVL, (R3)
5512 010756 004737 027164      1$:     JSR      PC,CHPS2      ;GO LOCK OUT INTRs
5513 010762 012706 001100                MOV      #STACK,SP    ;RESET SP FOR ERROR LOOPS
5514 010766 012711 004000                MOV      #BIT11, (R1)  ;CLEAR THE DH11
5515 010772 052761 100000 000016      BIS      #BIT15,SSR(R1) ;SET SILO MAINT. BIT TO LOAD SILO
5516 011000 012711 000100                MOV      #BIT06, (R1) ;ENABLE CHAR. AVAIL INTERRUPT
5517 011004 004737 027150                JSR      PC,CHPS1      ;GO CLEAR PSW
5518 011010 012703 001000                MOV      #1000,R3     ;INIT TIMER
5519 011014 005303                2$:     DEC      R3           ;DEC TIMER
5520 011016 001376                BNE     2$           ;BR IF NO TIMEOUT
5521
5522 011020 004737 027200                JSR      PC,SAPS      ;SAVE THE ERROR PSW
5523 011024 011103                MOV      (R1),R3      ;GET THE WAS DATA
5524 011026 012704 000300                MOV      #300,R4     ;SET UP S/B DATA
5525 011032 004737 024416                JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5526 011036 104013                ERROR   13           ;CHAR AVAIL FAILED TO SET ON TIME
5527 011040 000436                BR      5$           ;ESCAPE FROM THIS TEST - CATASTROPHIC ERROR
5528
5529 011042 016105 000016      3$:     MOV      SSR(R1),R5    ;SAVE THE SILO STATUS REG.
5530 011046 016103 000002                MOV      NRC(R1),R3   ;GET THE WAS DATA
5531 011052 012704 125252                MOV      #125252,R4   ;SET UP S/B DATA
5532 011056 020304                CMP     R3,R4         ;WAS = S/B = 125252 ??
5533 011060 001410                BEQ     4$           ;BR IF IT IS
5534
5535 011062 004737 027200                JSR      PC,SAPS      ;SAVE THE ERROR PSW
5536 011066 062702 000002                ADD     #NRC,R2       ;SET UP REGADR
5537 011072 004737 024416                JSR      PC,SUER2A    ;GO SET UP ERROR INFO
5538 011076 104052                ERROR   52           ;DATA COMPARE ERROR
5539 011100 000416                BR      5$           ;GET OUT
5540
5541 011102 010503      4$:     MOV      R5,R3         ;NOW GET THW SILO STATUS REG AGAIN
5542 011104 042703 140377                BIC     #140377,R3    ;CLR OUT JUNK
5543 011110 012704 000400                MOV     #400,R4       ;SET UP S/B DATA
5544 011114 020304                CMP     R3,R4         ;SSR CHAR COUNT = 1 ??
5545 011116 001407                BEQ     5$           ;BR IF IT IS
5546
5547 011120 004737 027200                JSR      PC,SAPS      ;SAVE THE ERROR PSW
5548 011124 062702 000016                ADD     #SSR,R2       ;SET UP REGADR
5549 011130 004737 024416                JSR      PC,SUER2A    ;SET UP ERROR INFO
5550 011134 104006                ERROR   6            ;SSR COUNT NOT CORRECT
5551
5552 011136 012706 001100      5$:     MOV      #STACK,SP    ;RESET THE STACK POINTER
5553 011142 004737 027150                JSR      PC,CHPS1     ;GO CLEAR PSW
5554 011146 005011                CLR     (R1)          ;RESET I.E. BIT
5555 011150 013703 027304                MOV     DHVCT,R3     ;GET FIRST VECTOR ADDR
5556 011154 010313                MOV     R3,(R3)      ;RESTORE TRAP CATCHER
5557 011156 062723 000002                ADD     #2,(R3)+
5558 011162 005013                CLR     (R3)
5559

```

5560
5561
5562
5563 011164 000004
5564
5565
5566
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603 011166 010102
5604 011170 062702 000016
5605 011174 012737 000001 001220
5606 011202 012711 004000
5607 011206 013705 001220
5608 011212 005004
5609 011214 052712 100000
5610 011220 012703 001000
5611 011224 005303
5612 011226 001376
5613 011230 042712 100000
5614 011234 005204
5615 011236 005305

*TEST 35 TEST THAT THE SILO STATUS REG COUNTS UP CORRECTLY

TST35: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE SILO FILL LEVEL COUNTS UP CORRECTLY
WHEN ALL COUNTS (0-77) ARE TESTED BY LOADING THE SILO USING
MAINT MODE. THE TEST SEQUENCE IS AS FOLLOWS:

1. INIT "\$TMP7" TO START WITH A COUNT=01
2. CLEAR THE DH11
3. LOAD THE SILO WITH 125252'S IN MAINT MODE
UNTIL # OF WORDS INDICATED BY THE COUNT ARE LOADED
4. AFTER LOADING REQUIRED COUNT CHECK THAT FILL LEVEL
BITS (SSR<13:08>) EQUAL COUNT - IF NOT REPORT ERROR
5. INCREMENT COUNT IN "\$TMP7" AND REPEAT 1 THRU 4
UNTIL ALL COUNTS (01-77) HAVE BEEN TESED

ERRORS:

1. ERROR 6 IS CALLED TO REPORT SILO FILL LEVEL ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. FAILURES IN THIS TEST MOST LIKELY INDICATE A BAD M7279 MODULE

KEY LOGIC:

M7279 SH2 SSR <13:08>
LOAD SILO L DJ2
5.068 MHZ (CLOCK) DN1
DATA READY L DV1

%
MOV R1,R2 ;MAKE REGADR = SSR
ADD #SSR,R2
MOV #1,\$TMP7 ;START WITH COUNT OF 1
1\$: MOV #BIT11,(R1) ;CLEAR THE DH11
MOV \$TMP7,R5 ;SAVE CHARACTER COUNT BEING TESTED
CLR R4 ;INIT A CHAR COUNTER
2\$: BIS #BIT15,(R2) ;SET THE SILO MAINT BIT
MOV #1000,R3 ;INIT A TIMER
3\$: DEC R3 ;STALL TO ALLOW TIME TO LOAD SILO
BNE 3\$
BIC #BIT15,(R2) ;CLEAR SILO MAINT. BIT
INC R4 ;COUNT A CHAR LOADED
DEC R5 ;DECREMENT TEST COUNT

```
5616 011240 001365          BNE      2$          ;BR UNTIL WE'VE LOADED THE TEST COUNT
5617
5618 011242 011203          MOV      (R2),R3     ;SET THE WAS COUNT
5619 011244 042703 140377    BIC      #140377,R3  ;CLR JUNK BITS
5620 011250 000304          SWAB    R4          ;SET UP S/B DATA
5621 011252 020304          CMP     R3,R4       ;TEST COUNT = SILO COUNTER ?
5622 011254 001406          BEQ     4$          ;BR IF YES
5623
5624 011256 004737 024412    JSR     PC,SUER2     ;GO SET UP ERROR INFO
5625 011262 012737 011202 001110 MOV     #1$,$LPERR  ;SET UP ERROR LOOP RETURN
5626 011270 104006          ERROR   6          ;SSR FAILED TO UP-COUNT CORRECTLY
5627
5628 011272 005237 001220    4$:    INC     $TMP7      ;INCREMENT TO NEXT COUNT TO TEST
5629 011276 022737 000100 001220 CMP     #100,$TMP7  ;MAXIMUM COUNT ??
5630 011304 001336          BNE     1$          ;BR IF NOT
5631
```

5632
5633
5634
5635 011306 000004
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666 011310 010102
5667 011312 062702 000016
5668 011316 012737 000001 001220
5669 011324 012711 004000
5670 011330 012705 000100
5671 011334 163705 001220
5672 011340 012703 000100
5673 011344 012704 001000
5674 011350 052712 100000
5675 011354 005304
5676 011356 001376
5677 011360 042712 100000
5678 011364 005303
5679 011366 001366
5680
5681 011370 013703 001220
5682 011374 012704 001000
5683 011400 005761 000002
5684 011404 005304
5685 011406 001376
5686 011410 005303
5687 011412 001370

*TEST 36 TEST THAT SILO STATUS REGISTER DOWN COUNTS CORRECTLY

TST36: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE SILO FILL LEVEL COUNTS DOWN PROPERLY
WHEN WORDS ARE READ FROM THE SILO. ALL COUNTS FROM 77-00 ARE
TESTED. THE TEST SEQUENCE IS AS FOLLOWS:

1. INIT "\$TMP7" TO START WITH A COUNT OF 1
2. CLEAR THE DH11 AND FILL SILO WITH 64. WORDS
3. READ THE NO. OF WORDS SPECIFIED BY COUNT
4. CHECK THAT FILL LEVEL=64. MINUS COUNT - REPORT ERRORS
5. INCREMENT "\$TMP7" AND REPEAT 2 THRU 4 UNTIL ALL
COUNTS TESTED.

ERRORS:

1. ERROR 6 IS CALLED TO REPORT SILO FILL LEVEL ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG: (REFER TO TEST 35)

KEY LOGIC: (REFER TO TEST 35)

%
MOV R1,R2 ;SET UP REGADR
ADD #SSR,R2
MOV #1,\$TMP7 ;START WITH COUNT = 1
1\$: MOV #BIT11,(R1) ;CLR THE DH11
MOV #100,R5 ;TEST COUNT SHOULD BE 64(10) MINUS
SUB \$TMP7,R5 ;THE NO. OF CHARS READ
MOV #100,R3 ;COUNTER USED TO FILL SILO
2\$: MOV #1000,R4 ;INIT TIMER
BIS #BIT15,(R2) ;SET SILO MAINT. BIT
3\$: DEC R4 ;STALL TO ALLOW SILO TO LOAD
BNE 3\$
BIC #BIT15,(R2) ;CLEAR THE SILO MAINT BIT
DEC R3 ;COUNT ONE CHAR LOADED
BNE 2\$;BR UNTIL ALL LOADED
MOV \$TMP7,R3 ;INIT COUNTER FOR READING SILO
4\$: MOV #1000,R4 ;INIT TIMER
TST NRC(R1) ;READ THE SILO
5\$: DEC R4 ;GIVE IT TIME TO SETTLE
BNE 5\$
DEC R3 ;COUNT ONE READ
BNE 4\$;BR UNTIL WE'VE READ TEST COUNT

```
5688
5689 011414 011203          MOV      (R2),R3          ;GET THE WAS DATA
5690 011416 042703 140377  BIC      #140377,R3      ;CLR JUNK BITS
5691 011422 010504          MOV      R5,R4          ;SET UP S/B DATA
5692 011424 000304          SWAB     R4
5693 011426 020304          CMP      R3,R4          ;DID IT DOWN COUNT OK ??
5694 011430 001406          BEQ      6$            ;BR IF YES
5695
5696 011432 004737 024412    JSR      PC,SUER2        ;GO SET UP ERROR INFO
5697 011436 012737 011324 001110  MOV      #1$,$LPERR     ;SET UP ERROR LOOP RETURN
5698 011444 104006          ERROR    6             ;SILO STATUS REG. DOWN-COUNTED INCORRECTLY
5699
5700 011446 005237 001220    6$:    INC      $TMP7         ;UPDATE COUNT
5701 011452 022737 000101 001220    CMP      #101,$TMP7     ;TESTED ALL COUNTS ??
5702 011460 001321          BNE      1$            ;BR IF NOT
5703
5704
```


5705
5706
5707
5708 011462 000004
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754 011464 012737 011504 001110
5755 011472 010102
5756 011474 062702 000016
5757 011500 005037 001220
5758 011504 012711 004000
5759 011510 013705 001220
5760 011514 010512

;*TEST 37 TEST SILO ALARM LEVEL FOR COUNTS 0,1,2,4,8,16, AND 32

TST37: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE SILO ALARM LEVEL WORKS PROPERLY FOR INTEGRAL POWER OF 2 COUNTS (0, 1, 2, 4, 8, 16, AND 32). THE TEST SEQUENCE IS AS FOLLOWS:

1. INIT "\$TMP7" TO START WITH ALARM LEVEL OF 000
2. CLEAR THE DH11 AND LOAD THEN SILO WITH THAT NO. OF WORDS THAT IS ONE GREATER THAN THE ALARM LEVEL.
3. VERIFY THAT "DATA READY" DOES NOT SET UNTIL THE FILL LEVEL EXCEEDS THE ALARM LEVEL.
4. REPORT ERRORS IF:
 - A. "READY" SETS TOO SOON
 - B. "READY" SETS TOO LATE
5. SHIFT "\$TMP7" LEFT TO GENERATE NEXT POWER OF 2 LEVEL
6. REPEAT 2 THRU 5 UNTIL ALL 7 TEST LEVELS CHECKED

NOTE: FOR (A) ABOVE IF "READY" SETS JUST ONE WORD TOO SOON, IT IS ALLOWED, BUT ANYTHING GREATER RESULTS IN AN ERROR MESSAGE.

ERRORS:

1. ERROR 6 IS CALLED TO REPORT BOTH TYPES OF ERRORS OUTLINED IN 4(A,B) ABOVE

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. ERRORS IN THIS TEST ONLY INDICATE BAD COMPARATOR CHIP (E23 OR E19) ON THE M7279 - SH2

KEY LOGIC:

M7279 SH2 E19 - PIN 5 (COMPARATOR)
ALSO SAME LOGIC AS TEST 35

```

%
MOV #15,$LPERR ;SET UP THE ERROR LOOP RETURN
MOV R1,R2 ;SET UP REGADR
ADD #SSR,R2
CLR $TMP7 ;START WITH LEVEL 00
1$: MOV #BIT11,(R1) ;CLEAR THE DH11
MOV $TMP7,R5 ;SAVE IT IN R5
MOV R5,(R2) ;SET ALARM LEVEL IN SSR

```

```

5761 011516 005205          INC      R5          ;LOAD ONE MORE THAN FILL LEVEL
5762 011520 052712 100000 2$:  BIS      #BIT15,(R2) ;SET SILO MAINT. TO LOAD A CHAR
5763 011524 012703 001000    MOV      #1000,R3    ;INIT STALL TIMER
5764 011530 005303          3$:  DEC      R3          ;WAIT FOR SILO TO SETTLE
5765 011532 001376          BNE     3$          ;BR TIL R3 GOES TO 000000
5766 011534 042712 100000    BIC      #BIT15,(R2) ;CLR THE SILO MAINT BIT
5767 011540 005305          DEC      R5          ;COUNT ONE LOADED
5768 011542 105711          TSTB    (R1)        ;CHAR AVAIL SET YET
5769 011544 100412          BMI     4$          ;BR IF IT IS
5770 011546 005705          TST     R5          ;SHOULD IT BE ??
5771 011550 001363          BNE     2$          ;BR IF NOT
5772
5773 011552 004737 027200    JSR     PC,SAPS     ;SAVE THE ERROR PSW
5774 011556 004737 011624    JSR     PC,5$       ;GO SET UP S/B DATA
5775 011562 004737 024416    JSR     PC,SUER2A   ;GO SET UP ERROR INFO
5776 011566 104006          ERROR   6          ;SILO ALARM LEVEL FAILED AT SELECTED COUNT
5777 011570 000426          BR      6$          ;GO CHECK NEXT COUNT
5778
5779 011572 005705          4$:  TST     R5          ;SHOULD IT HAVE BEEN SET (CHAR AVAIL)
5780 011574 001424          BEQ     6$          ;BR IF YES
5781
5782 011576 004737 027200    JSR     PC,SAPS     ;SAVE THE ERROR PSW
5783 011602 022705 000001    CMP     #1,R5      ;IS IT OFF BY ONLY ONE ??
5784 011606 001417          BEQ     6$          ;BR IF YES - WE'LL ALLOW HIM THIS
5785 011610 004737 011624    JSR     PC,5$       ;GO SET UP S/B DATA
5786 011614 004737 024416    JSR     PC,SUER2A   ;GO SET UP ERROR INFO
5787 011620 104006          ERROR   6          ;SILO ALARM LEVEL FAILED
5788 011622 0004:1          BR      6$          ;GO CHECK NEXT COUNT
5789
5790 011624 011203          5$:  MOV     (R2),R3     ;GET WAS DATA
5791 011626 013704 001220    MOV     $TMP7,R4   ;SET UP THE S/B DATA
5792 011632 005204          INC     R4
5793 011634 000304          SWAB   R4
5794 011636 105004          CLR    R4
5795 011640 153704 001220    BISB   $TMP7,R4
5796 011644 000207          RTS     PC          ;RETURN TO SET UP AND REPORT ERROR
5797
5798 011646 005737 001220    6$:  TST     $TMP7     ;COUNT AT ZERO
5799 011652 001002          BNE     7$          ;BR IF NOT
5800 011654 000261          SEC
5801 011656 000401          BR      8$          ;SET THE 'C' BIT
5802 011660 000241          7$:  CLC
5803 011662 006137 001220    8$:  ROL     $TMP7     ;GO SET UP COUNT
5804 011666 032737 000100 001220  ROL     #BIT6,$TMP7 ;CLEAR THE 'C' BIT
5805 011674 001703          BEQ     1$          ;SHIFT POWER OF TWO BIT
5806
5807
5808
5809

```

5810
5811
5812
5813 011676 000004
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865

: *TEST 40 TRANSMITTER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS
: *****
TST40: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST PERFORMS A "RELATIVE" TIMING TEST FOR ALL BAUD RATES ON ALL SELECTED LINES. IT DOES NOT MEASURE ABSOLUTE TIMES BUT SIMPLY VERIFIES THAT EACH SUCCESSIVE SPEED FROM 50 TO 9600 BAUD IS FASTER THAN THE PREVIOUS SPEED. THE TEST SEQUENCE IS AS FOLLOWS:

1. SELECT A LINE # TO TEST (AS DEFINED BY "LINSEL:")
2. INIT "STMP7" TO START WITH 50 BAUD AND A RELATIVE TIMER "TIMEC" TO -1 (177777)
3. CLEAR THE DH11 AND ACTIVATE SELECTED LINE TO TRANSMIT THREE CHARS.
4. ACTIVATE TIMER TO UPDATE "TIMEB" THE LINE SPEED TIMER
5. IF "XMIT DONE" FAILS TO SET ON TIME - REPORT ERROR AND REPEAT 3 THRU 4 UNTIL ALL SPEEDS CHECKED - THEN REPEAT 1 THRU 5 UNTIL ALL LINES CHECKED
6. IF "XMIT DONE" SETS VERIFY TIMEB LESS THAN TIMEC IF NOT REPORT ERROR - MAKE TIMEC = TIMEB AND REPEAT 3 THRU 5 UNTIL ALL SPEEDS CHECKED
7. REPEAT 1 THRU 6 FOR ALL SELECTED LINES.

ERRORS:

1. ERROR 53 IS CALLED TO REPORT XMIT TIMEOUT ERRORS
2. ERROR 17 IS CALLED TO REPORT TIMING ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF ALL LINES FAIL ON ALL SPEEDS SUSPECT THE CLOCK MODULE M4540
2. IF ALL LINES FAIL ON JUST ONE SPEED (THE SAME ONE) SUPECT EITHER THE CLOCK MODULE OR THE M7288 MODULE (TIMING SELECT MUXES)
3. IF JUST ONE LINE FAILS SUSPECT EITHER THE UART MODULE (M7280) EITHER FOR LINES <15:08> OR <07:00> OR THE M7288 MODULE

KEY LOGIC:

M4540	SH2	<9600:50> BAUD SIGNALS
M7288	SH3	BOT AND TOP BUF CLOCK SIGNALS
	SH4,6,8,OR 10	TX CLOCK NN L SIGNALS
M7280	TBMT LINE "N"	SIGNALS ON UART PIN 22
	TX CLOCK LINE "N"	SIGNALS ON UART PIN 40

%

TRANSMITTER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS

5866	011700	012737	011730	001110		MOV	#2\$,SLPERR	;SET UP ERROR LOOP RETURN
5867	011706	004737	024544		1\$:	JSR	PC,SELINE	;GO SELECT A LINE TO TEST
5868	011712	000534				BR	TST41	;BR IF TESTED ALL SELECTED LINES
5869	011714	012737	002100	001220		MOV	#2100,\$TMP7	;INIT TI START WITH LOWEST SPEED
5870	011722	012737	177777	030354		MOV	#-1,TIMEC	;INIT RELATIVE TIME CHECKER
5871	011730	012711	004000		2\$:	MOV	#BIT11,(R1)	;CLEAR THE DH11
5872	011734	153711	030322		3\$:	BISB	LINE,(R1)	;SELECT IT IN THE SCR
5873	011740	012761	177775	000010		MOV	#-3,BCR(R1)	;SET BYTE COUNT TO XFER 3 CHARS
5874	011746	005061	000006			CLR	CAR(R1)	;GET TEST DATA STARTING AT LOC. 0
5875	011752	013761	001220	000004		MOV	\$TMP7,LPR(R1)	;SELECT A XMIT SPEED
5876	011760	013761	027314	000012		MOV	LINMSK,BAR(R1)	;ACTIVATE THE TRANSMITTER
5877								
5878	011766	012737	000001	030350		MOV	#1,TIMEA	;INIT TIMER A
5879	011774	005037	030352			CLR	TIMEB	;INIT TIMER B
5880	012000	005711			4\$:	TST	(R1)	;XMITTR DONE SET YET ?
5881	012002	100437				BMI	5\$;BR IF YES
5882	012004	004737	027016			JSR	PC,TIMEIT	;CALL THE TIMER
5883	012010	000773				BR	4\$;TIMER ROUTINE WILL MOVE RETURN PC
5884								;AROUND THIS BRANCH IF TIME OUT OCCURS
5885								
5886	012012	013737	001220	001202		MOV	\$TMP7,\$TMP0	;SAVE AND SET UP THE SPEED CODE
5887	012020	000337	001202			SWAB	\$TMP0	
5888	012024	006237	001202			ASR	\$TMP0	
5889	012030	006237	001202			ASR	\$TMP0	
5890	012034	042737	177760	001202		BIC	#177760,\$TMP0	
5891	012042	011103				MOV	(R1),R3	;GET THE WAS DATA
5892	012044	042703	000200			BIC	#BIT07,R3	;CLEAR UNINTERESTING BITS
5893	012050	010102				MOV	R1,R2	;MAKE REGADR = DEVADR
5894	012052	012704	100000			MOV	#BIT15,R4	;SET UP S/B DATA
5895	012056	153704	030322			BISB	LINE,R4	
5896	012062	004737	024416			JSR	PC,SUER2A	;GO SET UP ERROR INFO
5897	012066	004537	024636			JSR	R5,SUNUM	;GO SET LINE NO. IN MSG
5898	012072	030322				LINE		
5899	012074	034254				EM50+53		
5900	012076	104053				ERROR	53	;TIMED OUT WAITING FOR XMIT DONE
5901	012100	000426				BR	8\$;GO TEST NEXT SPEED
5902								
5903	012102	013703	030352		5\$:	MOV	TIMEB,R3	;GET THE WAS COUNT
5904	012106	013704	030354			MOV	TIMEC,R4	;GET LASTR CHECK COUNT
5905	012112	020304				CMP	R3,R4	;COMPARE RELATIVE TIMES
5906	012114	103420				BLO	8\$;BR IF THIS SPEED FASTER THAN LAST
5907								;SPEED TESTED
5908								
5909	012116	004737	027200		7\$:	JSR	PC,SAPS	;SAVE THE ERROR PSW
5910	012122	013702	001220			MOV	\$TMP7,R2	;GET SPEED CODE AND RIGHT JUSTIFY
5911	012126	000302				SWAB	R2	
5912	012130	006202				ASR	R2	
5913	012132	006202				ASR	R2	
5914	012134	042702	177760			BIC	#177760,R2	;STRIP AWAY ALL JUNK
5915	012140	004737	024416			JSR	PC,SUER2A	;GO SET UP ERROR INFO
5916	012144	004537	024636			JSR	R5,SUNUM	;GO PUT LINE NO. IN MSG
5917	012150	030322				LINE		
5918	012152	031617				EM17+41		
5919	012154	104017				ERROR	17	;TRANSMITTER SPEED INCORRECT
5920								
5921	012156	013737	030352	030354	8\$:	MOV	TIMEB,TIMEC	;SET UP NEW CHECK TIMER COUNT

CZDMM-D-0
CZDMM-D.P11

MACY11 30A(1052) 10-MAR-78
09-MAR-78 15:32 T40

08:05 PAGE 142

J 11

TRANSMITTER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS

SEQ 0140
SEQ 0139

5922	012164	062737	002100	001220	ADD	#2100,\$TMP7	:GENERATE NEXT SPEED
5923	012172	022737	035600	001220	OMP	#35600,\$TMP7	:DONE ALL SPEEDS ?
5924	012200	001253			BNE	2\$:BR IF NOT
5925	012202	000641			BR	1\$:GO TEST NEXT LINE
5926							

5927
5928
5929
5930 012204 000004

*TEST 41 RECEIVER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS

TST41: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST IS IDENTICAL TO TEST 40 EXCEPT IT WAITS FOR "DATA READY"
TO CHECK RECEIVER TIMING. THE SEQUENCE IS SIMILAR AND THE SAME TIMERS
ARE USED FOR ERROR CHECKING.

ERRORS:

- 1. ERROR 54 IS CALLED TO REPORT RCVR TIMEOUT ERRORS
- 2. ERROR 20 IS CALLED TO REPORT RCVR TIMING ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG: (SAME AS TEST 42)

KEY LOGIC: (SAME AS TEST 42 PLUS)

M7288 SH5,7,9,11 RX CLOCK NM L SIGNALS
M7280 BUF DA LINE 'N' UART PIN 19
RX CLOCK LINE 'N' UART PIN 17

5958 %
5959 012206 012737 012236 001110 MOV #2\$,SLPERR ;SET UP ERROR LOOP RETURN
5960 012214 004737 024544 1\$: JSR PC,SELINE ;GO SELECT A LINE TO TEST
5961 012220 000534 BR TST42 ;:BR IF TESTED ALL SELECTED LINES
5962 012222 012737 002100 001220 MOV #2100,\$TMP7 ;INIT TO START WITH LOWEST SPEED
5963 012230 012737 177777 030354 MOV #-1,TIMEC ;INIT RELATIVE TIME CHECKER
5964 012236 012711 004000 2\$: MOV #BIT11,(R1) ;CLEAR THE DH11
5965 012242 012711 001000 MOV #BIT9,(R1) ;SET MAINTENANCE MODE ENABLE
5966 012246 153711 030322 3\$: BISB LINE,(R1) ;SELECT IT IN THE SCR
5967 012252 012761 177777 000010 MOV #-1,BCR(R1) ;SET BYTE COUNT TO XFER 1 CHAR
5968 012260 005061 000006 CLR CAR(R1) ;GET TEST DATA STARTING AT LOC. 0
5969 012264 013761 001220 000004 MOV \$TMP7,LPR(R1) ;SELECT A XMIT SPEED
5970 012272 013761 027314 000012 MOV LINMSK,BAR(R1) ;ACTIVATE THE TRANSMITTER
5971
5972 012300 012737 000001 030350 MOV #1,TIMEA ;INIT TIMER A
5973 012306 005037 030352 CLR TIMEB ;INIT TIMER B
5974 012312 105711 4\$: TSTB (R1) ;RCVR DONE YET ??
5975 012314 100435 BMI 5\$;BR IF YES
5976 012316 004737 027016 JSR PC,TIMEIT ;CALL THE TIMER
5977 012322 000773 BR 4\$;TIMER ROUTINE WILL MOVE RETURN PC
5978 ;AROUND THIS BRANCH IF TIME OUT OCCURS
5979
5980 012324 013737 001220 001202 MOV \$TMP7,\$TMP0 ;SAVE AND SET UP THE SPEED CODE
5981 012332 006337 001202 ASL \$TMP0
5982 012336 006337 001202 ASL \$TMP0

5983	012342	000337	001202		SWAB	\$TMP0		
5984	012346	042737	177760	001202	BIC	#177760,\$TMP0		
5985	012354	011103			MOV	(R1),R3	;GET THE WAS DATA	
5986	012356	010102			MOV	R1,R2	;MAKE REGADR = DEVADR	
5987	012360	012704	100200		MOV	#BIT15+BIT07,R4	;SET UP S/B DATA	
5988	012364	153704	030322		BISB	LINE,R4		
5989	012370	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO	
5990	012374	004537	024636		JSR	R5,SUNUM	;GO SET LINE NO. IN MSG	
5991	012400	030322			LINE			
5992	012402	032076			EM22,51			
5993	012404	104054			ERRR	54	;TIMED OUT WAITING FOR CHAR AVAIL	
5994	012406	000426			BR	8\$;GO TEST NEXT SPEED	
5995								
5996	012410	013703	030352	5\$:	MOV	TIMEB,R3	;GET THE WAS COUNT	
5997	012414	013704	030354		MOV	TIMEC,R4	;GET THE CHECK COUNT	
5998	012420	020304			CMP	R3,R4	;COMPARE RELATIVE TIMES	
5999	012422	103420			BLO	8\$;BR IF TIME INDICATES THIS SPEED FASTER ;THAN LAST SPEED	
6000								
6001								
6002	012424	004737	027200	7\$:	JSR	PC,SAPS	;SAVE THE ERROR PSW	
6003	012430	013702	001220		MOV	\$TMP7,R2	;GET SPEED CODE AND RIGHT JUSTIFY	
6004	012434	006302			ASL	R2		
6005	012436	006302			ASL	R2		
6006	012440	000302			SWAB	R2		
6007	012442	042702	177760		BIC	#177760,R2	;STRIP AWAY ALL JUNK	
6008	012446	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO	
6009	012452	004537	024636		JSR	R5,SUNUM	;GO PUT LINE NO. IN MSG	
6010	012456	030322			LINE			
6011	012460	031756			EM20+36			
6012	012462	104020			ERROR	20	;RECEIVER SPEED INCORRECT	
6013								
6014	012464	013737	030352	030354	8\$:	MOV	TIMEB,TIMEC	;SET UP NEW CHECK TIMER COUNT
6015	012472	062737	002100	001220	ADD	#2100,\$TMP7	;GENERATE NEXT SPEED	
6016	012500	022737	035600	001220	CMP	#35600,\$TMP7	;DONE ALL SPEEDS ?	
6017	012506	001253			BNE	2\$;BR IF NOT	
6018	012510	000641			BR	1\$;GO TEST NEXT LINE	
6019								

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
T41

08:05 PAGE 145

M 11

RECEIVER TIMING TEST - ALL SELECTED LINES - ALL SPEEDS

SEQ 0143
SEQ 0142

6020

6021
6022
6023
6024 012512 000004
6025
6026
6027
6028
6029
6030
6031
6032
6033
6034
6035
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076

*TEST 42 VERIFY STORAGE OVERFLOW - NON MAINT. MODE - ALL SFLECTED LINES

TST42: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE STORAGE OVERFLOW BIT (SCR14) SETS
AND CLEARS PROPERLY FOR ALL SELECTED LINES. THE TEST SEQUENCE IS AS
FOLLOWS:

1. SET UP THE ERROR RETURN
2. SELECT A LINE NUMBER TO TEST - GO TO NEXT TEST IF ALL
SELECTED LINES HAVE BEEN TESTED.
3. PRIME THE SELECTED LINE TO XMIT 65(10) CHARS.
4. ACTIVATE THE SELECTED LINE AND WAIT FOR STORAGE OVERFLOW
TO SET (SCR14=1)
5. IF SCR14 FAILS TO SET ON TIME - REPORT ERROR AND THEN
CONTINUE WITH THE NEXT LINE (STEP 2).
6. IF IT SETS OK - READ THE "NRC" REG TWICE TO EMPTY TWO
WORDS FROM THE SILO.
7. AFTER A BRIEF STALL, VERIFY THAT SCR14 HAS CLEARED -
IF NOT REPORT ERROR AND CONTINUE WITH NEXT LINE (STEP2)
8. IF IT CLEARS OK, VERIFY THAT THE FILL COUNT (SSR<15:08>)
CONTAINS A 77(8) - IF NOT REPORT ERROR AND CONTINUE
WITH NEXT LINE (STEP 2).
9. REPEAT STEPS 2 THRU 8 UNTIL ALL SELECTED LINES HAVE BEEN
TESTED.

ERRORS:

1. ERROR 57 IS CALLED TO REPORT ALL ERRORS DETECTED.

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. PROBLEM CAN VERY LIKELY BE THAT NO TEST CONNECTOR IS PRESENT.
BE SURE YOU HAVE ONE ON FOR THIS TEST.
2. PROBLEM IS POSSIBLY ON THE M7289 MODULE (SH4) OR SOME
SIGNAL FEEDING THIS LOGIC.

KEY LOGIC:

M7289	SH4	STORAGE OVERFLOW L	E43-12
		READY IN PULSE H	E40-11
		UC1 MASTER DA H	BH2
		UC2 MASTER DA H	BD2

6077					Z						
6078	012514	012737	012530	001110		MOV	#28,SLPERR			;SET UP ERROR RETURN	
6079	012522	004737	024544		1\$:	JSR	PC,SELINE			;GO REFLECT A LINE NO. TO TEST	
6080	012526	000535				BR	TST43			;BR IF DONE ALL SELECTED LINES	
6081	012530	012711	004000		2\$:	MOV	#BIT11,(R1)			;CLEAR THE DH11	
6082	012534	010102				MOV	R1,R2			;MAKE REGADR = DEVADR	
6083	012536	113711	030322			MOVB	LINE,(R1)			;SELECT A LINE	
6084	012542	005061	000006			CLR	CAR(R1)			;SET UP CURRENT ADDRESS	
6085	012546	012761	177677	000010		MOV	#-65,BCR(R1)			;SET UP BYTE COUNT	
6086	012554	012761	033500	000004		MOV	#33500,LPR(R1)			;SET UP LPR: 9600 BAUD,5 BIT CHARS	
6087	012562	013761	027314	000012		MOV	LINMSK,BAR(R1)			;ACTIVATE THE SELECTED LINE	
6088											
6089	012570	012737	000001	030350		MOV	#1,TIMEA			;INIT TIMERS	
6090	012576	005037	030352			CLR	TIMEB				
6091	012602	032711	040000		3\$:	BIT	#BIT14,(R1)			;STORAGE OVERFLOW YET ??	
6092	012606	001024				BNE	4\$;BR IF YES YOU SHOULD GET IT	
6093	012610	004737	027016			JSR	PC,TIMEIT			;CALL TIMER	
6094	012614	000772				BR	3\$;BR IF NO TIME OUT	
6095											
6096	012616	004737	027200			JSR	PC,SAPS			;GO SAVE PSW	
6097	012622	012704	040000			MOV	#BIT14,R4			;SET UP S/B DATA	
6098	012626	153704	030322			BISB	LINE,R4				
6099	012632	011103				MOV	(R1),R3			;SET UP WAS DATA	
6100	012634	042703	137760			BIC	#137760,R3			;CLEAR UNINTERESTING BITS	
6101	012640	004737	024416			JSR	PC,SUER2A			;GO SET UP ERROR INFO	
6102	012644	004537	024636			JSR	R5,SUNUM			;PUT LINE NO. IN MESSAGE	
6103	012650	030322				LINE					
6104	012652	034741				EM57+44					
6105	012654	104057				ERROR	57			;STORAGE OVERFLOW FAILED TO SET	
6106	012656	000721				BR	1\$;GO TRY NEXT LINE	
6107											
6108	012660	016137	000002	001220	4\$:	MOV	NRC(R1),STMP7			;READ THE SILO	
6109	012666	016137	000002	001220		MOV	NRC(R1),STMP7			;READ IT AGAIN	
6110	012674	012705	001000			MOV	#1000,R5			;INIT STALL COUNTER	
6111	012700	005305			41\$:	DEC	R5			;COUNT TIMER	
6112	012702	001376				BNE	41\$;BR IF NO TIMEOUT	
6113	012704	032711	040000			BIT	#BIT14,(R1)			;DID OVERFLOW GO AWAY ?	
6114	012710	001420				BEQ	5\$;BR IF YES	
6115											
6116	012712	004737	027200			JSR	PC,SAPS			;GO SAVE THE PSW	
6117	012716	005004				CLR	R4			;SET UP S/B DATA	
6118	012720	153704	030322			BISB	LINE,R4				
6119	012724	011103				MOV	(R1),R3			;SET UP WAS DATA	
6120	012726	042703	137760			BIC	#137760,R3			;CLEAR UNINTERESTING BITS	
6121	012732	004737	024416			JSR	PC,SUER2A			;GO SET UP ERROR INFO	
6122	012736	004537	024636			JSR	R5,SUNUM			;PUT LINE NO. IN MSG	
6123	012742	030322				LINE					
6124	012744	034741				EM57+44					
6125	012746	104057				ERROR	57			;STORAGE BIT FAILED TO CLEAR	
6126	012750	000664				BR	1\$;GO TRY NEXT LINE	
6127											
6128	012752	122761	000077	000017	5\$:	CMPB	#77,SSR+1(R1)			;WAS IT REALLY 65. ??	
6129	012760	001660				BEQ	1\$				
6130											
6131	012762	004737	027200			JSR	PC,SAPS			;GO SAVE PSW	
6132	012766	012704	037400			MOV	#37400,R4			;SET UP S/B DATA	

CZDHM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
T42

08:05 PAGE 148

C 12

VERIFY STORAGE OVERFLOW - NON MAINT. MODE - ALL SELECTED LINES

SEG 0146
SEQ 0145

6133 012772 062702 000016
6134 012776 016103 000016
6135 013002 004737 024416
6136 013006 004537 024636
6137 013012 030322
6138 013014 034741
6139 013016 104057
6140
6141 013020 000640

ADD #SSR,R2
MOV SSR(R1),R3
JSR PC,SUER2A
JSR R5,SUNUM
LINE
EM57+44
ERROR 57
BR 18

;SET UP RFGADR
;SAVE WAS DATA
;GO SET UP ERROR INFO
;PUT LINE NO. IN MSG
;READING SILO FAILED TO DEC SSR OR
;STORAGE OVFL SET AT WRONG COUNT
;GO TRY NEXT LINE

6142
6143
6144
6145 013022 000004
6146
6147
6148
6149
6150
6151
6152
6153
6154
6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
6188
6189
6190
6191
6192
6193
6194
6195
6196
6197 013024 012737 013052 001110

: *TEST 43 BASIC DATA TEST - ALL SELECTED LINES/ALL CHAR LENGTHS
: *****

TST43: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT A SINGLE ALL ONES CHAR. CAN BE TRANS-
MITTED AND RECEIVED ON ALL SELECTED LINES AT ALL FOUR CHAR LENGTHS
(5, 6, 7, AND 8 BITS). THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. SELECT A LINE NO. TO TEST - GO TO THE NEXT TEST IF DONE ALL SELECTED LINES.
3. GET A TEST CHARACTER FROM THE DATA TABLE AND UPDATE THE TABLE POINTER.
4. CLEAR THE DH11
5. PRIME THE SELECTED LINE TO XMIT ONE CHAR AT 9600 BAUD
6. WAIT FOR "CHAR AVAIL" TO SET - IF TIMEOUT REPORT ERROR AND RESTART AT STEP 8.
7. IF NO TIMEOUT - CHECK DATA AND REPORT ANY ERRORS
8. INCREMENT "SCR" REG TO CHANGE CHAR LENGTH - IF DONE ALL FOUR GO TO STEP 2 - IF NOT THEN STEP 4.

ERRORS:

1. ERROR 55 IS CALLED TO REPORT RCVR TIMEOUT.
2. ERROR 23 IS CALLED TO REPORT DATA COMPARE ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF FAULT AFFECTS ONLY ONE LINE AT ALL CHAR LENGTHS, SUSPECT A BAD UART MODULE M7280.
2. IF FAULT AFFECTS ONLY ONE BIT ON ALL LINES, SUSPECT THE THE M7279 MODULE.
3. IF FAULT AFFECTS ONLY CERTAIN CHAR LENGTHS, SUSPECT EITHER THE M7278 OR THE UART MODULE M7280.

KEY LOGIC:

M7280'S UART CHIPS PINS <12:05>

M7279 SH1 E1,E2,E6, OR E7

M7278 SH8 NB2 LPR 01 H FH1
NB1 LPR 00 H FH2

MOV #3\$, \$LPERR ;SET UP ERROR LOOP RETURN

BASIC DATA TEST - ALL SELECTED LINES/ALL CHAR LENGTHS

6198	013032	004737	024544	1\$:	JSR	PC,SELINE	:GO SELECT A LINE TO TEST
6199	013036	000511			BR	TST44	:BR IF DONE ALL SELECTED LINES
6200	013040	012705	030336		MOV	#TDATA2,R5	:GET POINTER TO DATA TABLE
6201	013044	005002			CLR	R2	:INIT R2 TO START AT CHAR LENGTH OF 5 BITS
6202	013046	012537	001220	2\$:	MOV	(R5)+,\$TMP7	:PUT TEST CHAR IN XMIT BUFFER
6203	013052	012711	004000	3\$:	MOV	#BIT11,(R1)	:CLEAR THE DH11
6204	013056	153711	030322		BISB	LINE,(R1)	:SELECT THE LINE
6205	013062	012761	177777	000010	MOV	#-1,BCR(R1)	:SET BYTE COUNT TO -1
6206	013070	012761	001220	000006	MOV	#\$TMP7,CAR(R1)	:SET CURRENT ADDRESS REG
6207	013076	012761	033500	000004	MOV	#33500,LPR(R1)	:SET BAUD RATE TO 9600
6208	013104	050261	000004		BIS	R2,LPR(R1)	:SELECT CHAR LENGTH
6209	03110	053761	027314	000012	BIS	LINMSK,BAR(R1)	:ACTIVATE THE SELECTED LINE
6210							
6211	013116	012757	000001	030350	MOV	#1,TIMEA	:INIT TIMER A
6212	013124	005037	030352		CLR	TIMEB	:INIT TIMER B
6213	013130	105711		4\$:	TSTB	(R1)	:RCVR DONE YET ??
6214	013132	100424			BMI	5\$:BR IF YES
6215	013134	004737	027016		JSR	PC,TIMEIT	:CALL THE TIMER
6216	013140	000773			BR	4\$:TIMER ROUTINE WILL MOVE RETURN PC
6217							:AROUND THIS BRANCH IF TIME OUT OCCURS
6218							
6219	013142	004737	027200		JSR	PC,SAPS	:SAVE THE ERROR PSW
6220	013146	011103			MOV	(R1),R3	:GET THE SCR
6221	013150	042703	177560		BIC	#177560,R3	:CLEAR UNINTERESTING BITS
6222	013154	012704	000200		MOV	#200,R4	:SET UP S/B DATA
6223	013160	153704	030322		BISB	LINE,R4	
6224	013164	004737	024416		JSR	PC,SUER2A	:GO SET UP ERROR INFO
6225	013170	004537	024636		JSR	R5,SUNUM	:GO SET LINE NO. IN MSG
6226	013174	030322			LINE		
6227	013176	032076			EM22+51		
6228	013200	104055			ERROR	5\$:CHAR AVAIL FAILED TO SET ON TIME
6229	013202	000422			BR	6\$:GO TEST NEXT CHAR LENGTH
6230							
6231	013204	016103	000002	5\$:	MOV	NRC(R1),R3	:GET THE WAS DATA
6232	013210	012704	000200		MOV	#200,R4	:SET UP THE S/B DATA IN R4
6233	013214	153704	030322		BISB	LINE,R4	
6234	013220	000304			SWAB	R4	
6235	013222	153704	001220		BISB	\$TMP7,R4	
6236	013226	020304			CMP	R3,R4	:WAS THE RCVD DATA CORRECT ??
6237	013230	001407			BEQ	6\$:BR IF YES
6238							
6239	013232	004737	024412		JSR	PC,SUER2	:GO SET UP THE ERROR INFO
6240	013236	004537	024636		JSR	R5,SUNUM	:GO PUT LINE NO. IN MSG
6241	013242	030322			LINE		
6242	013244	032137			EM23+36		
6243	013246	104023			ERROR	23	:DATA COMPARE ERROR
6244							
6245	013250	005202		6\$:	INC	R2	:DO NEXT CHAR LENGTH ON SELECTED LINE
6246	013252	022702	000004		CMP	#4,R2	:HAVE WE DONE ALL FOUR CHAR LENGTHS ??
6247	013256	001273			BNE	2\$:BR IF NOT
6248	013260	000664			BR	1\$:GO DO NEXT LINE

6249
6250
6251
6252 013262 000004
6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304

```
*****
:TEST 44      SINGLE LINE DATA TEST - ALL SELECTED LINES
:*****
TST44: SCOPE
.REM      %
TEST ABSTRACT:
*****
```

THIS TEST TRANSMITS AND RECEIVES A BINARY COUNT PATTERN
(000 - 377) ON ALL SELECTED LINES. THE TEST SEQUENCE IS AS
FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. GO SELECT A LINE NO. TO TEST - IF DONE ALL SELECTED LINES THEN GO TO THE NEXT TEST .
3. CLEAR THE DH11 AND PRIME THE SELECTED LINE TO XMIT TO XMIT 256. CHARS AT 9600. BAUD - 8 BIT CHARS.
4. SET UP R5 TO POINT TO RCVR CORE BUFFER.
5. ACTIVATE THE SELECTED XMITTER.
6. WAIT FOR "CHAR AVAIL" TO SET BEFORE READING THE SILO. IF RCVR TIMEOUT REPORT ERROR AND RESTART AT STEP 2.
7. IF NO TIMEOUT READ THE SILO AND STORE THE WORD IN THE RCVR CORE BUFFER - WHEN THE BUFFER IS FULL GO TO STEP8 IF NOT THEN GO TO STEP 6.
8. COMPARE THE XMIT AND RCVR CORE IMAGE BUFFERS AND REPORT ALL DATA COMPARE ERRORS.
9. CHECK THE "BAR", "BCR", AND "CAR" REGISTERS FOR CORRECT CONTENTS - REPORT ALL ERRORS.
10. GO TO STEP 2

ERRORS:

1. ERROR 22 IS CALLED TO REPORT "DATA AVAIL" TIMEOUT
2. ERROR 37 " " " DATA COMPARE ERRORS
3. ERROR 40 " " " "BAR" REG NOT CLEARED
4. ERROR 10 " " " "BCR" REG NOT ALL ZEROES
5. ERROR 7 " " " "CAR" REG NOT UPDATED CORRECTLY

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. IF THE FAULT AFFECTS ONE OR MORE LINES IN AN 8 LINE GROUP <15:08> OR <07:00>, SWAP THE M7280 MODULES. IF THE FAULT SHIFTS SO THAT THE ERROR INDICATES DIFFERENT LINES THE PROBLEM IS MOST LIKELY THE M7280 THE SYMPTOM SHIFTED TO.
2. IF THE FAULT GIVES DATA ERRORS BUT AFFECTS ONLY CERTAIN PATTERNS ON ONE LINE THE FAULT IS MOST LIKELY A "UART" CHIP.
3. IF THE FAULT GIVES DATA ERRORS BUT AFFECTS ONLY CERTAIN PATTERNS ON ALL LINES SUSPECT THE DATA PATHS EXTERNAL TO, THE M7280 MODULES.

6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350
6351
6352
6353
6354
6355
6356
6357
6358
6359
6360

4. IF THE FAULT CAUSES NO DATA ERRORS BUT GIVES 'BAR', 'BCR', OR 'CAR' ERRORS
SUSPECT THE M7278 OR M7277 MODULES.

KEY LOGIC (REFER TO TEST 43)

%

```

001110      MOV      #2$, $LPERR      ;SET UP ERROR LOOP RETURN
1$:         JSR      PC, SELINE      ;GO SELECT A LINE TO TEST
           BR       11$              ;BR IF ALL SELECTED LINES DONE
           BR       2$              ;GO TEST IT
11$:        JMP      8$              ;EXIT TEST
2$:         MOV      DHADR, R1       ;RESET DEVADR
           MOV      #BIT11, (R1)    ;CLEAR THE DH11
           BISB     LINE, (R1)      ;SET SELECT BITS IN SCR
           MOV      #TBUF, CAR(R1)  ;SET UP BUS ADDRESS REG
           MOV      #-400, BCR(R1)  ;SET UP BYTE COUNT
           MOV      #33503, LPR(R1) ;SET LINE PARAMETERS
           MOV      #RBUF, R5       ;SET UP POINTER TO INPUT DATA BUFFER
           BIS      #BIT09, (R1)    ;SET MAINT MODE BIT
           MOV      LINMSK, BAR(R1) ;ACTIVATE THE SELECTED LINE

030350      MOV      #2, TIMEA       ;INIT TIMER A
           CLR      TIMEB           ;INIT TIMER B
3$:         TSTB     (R1)           ;RCVR DONE YET ??
           BMI      4$              ;BR IF YES
           JSR      PC, TIMEIT      ;CALL THE TIMER
           BR       3$              ;TIMER ROUTINE WILL MOVE RETURN PC
           ;AROUND THIS BRANCH IF TIME OUT OCCURS

           JSR      PC, SAPS        ;SAVE THE ERROR PSW
           MOV      R1, R2          ;SET UP REGADR
           MOV      (R2), R3        ;GET THE WAS DATA
           BIC      #176400, R3     ;CLEAR JUNK BITS
           MOV      #1200, R4       ;SET UP S/B DATA
           BISB     LINE, R4
           JSR      PC, SUER2A      ;GO SET UP ERROR INFO
           JSR      R5, SUNUM       ;PUT LINE NO. IN MESSAGE
           LINE     EM22+51
           ERROR    22              ;CHAR AVAIL TIMEOUT
           BR       1$              ;GO TRY NEXT LINE

4$:         MOV      NRC(R1), (R5)+ ;SAVE THE RECEIVED DATA
           CMP      #RBUF+1000, R5 ;INPUT BUFFER FULL ??
           BNE     3$              ;BR IF NOT

5$:         MOV      #TBUF, R2      ;SET UP POINTER TO OUTPUT BUFFER
           MOV      #RBUF, R1      ;SET UP POINTER TO INPUT BUFFER
           MOVB    (R2), R4        ;SET UP S/B DATA IN R4
           BIC      #177400, R4
           SWAB    R4
           BISB    LINE, R4
           BISB    #200, R4
           SWAB    R4

```

6361	013516	011103		MOV	(R1),R3	;GET THE WAS DATA
6362	013520	020304		CMP	R3,R4	;DATA CORRECT ??
6363	013522	001407		BEQ	6\$;BR IF YES
6364						
6365	013524	004737	024412	JSR	PC,SUER2	;GO SET UP ERROR INFO
6366	013530	004537	024636	JSR	R5,SUNUM	;PUT LINE NO. IN MESSAGE
6367	013534	030322		LINE		
6368	013536	033354		EM37+33		
6369	013540	104037		ERROR	37	;DATA COMPARE ERROR
6370						
6371						
6372	013542	005202		6\$: INC	R2	;UPDATE DATA BUFFER POINTERS
6373	013544	062701	000002	ADD	#2,R1	
6374	013550	022701	037312	CMP	#RBUF+1000,R1	;COMPARED ALL 256. CHARS ??
6375	013554	001347		BNE	5\$;BR IF NOT
6376						
6377	013556	013701	027302	MOV	DHADR,R1	;RESET DEVADR
6378	013562	010102		MOV	R1,R2	;SET UP REGADR
6379	013564	062702	000012	ADD	#BAR,R2	
6380	013570	005712		TST	(R2)	;WAS THE 'BAR' ALL ZEROES ??
6381	013572	001413		BEQ	7\$;BR IF YES
6382						
6383	013574	004737	027200	JSR	PC,SAPS	;SAVE THE ERROR PSW
6384	013600	011203		MOV	(R2),R3	;GET THE WAS DATA
6385	013602	005004		CLR	R4	;SET UP S/B DATA
6386	013604	004737	024416	JSR	PC,SUER2A	;GO SET UP ERROR INFO
6387	013610	004537	024636	JSR	R5,SUNUM	;PUT LINE NO. IN MESSAGE
6388	013614	030322		LINE		
6389	013616	033417		EM40+40		
6390	013620	104040		ERROR	40	;'BAR' REG NOT ALL ZEROES
6391						
6392	013622	010102		7\$: MOV	R1,R2	;SET UP REGADR
6393	013624	062702	000010	ADD	#BCR,R2	
6394	013630	005712		TST	(R2)	;BYTE COUNT REG ALL ZEROES ?
6395	013632	001413		BEQ	71\$;BR IF BYTE COUNT ZERO
6396						
6397	013634	004737	027200	JSR	PC,SAPS	;SAVE THE ERROR PSW
6398	013640	011203		MOV	(R2),R3	;GET THE WAS DATA
6399	013642	005004		CLR	R4	;SET UP THE S/B DATA
6400	013644	004737	024416	JSR	PC,SUER2A	;GO SET UP ERROR INFO
6401	013650	004537	024636	JSR	R5,SUNUM	;PUT LINE NO. IN MESSAGE
6402	013654	030322		LINE		
6403	013656	031224		EM10+44		
6404	013660	104010		ERROR	10	;BYTE COUNT NOT ALL ZEROES
6405						
6406	013662	010102		71\$: MOV	R1,R2	;SET UP REGADR
6407	013664	062702	000006	ADD	#CAR,R2	
6408	013670	022712	037712	CMP	#TBUF+400,(R2)	;DID 'CAR' INCREMENT PROPERLY ?
6409	013674	001414		BEQ	72\$;BR IF YES
6410						
6411	013676	004737	027200	JSR	PC,SAPS	;SAVE THE ERROR PSW
6412	013702	011203		MOV	(R2),R3	;GET THE WAS DATA
6413	013704	012704	037712	MOV	#TBUF+400,R4	;SET UP S/B DATA
6414	013710	004737	024416	JSR	PC,SUER2A	;GO SET UP ERROR INFO
6415	013714	004537	024636	JSR	R5,SUNUM	;GO PUT LINE NO IN MESSAGE
6416	013720	030322		LINE		

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
T44

08:05 PAGE 154

I 12

SINGLE LINE DATA TEST - ALL SELECTED LINES

SEQ 0152
SEQ 0151

6417	013722	031155			EM7+47	
6418	013724	104007			ERROR 7	;'CAR' NOT UPDATED CORRECTLY
6419						
6420	013726	000137	013272	72\$:	JMP 1\$:GO DO NEXT LINE
6421						
6422	013732	000240		8\$:	NOP	:EXIT POINT

6423
6424
6425
6426 013734 000004
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458 013736 012737 013752 001110
6459 013744 004737 024544
6460 013750 000506
6461 013752 012711 004000
6462 013756 012704 000260
6463 013762 153704 030322
6464 013766 000304
6465 013770 153711 030322
6466 013774 012737 000377 030334
6467 014002 012761 073563 000004
6468 014010 012761 177777 000010
6469 014016 012761 030334 000006
6470 014024 013761 027314 000014
6471 014032 013761 027314 000012
6472
6473 014040 012737 000001 030350
6474 014046 005037 030352
6475 014052 105711
6476 014054 100423
6477 014056 004737 027016
6478 014062 000773

*TEST 45 BASIC PARITY LOGIC TEST - ALL SELECTED LINES - ODD PARITY

TST45: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THE ODD PARITY FUNCTION FOR ALL SELECTED LINES USING THE "BREAK" FUNCTION TO FORCE PARITY ERRORS. REFER TO THE FLOW CHARTS IN THE PROGRAM DOCUMENTATION FOR TEST SEQUENCES.

ERRORS:

- 1. ERROR 22 IS CALLED TO REPORT RCVR TIMEOUT
- 2. ERROR 33 IS CALLED TO REPORT DATA/PARITY ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

- 1. IF FAULT AFFECTS ALL LINES SUSPECT THE M7278 MODULE.
- 2. IF IT AFFECTS ONLY ONE LINE SUSPECT THE "UART" MODULE FOR THAT LINE.

KEY LOGIC:

```

M7278 SH7 PEN LPR04 L FF2
PEV LPR05 L FN1
%
MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN
1$: JSR PC, SELINE ;GO SELECT A LINE NO.
BR TST46 ;:BR IF ALL SELECTED LINES DONE
2$: MOV #BIT11, (R1) ;CLEAR OUT THE DH11
MOV #260, R4 ;SET UP THE S/B DATA IN R4
BISB LINE, R4
SWAB R4
BISB LINE, (R1) ;SET LINE NO. IN SCR
MOV #377, TDATA1 ;LOAD XMIT BUFFER WITH TEST CHARACTER
MOV #73563, LPR(R1) ;SET UP THE LINE PARAMETERS
MOV #-1, BCR(R1) ;LOAD THE BYTE COUNT REG
MOV #TDATA1, CAR(R1) ;LOAD THE BUS ADDR REG
MOV LINMSK, BKR(R1) ;SET BREAK BIT FOR SELECTED LINE
MOV LINMSK, BAR(R1) ;ACTIVATE THE XMITTR
MOV #1, TIMEA ;INIT TIMER A
CLR TIMEB ;INIT TIMER B
3$: TSTB (R1) ;RCVR DONE YET ??
BMI 4$ ;BR IF YES
JSR PC, TIMEIT ;CALL THE TIMER
BR 3$ ;TIMER ROUTINE WILL MOVE RETURN PC
```

```

6479                                     ;AROUND THIS BRANCH IF TIME OUT OCCURS
6480
6481 014064 004737 027200             JSR   PC,SAPS           ;SAVE THE ERROR PSW
6482 014070 011103                   MOV   (R1),R3          ;GET THE WAS DATA
6483 014072 012704 100200             MOV   #100200,R4      ;SET UP THE S/B DATA
6484 014076 153704 030322             BISB  LINE,R4
6485 014102 010102                   MOV   R1,R2           ;SET UP REGADR
6486 014104 004737 024416             JSR   PC,SUER2A       ;GO SET UP ERROR INFO
6487 014110 004537 024636             JSR   R5,SUNUM        ;PUT LINE NO. IN MESSAGE
6488 014114 030322                   LINE
6489 014116 032076                   EM22+51
6490 014120 104022                   ERROR 22              ;TIMED OUT WAITING FOR DATA AVAIL
6491 014122 000710                   BR    1$              ;GO TEST NEXT LINE
6492
6493 014124 016103 000002             4$: MOV   NRC(R1),R3   ;GET THE WAS DATA
6494 014130 020304                   CMP   R3,R4           ;CORRECT DATA RECEIVED ??
6495 014132 001704                   BEQ   1$              ;BR IF YES
6496
6497 014134 004737 027200             JSR   PC,SAPS           ;SAVE THE ERROR PSW
6498 014140 010102                   MOV   R1,R2           ;SET UP THE REGADR
6499 014142 062702 000002             ADD   #NRC,R2
6500 014146 004737 024416             JSR   PC,SUER2A       ;GO SET UP ERROR INFO
6501 014152 004537 024636             JSR   R5,SUNUM        ;PUT LINE NO. IN MESSAGE
6502 014156 030322                   LINE
6503 014160 033070                   EM33+40
6504 014162 104033                   ERROR 33              ;INCORRECT DATA OR PARITY ERROR
6505 014164 000667                   BR    1$              ;GO TEST NEXT LINE

```

6506
6507
6508
6509 014166 000004

```
*****  
: *TEST 46 MULTI-LINE PARITY DATA TEST - ALL SELECTED LINES  
: *****  
TST46: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

6510
6511
6512
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526
6527
6528
6529
6530
6531
6532
6533
6534
6535
6536
6537
6538
6539

THIS TEST VERIFIES ALL SELECTED LINES CAN TRANSMIT AND RECEIVE A BINARY COUNT PATTERN WHEN RUN CONCURRENTLY. ALL CHAR LENGTHS (5, 6, 7, AND 8 BITS) ARE TESTED WITH BOTH EVEN AND ODD PARITY CHECKING SPECIFIED. THE TEST ACTUALLY INCLUDES EIGHT SUB-TESTS - THE PARAMETERS FOR EACH SUB-TEST RETRIEVED FROM A TABLE TAGGED "PRTYTB:". REFER TO THIS TABLE TO DETERMINE THE TEST SEQUENCE.

ERRORS:

- 1. ERROR 41 IS CALLED TO REPORT FALSE RECEIVER INTRs.
- 2. ERROR 42 IS CALLED TO REPORT SILO OVERFLOW ERRORS
- 3. ERROR 34 IS CALLED TO REPORT PARITY/DATA ERRORS
- 4. ERROR 35 IS CALLED TO REPORT TEST TIMEOUTS

SYNC: (NONE)

DEBUG: (REFER TO TEST 45)

KEY LOGIC: (REFER TO TEST 45)

014170 012737 014206 001110
014176 012705 027364
014202 005037 001220
014206 162705 000004
014212 005337 001220
014216 022705 027420
014222 001456
014224 012706 001100
014230 013701 027302
014234 012537 001216
014240 012537 001214
014244 005237 001220
014250 012711 004000
014254 004737 025020
014260 016137 000004 001202
014266 013737 027312 001210
014274 004737 027164
014300 013702 027304
014304 012722 014376
014310 113712 030316
014314 012711 000100
014320 013737 027312 027560

```
%  
MOV #1$, $LPERR ;SET UP THE ERROR LOOP RETURN  
MOV #PRTYTB+4, R5 ;SET UP POINTER TO TEST PARAMETERS  
CLR $TMP7 ;START WITH SUB TEST #00  
1$: SUB #4, R5 ;RESET POINTER FOR ERROR LOOPS  
DEC $TMP7 ;RESET SUB TEST # FOR ERROR LOOP  
2$: CMP #PRTYTB+40, R5 ;DONE ALL 8. SUB TESTS ??  
BEQ 21$ ;:BR IF YES  
MOV #STACK, SP ;RESET STACK POINTER FOR ERROR LOOPS  
MOV DHADR, R1 ;RESET DEVADR FOR ERROR LOOPS  
MOV (R5)+, $TMP6 ;GET THE BYTE COUNT PARAMETER  
MOV (R5)+, $TMP5 ;GET THE LINE PARAMETERS  
INC $TMP7 ;GENERATE NEW SUB-TEST NO.  
MOV #BIT11, (R1) ;CLEAR THE DH11  
JSR PC, SUPPAR ;GO SET UP PARAMETERS  
MOV LPR(R1), $TMP0 ;SAVE CURRENT LINE PARAMETERS  
MOV LINSEL, $TMP3 ;SAVE SELECTED LINES PARAMETER  
JSR PC, CHPS2 ;GO LOCK OUT INTRs  
MOV DHVCT, R2 ;SET UP THE VECTOR  
MOV #3$, (R2)+ ;GO TO 3$ ON RCVR INTERRUPT  
MOVB DHRLVL, (R2)  
MOV #100, (R1) ;ENABLE CHAR AVAIL INTERRUPTS  
MOV LINSEL, LINACT ;FLAG ALL SELECTED LINES ACTIVE
```

```

6562 014326 013761 027312 000012      MOV    LINSEL, BAR(R1)  ;ACTIVATE ALL SELECTED LINES
6563 014334 113737 001102 001206      MOVB   $STNM, $TMP2    ;SAVE THE TEST NO.
6564 014342 042737 177400 001206      BIC    #177400, $TMP2
6565 014350 004737 027150      JSR    PC, CHPS1       ;GO CLEAR PSW
6566 014354 000137 014616      JMP    7$              ;GO WAIT FOR INTERRUPTS
6567
6568 014360 012706 001100      21$:  MOV    #STACK, SP    ;RESTORE THE SP
6569 014364 004737 027150      JSR    PC, CHPS1       ;GO CLEAR PSW
6570 014370 004737 026770      JSR    PC, RESTRP      ;RESTORE TRAP CATCHER
6571 014374 000556      BR     TST47           ;GO TO NEXT TEST
6572
6573      ;RECEIVER INTERRUPT SERVICE ROUTINE
6574
6575 014376 005037 001212      3$:   CLR    $TMP4
6576 014402 105711      TSTB   (R1)            ;CHAR AVAIL SET
6577 014404 100404      BMI    4$              ;BR IF YES
6578
6579 014406 012711 004000      MOV    #BIT11, (R1)    ;CLEAR OUT THE DH11
6580 014412 104041      ERROR  41              ;RCVR FALSE INTERRUPT - CHAR AVAIL NOT SET
6581 014414 000700      BR     2$              ;GO TRY NEXT SUB TEST
6582
6583 014416 032711 040000      4$:   BIT    #BIT14, (R1) ;SILO OVERFLOW ??
6584 014422 001404      BEQ    5$              ;BR IF NOT
6585
6586 014424 012711 004000      MOV    #BIT11, (R1)    ;CLEAR OUT THE DH11
6587 014430 104042      ERROR  42              ;SILO OVERFLOW ERROR
6588 014432 000671      BR     2$              ;GO TRY NEXT SUB TEST
6589
6590 014434 016103 000002      5$:   MOV    NRC(R1), R3   ;GET THE WAS DATA
6591 014440 010302      MOV    R3, R2          ;EXTRACT AND SAVE LINE NO.
6592 014442 000302      SWAB   R2
6593 014444 042702 177760      BIC    #177760, R2
6594 014450 006302      ASL    R2              ;GENERATE TABLE OFFSETR
6595 014452 005237 001212      INC    $TMP4
6596 014456 022737 000101 001212      CMP    #101, $TMP4
6597 014464 001744      BEQ    3$
6598 014466 036237 030246 027312      BIT    $LINSEL(R2), LINSEL ;IS THIS ONE OF THE SELECTED LINES?
6599 014474 001002      BNE    51$            ;IF SO, GO ANALYZE THE CHARACTER
6600 014476 104000      ERROR  ;INDICATE SOME KIND OF ERROR
6601 014500 000755      BR     5$              ;CHECK THE NEXT SILO ENTRY
6602 014502 010237 001212      51$:  MOV    R2, $TMP4
6603 014506 006237 001212      ASR    $TMP4
6604 014512 026203 036312      CMP    RBUF(R2), R3    ;CORRECT DATA RECEIVED ??
6605 014516 001426      BEQ    6$              ;BR IF YES
6606
6607 014520 004737 027200      JSR    PC, SAPS        ;SAVE THE ERROR PSW
6608 014524 012711 004000      MOV    #BIT11, (R1)    ;CLEAR OUT THE DH11
6609 014530 016204 036312      MOV    RBUF(R2), R4    ;SET UP S/B DATA
6610 014534 062701 000002      ADD    #NRC, R1        ;SET UP WAS ADDRESS
6611 014540 062702 036312      ADD    #RBUF, R2       ;SET UP S/B ADDRESS
6612 014544 004737 024416      JSR    PC, SUER2A     ;GO SET UP ERROR INFO
6613 014550 004537 024636      JSR    R5, SUNUM      ;PUT LINE NO. IN MESSAGE
6614 014554 001212      $TMP4
6615 014556 033144      EM34+51
6616 014560 004537 024636      JSR    R5, SUNUM      ;PUT SUBTEST NO. IN MESSAGE
6617 014564 001220      $TMP7

```

```

6618 014566 033162          EM34+67
6619 014570 104034          ERROR 34          ;PARITY DATA COMPARE ERROR
6620 014572 000611          BR 2$            ;GO TRY NEXT SUBTEST
6621
6622 014574 105262 036312    6$:  INCB  RBUF(R2)      ;GENERATE NEW RCVD DATA
6623 014600 005262 027562          INC  MULPTB(R2)     ;COUNT ONE BYTES RECEIVED
6624 014604 001003          BNE  61$         ;BR IF NOT DONE
6625 014606 046237 027520 027560 BIC  LINBIT(R2),LINACT ;FLAG THIS LINE DONE
6626 014614 000002          61$: RTI          ;RETURN TO WAIT ROUTINE
6627
6628          ;WAIT ROUTINE
6629
6630 014616 012737 000002 030350 7$:  MOV  #2,TIMEA      ;INIT TIMER A
6631 014624 005037 030352          CLR  TIMEB        ;INIT TIMER B
6632 014630 005761 000012          8$:  TST  BAR(R1)   ;ALL LINES DONE XMITTING ??
6633 014634 001413          BEQ  9$           ;BR IF YES
6634 014636 004737 027016          JSR  PC,TIMEIT    ;CALL THE TIMER
6635 014642 000772          BR  8$           ;TIMER ROUTINE WILL MOVE RETURN PC
6636          ;AROUND THIS BRANCH IF TIME OUT OCCURS
6637
6638 014644 016137 000012 001210 MOV  BAR(R1),$TMP3 ;SAVE THE ACTIVE LINES FLAG
6639 014652 012711 004000          MOV  #BIT11,(R1) ;CLEAR OUT THE DH11
6640 014656 104035          ERROR 35         ;TIMED OUT WAITING FOR TRANSMITTERS TO FINISH
6641 014660 000137 014216          JMP  2$          ;GO TRY NEXT SUBTEST
6642
6643
6644 014664 012737 000001 030350 9$:  MOV  #1,TIMEA      ;INIT TIMER A
6645 014672 005037 030352          CLR  TIMEB        ;INIT TIMER B
6646 014676 005737 027560          10$: TST  LINACT    ;ALL CHARS RECEIVED ?
6647 014702 001411          BEQ  11$         ;BR IF YES
6648 014704 004737 027016          JSR  PC,TIMEIT    ;CALL THE TIMER
6649 014710 000772          BR  10$          ;TIMER ROUTINE WILL MOVE RETURN PC
6650          ;AROUND THIS BRANCH IF TIME OUT OCCURS
6651
6652 014712 013737 027560 001210 MOV  LINACT,$TMP3 ;SET UP ACTIVE LINE PARAMETER
6653 014720 012711 004000          MOV  #BIT11,(R1) ;CLEAR OUT THE DH11
6654 014724 104035          ERROR 35         ;SILO EMPTY TIMEOUT
6655 014726 000137 014216          11$: JMP  2$          ;GO TRY NEXT SUB TEST
    
```

6656
6657
6658
6659 014732 000004
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677
6678
6679
6680
6681
6682
6683
6684
6685
6686
6687
6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703
6704
6705
6706
6707
6708
6709
6710
6711

*TEST 47 AUTO ECHO TEST 1 - ALL SELECTED LINES

TST47: SCOPE
.REM X
TEST ABSTRACT:

THIS TEST VERIFIES THAT ALL SELECTED LINES CAN TURN AROUND
A SINGLE TEST CHARACTER (377) AND (000) IN AUTO ECHO MODE. THE TEST SEQUENCE
IS AS FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. RETRIEVE THE AUTO-ECHO TEST DATA FROM "AETAB:" AND
UPDATE THE POINTER.
3. GO SELECT A LINE NO. TO TEST - GO TO STEP 10 IF DONE
ALL SELECTED LINES.
4. CLEAR THE "CAR" AND "BCR" MEMORIES.
5. PRIME THE SELECTED LINE TO XMIT ONE CHAR WITH A.E. ENABLED.
6. ACTIVATE THE SELECTED TRANSMITTER.
7. WAIT FOR "CHAR AVAIL" - IF TIMEOUT REPORT ERROR AND RESTART
AT STEP 2.
8. IF NO TIMEOUT - READ SILO AND COMPARE AUTO ECHO DATA RECEIVED
REPORT DATA COMPARE ERRORS AND RESTART AT STEP 2
9. IF NO ERRORS REPEAT STEPS 7 AND 8 SIXTY-FOUR TIMES THEN TURN
OFF A.E. ENABLE AND READ LAST CHAR FROM SILO.. CHECK LAST
CHAR FOR DATA COMPARE ERRORS - REPORT ERRORS IF ANY - AND RE-
START AT STEP 2.
10. CHANGE A.E. TABLE POINTER TO POINT TO "AETAB0:" (0'S DATA)
AND REPEAT STEPS 2 THRU 9.

ERRORS:

1. ERROR 24 IS CALLED TO REPORT ALL ERRORS

SYNC: M7277 SH4 LOAD BAR LB+HB L CN2

DEBUG:

1. IF ALL LINES FAIL, SUSPECT EITHER THE M7277 OR M7289
2. IF ONLY ONE LINE FAILS SUSPECT THE M7288
3. LOOP ON THE FAILING LINE AND TRACK BACK THROUGH THE KEY LOGIC.

KEY LOGIC:

M7277 SH3 AE GO L CS2
7402 "OR" GATE CHIPS E38 OR E41
74157 MUX CHIPS E39 OR 342
SH4 E35 - PIN 2 STUCK LOW

```

6712
6713
6714
6715
6716
6717
6718
6719
6720
6721
6722 014734 012737 015016 001110      %
6723 014742 005037 001210
6724 014746 012711 004000
6725 014752 012705 027420
6726 014756 005037 027316      7$:
6727 014762 000261
6728 014764 000401
6729 014766 000241      1$:
6730 014770 006137 027316      13$:
6731 014774 001407
6732 014776 012504
6733 015000 033737 027316 027312
6734 015006 001767
6735 015010 004737 024544      11$:
6736 015014      12$:
6737 015014 000522
6738 015016 004737 024716      2$:
6739 015022 113711 030322
6740 015026 012761 177777 000010
6741 015034 010561 000006
6742 015040 162761 000002 000006
6743 015046 012737 000100 001220
6744 015054 005037 001216
6745 015060 012761 133503 000004
6746 015066 013761 027314 000012
6747
6748 015074 012737 000002 030350
6749 015102 005037 030352
6750 015106 105711      3$:
6751 015110 100427
6752 015112 004737 027016
6753 015116 000773
6754
6755
6756 015120 004737 027200
6757 015124 005061 000004
6758 015130 010102
6759 015132 011103
6760 015134 042703 100000
6761 015140 012704 000200
6762 015144 153704 030322
6763 015150 004737 024416
6764 015154 004537 024636
6765 015160 030322
6766 015162 032274
6767 015164 104024

M7289 SH3 AE GO L EK1
AE SCAN MUX E22 PIN 10
SH4 SAMPLE STATUS H E21-12

M7288 SH5, 7, 9, 11
AE ENABLE 'NM' H CONTROL FLOPS
74174 CHIPS PIN 15

MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN
CLR $TMP3 ;INIT I/O DATA FLAG
MOV #BIT11, (R1) ;CLEAR THE DH11
MOV #AETAB, R5 ;GET POINTER TO AUTO ECHO DATA TABLE
CLR LMSK1 ;INIT BIT TEST MARKER
SEC ;SET 'C' BIT FOR MARKER
BR 13$ ;GO SHIFT MASK
CLC ;INIT THE 'C' BIT
ROL LMSK1 ;SHIFT BIT MARKER
BEQ 12$ ;BR IF DONE ALL LINES
MOV (R5)+, R4 ;SET UP THE S/B DATA
BIT LMSK1, LINSEL ;TEST THIS LINE ?
BEQ 1$ ;BR IF NOT
JSR PC, SELINE ;GO SELECT A LINE TO TEST

BR 6$ ;:BR IF ALL SELECTED LINES TESTED
JSR PC, CLCABC ;GO CLEAR 'CAR' AND 'BCR' MEMORIES
MOVB LINE, (R1) ;SET SELECT BITS IN SCR REG
MOV #-1, BCR(R1) ;SET UP TO XFER ONE CHAR
MOV R5, CAR(R1) ;SET UP THE BUS ADDRESS REG
SUB #2, CAR(R1) ;CORRECT BUS ADDRESS
MOV #100, $TMP7 ;COUNT 64 CHARS TO BE RECEIVED IN AUTO FCHO
CLR $TMP6 ;INIT CHAR COUNTER
MOV #133503, LPR(R1) ;SET UP LINE PARAMETER REG
MOV LINMSK, BAR(R1) ;ACTIVATE THE LINE

MOV #2, TIMEA ;INIT TIMER A
CLR TIMEB ;INIT TIMERB
TSTB (R1) ;CHAR AVAIL SET ??
BMI 4$ ;BR IF YES
JSR PC, TIMEIT ;CALL THE TIMER
BR 3$ ;TIMER ROUTINE WILL MOVE RETURN PC
;AROUND THIS BRANCH IF TIME OUT OCCURS

JSR PC, SAPS ;SAVE THE ERROR PSW
CLR LPR(R1) ;TURN OFF AUTO ECHO MODE
MOV R1, R2 ;MAKE REGADR = DEVADR
MOV (R1), R3 ;GET THE WAS DATA
BIC #BIT15, R3 ;CLEAR JUNK BIT
MOV #200, R4 ;SET UP S/B DATA
BISB LINE, R4
JSR PC, SUER2A ;GO SET UP ERROR INFO
JSR R5, SUNUM ;GO SET LINE NO. IN MSG
LINE
EM24+35
ERROR 24 ;DATA AVAIL FAILED TO SET ON TIME

```



```

6768 015156 000677          BR      1$          ;GO TRY NEXT LINE
6769
6770 015170 005237 001216      4$:    INC      $TMP6          ;COUNT ONE CHAR RECVD
6771 015174 016103 000002      MOV      NRC(R1),R3      ;GET THE WAS DATA
6772 015200 020304          CMP      R3,R4          ;WAS CHAR AUTO ECHOED CORRECTLY ?
6773 015202 001417          BEQ      5$          ;BR IF YES
6774
6775 015204 004737 027200      JSR      PC,SAPS        ;SAVE THE ERROR PSW
6776 015210 005061 000004      CLR      LPR(R1)        ;DISABLE AUTO ECHO
6777 015214 010102          MOV      R1,R2          ;SET UP REGADR
6778 015216 062702 000002      ADD      #NRC,R2
6779 015222 004737 024416      JSR      PC,SUER2A      ;GO SET UP ERROR INFO
6780 015226 004537 024636      JSR      R5,SUNUM       ;PUT LINE NO. IN ERROR MSG
6781 015232 030322          LINE
6782 015234 032274          EM24+35
6783 015236 104024          ERROR  24
6784 015240 000652          BR      1$          ;CHAR AUTO ECHOED INCORRECTLY
6785                                     ;GO TRY NEXT LINE
6786 015242 005337 001220      5$:    DEC      $TMP7          ;COUNT ONE CHAR READ OUT OF 64
6787 015246 003317          BGT      3$          ;BR IF NOT LAST ONE
6788 015250 100646          BMI      1$          ;BR IF LAST ONE READ
6789 015252 042761 100000 000004 BIC      #BIT15,LPR(R1) ;DISABLE AUTO ECHO
6790 015260 000712          BR      3$          ;GO READ LAST CHAR
6791
6792 015262 005137 001210      6$:    COM      $TMP3          ;TOGGLE 1/0 FLAG
6793 015266 001406          BEQ      TST50          ;BR IF DONE BOTH 1/0 DATA
6794 015270 005037 030322      CLR      LINE          ;INIT LINE NO TO 00
6795 015274 012705 027460      MOV      #AETAB0,R5     ;SET POINTER TO 0'S TABLE
6796 015300 000137 014756      JMP      7$          ;REPEAT TEST FOR ZERO PATTERNS
6797

```

6798
6799
6800
6801 015304 000004
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831 015306 012737 015362 001110
6832 015314 012705 027420
6833 015320 005037 027316
6834 015324 000261
6835 015326 000401
6836 015330 000241
6837 015332 006137 027316
6838 015336 001410
6839 015340 012537 001220
6840 015344 033737 027316 027312
6841 015352 001766
6842 015354 004737 024544
6843 015360
6844 015360 000575
6845
6846 015362 013701 027302
6847 015366 012711 004000
6848 015372 004737 027040
6849
6850 015376 153711 030322
6851 015402 010561 000006
6852 015406 162761 000002 000006
6853 015414 012761 177777 000010

```
*****
:*TEST 50      AUTO ECHO TEST 2 - ALL SELECTED LINES
*****
TST50: SCOPE
.REM %
TEST ABSTRACT:
*****
```

THIS TEST IS SIMILAR TO TEST 47 EXCEPT ALL SELECTED LINES OTHER THAN THE A.E. TEST LINE ARE ACTIVELY TURNING AROUND A BINARY COUNT TEST PATTERN IN NON-AUTO ECHO MODE AND THE A.E. TEST LINE IS TESTED FOR ALL 1'S DATA ONLY.

ERRORS:

- 1. ERROR 32 IS CALLED TO REPORT A.E. TEST TIMEOUTS
- 2. ERROR 31 IS CALLED TO REPORT ALL DATA COMPARE ERRORS

SYNC: M7277 SH4 LOAD BAR LB+HB L CN2

DEBUG:

REFER TO TEST 47

KEY LOGIC:

REFER TO TEST 47

```
%
MOV #2$, $LPER, ;SET UP ERROR LOOP RETURN
MOV #AETAB, R5 ;SET POINTER TO A.E. TEST DATA TABLE
CLR LMSK1 ;INIT BIT TEST MASK
SEC ;GENERATER MARKER BIT IN "C"
BR 12$ ;GO SHIFT MASK
1$: CLC ;INIT THE "C" BIT
12$: ROL LMSK1 ;SHIFT TEST BIT
BEQ 11$ ;BR IF TESTED ALL LINES
MOV (R5)+, $TMP7 ;GET THE A.E. TEST DATA FOR THIS LINE
BIT LMSK1, LINSSEL ;TEST THIS LINE ?
BEQ 1$ ;BR IF NOT
JSR PC, SELINE ;GO SELECT A LINE
11$: BR TST51 ;:BR IF DONE ALL SELECTED LINES
2$: MOV DHADR, R1 ;RESET DEVADR IN CASE OF ERROR LOOP
MOV #BIT11, (R1) ;CLEAR OUT THE DH11
JSR PC, SETALL ;GO SET UP FOR BINARY COUNT XFER ON
;ALL LINES OTHER THAN THE SELECTED ONE
BISB LINE, (R1) ;SELECT THE LINE FOR A.E. TEST
MOV R5, CAR(R1) ;SET BUS ADDR TO XMIT TEST CHAR
SUB #2, CAR(R1) ;CORRECT THE ADDRESS
MOV #-1, BCR(R1) ;XMIT ONE CHAR ON THIS LINE
```

```

6854 015422 012761 133503 000004      MOV      #133503,LPR(R1) ;DO IT AT 9600 BAUD/8 BITS
6855 015430 113737 001102 001206      MOV      $STNM,$TMP2 ;SAVE THE TEST NO.
6856 015436 042737 177400 001206      BIC      #177400,$TMP2
6857 015444 043737 027314 027560      BIC      LINMSK,LINACT ;MAKE THIS LINE APPEAR INACTIVE
6858 015452 013761 027312 000012      MOV      LINSEL,BAR(R1) ;ACTIVATE ALL SELECTED TRANSMITTERS
6859
6860 015460 012737 000002 030350 21$:      MOV      #2,TIMEA ;INIT TIMER A
6861 015466 005037 030352          CLR      TIMEB ;INIT TIMER B
6862 015472 016103 000002          MOV      NRC(R1),R3 ;GET THE WAS DATA
6863 015476 100414          BMI      4$ ;BR IF YES
6864 015500 004737 027016          JSR      PC,TIMEIT ;CALL THE TIMER
6865 015504 000772          BR       3$ ;TIMER ROUTINE WILL MOVE RETURN PC
6866                                     ;AROUND THIS BRANCH IF TIME OUT OCCURS
6867
6868 015506 016137 000004 001202      MOV      LPR(R1),$TMP0 ;SAVE THE CURRENT 'LPR'
6869 015514 004537 024636          JSR      R5,SUNUM ;PUT LINE NO. IN MESSAGE
6870                                     LINE
6871 015522 032765          EM32+35
6872 015524 104032          ERROR   32 ;AUTO ECHO TIMEOUT
6873 015526 000700          BR       1$ ;GO TRY NEXT LINE
6874
6875                                     4$:      MOV      R3,R4 ;EXTRACT LINE NUMBER OF RCVD CHAR
6876 015532 000304          SWAB    R4
6877 015534 042704 177760          BIC      #177760,R4
6878 015540 010402          MOV      R4,R2 ;SAVE IT IN R2
6879 015542 006302          ASL     R2 ;GENERATE TABLE INDEX IN R2
6880 015544 123704 030322          CMPB    LINE,R4 ;IS THIS THE A.E. TEST LINE ??
6881 015550 001432          BEQ     5$ ;BRANCH IF YES.
6882 015552 036237 030246 027312      BIT     $LNSEL(R2),LINSEL
6883 015560 001737          BEQ     21$
6884
6885 015562 026203 036312          CMP     RBUF(R2),R3 ;RECVD DATA CORRECT ??
6886 015566 001447          BEQ     6$ ;BR IF IT WAS
6887
6888 015570 004737 027200          JSR     PC,SAPS ;SAVE THE ERROR PSW
6889 015574 010437 001214          MOV     R4,$TMP5 ;SAVE THE LINE NUMBER
6890 015600 016204 036312          MOV     RBUF(R2),R4 ;SET UP S/B DATA
6891 015604 062702 036312          ADD     #RBUF,R2 ;SET UP S/B ADDRESS
6892 015610 012701 177703          MOV     #177703,R1 ;SET UP THE WAS ADDRESS
6893 015614 004737 024416          JSR     PC,SUER2A ;GO SET UP ERROR INFO
6894 015620 004537 024636          JSR     R5,SUNUM ;PUT LINE NO. IN MESSAGE
6895 015624 001214          $TMP5
6896 015626 032630          EM31+45
6897 015630 104031          ERROR   31 ;NON-ECHO DATA COMPARE ERROR
6898 015632 000137 015330          JMP     1$ ;GO TRY NEXT LINE
6899
6900
6901 015636 020337 001220          5$:      CMP     R3,$TMP7 ;CHAR ECHOED OK ??
6902 015642 001427          BEQ     7$ ;BR IF YES
6903
6904 015644 004737 027200          JSR     PC,SAPS ;SAVE THE ERROR PSW
6905 015650 012702 001220          MOV     #$TMP7,R2 ;SAVE THE S/B ADDRESS
6906 015654 013704 001220          MOV     $TMP7,R4 ;SAVE THE S/B DATA
6907 015660 012701 177703          MOV     #177703,R1 ;SAVE THE WAS ADDRESS
6908 015664 004737 024416          JSR     PC,SUER2A ;GO SET UP ERROR INFO
6909 015670 004537 024636          JSR     R5,SUNUM ;GO SET UP LINE NO. IN MESSAGE

```

6910	015674	030322				LINE				
6911	015676	032630				EM31+45				
6912	015700	104031				ERROR	31			; AUTO ECHO LINE DATA ERROR
6913	015702	000137	015330			JMP	1\$; GO TRY NEXT LINE
6914										
6915	015706	105262	036312		6\$:	INCB	RBUF(R2)			; GENERATE NEXT EXPECTED DATA ON THIS LINE
6916	015712	001262				BNE	21\$; BR IF ITS NOT BACK TO 000
6917	015714	046237	027520	027560		BIC	LINBIT(R2), LINACT			; INDICATE THIOS LINE DONE 256 BYTES
6918	015722	005737	027560		7\$:	TST	LINACT			; ALL LINES INACTIVE
6919	015726	001254				BNE	21\$; BR IF NOT
6920	015730	042761	100000	000004		BIC	#BIT15, LPR(R1)			; TURN OFF THE A.E. BIT
6921	015736	105761	000017			TSTB	SSR+1(R1)			; SILO EMPTY ??
6922	015742	001002				BNE	8\$; BR IF NOT
6923	015744	000137	015330			JMP	1\$; GO TEST NEXT LINE
6924	015750	000137	015460		8\$:	JMP	21\$; GO EMPTY IT

6925
6926
6927
6928 015754 000004
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943
6944
6945
6946
6947
6948
6949
6950
6951
6952
6953
6954
6955
6956
6957 015756 012737 015764 001110
6958 015764 012711 004000
6959 015770 012705 000020
6960 015774 012702 027420
6961 016000 012703 036252
6962 016004 010261 000006
6963 016010 012761 177777 000010
6964 016016 012761 131403 000004
6965 016024 005023
6966 016026 062702 000002
6967 016032 005211
6968 016034 005305
6969 016036 001362
6970 016040 113737 001102 001206
6971 016046 042737 177400 001206
6972 016054 013737 027312 027560
6973 016062 013761 027312 000012
6974
6975 016070 012737 000002 030350
6976 016076 005037 030352
6977 016102 005037 001212
6978 016106 105711
6979 016110 100410
6980 016112 004737 027016

*TEST 51 AUTO ECHO TEST 3 - ALL SELECTED LINES

TST51: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST IS IDENTICAL TO TEST 47 EXCEPT ALL SELECTED LINES
ARE ACTIVATED CONCURRENTLY RATHER THAN ONE AT A TIME AND ONLY
THE ALL 1'S DATA IS USED.

ERRORS:

- 1. ERROR 36 IS CALLED TO REPORT "DATA AVAIL" TIMEOUTS
- 2. ERROR 31 IS CALLED TO REPORT A.E. DATA ERRORS

SYNC: M7277 SH4 LOAD BAR LB+HB L CN2

DEBUG:

REFER TO TEST 47

KEY LOGIC:

REFER TO TEST 47

%
1\$: MOV #1\$, \$LPERR ;SET UP THE ERROR LOOP RETURN
MOV #BIT11, (R1) ;CLEAR OUT THE DH11
MOV #20, R5 ;INIT COUNTER TO SET UP 16. LINES
MOV #AETAB, R2 ;SET UP POINTER TO AUTO ECXHO TEST DATA
MOV #RCNT, R3 ;R3 POINTS TO TABLE OF CHAR COUNTERS
2\$: MOV R2, CAR(R1) ;SET UP BUS ADDRESS REG
MOV #-1, BCR(R1) ;SET UP BYTE COUNT REG
MOV #131403, LPR(R1) ;SET UP LINE PARAMETERS
CLR (R3)+ ;CLEAR A COUNTER
ADD #2, R2 ;UPDATE POINTERS
INC (R1) ;SELECT NEXT LINE
DEC R5 ;COUNT ONE DONE
BNE 2\$;BR TILL 16. DONE
MOVB \$TSTNM, \$TMP2 ;SAVE THE TEST NO.
BIC #177400, \$TMP2
MOV LINSEL, LINACT ;SET FLAG TO INDICATE ALL 16. ACTIVE
MOV LINSEL, BAR(R1) ;ACTIVATE ALL XMITTERS
3\$: MOV #2, TIMEA ;INIT TIMER A
CLR TIMEB ;INIT TIMERB
CLR \$TMP4
TSTB (R1) ;CHAR AVAIL SET YET ?
BMI 4\$;BR IF YES
JSR PC, TIMEIT ;CALL THE TIMER

```

6981 016116 000771          BR      3$          ;TIMER ROUTINE WILL MOVE RETURN PC
6982                                     ;AROUND THIS BRANCH IF TIME OUT OCCURS
6983
6984 016120 016137 000004 001202  MOV     LPR(R1), $TMP0  ;SAVE THE 'LPR' REG
6985 016126 104036          ERROR   36          ;DATA AVAILABLE TIMEOUT
6986 016130 000467          BR      TST52         ;:EXIT TEST ON ERROR
6987
6988 016132 016103 000002          4$:   MOV     NRC(R1), R3      ;GET THE WAS DATA
6989 016136 010302          MOV     R3, R2         ;BUILD AND SAVE LINE NO.
6990 016140 000302          SWAB   R2
6991 016142 042702 177760          BIC    #177760, R2
6992 016146 010237 001216          MOV     R2, $TMP6     ;SAVE THE LINE NO.
6993 016152 006302          ASL    R2             ;GENERATE TABLE OFFSET
6994 016154 005237 001212          INC     $TMP4
6995 016160 022737 000101 001212  CMP     #101, $TMP4
6996 016166 001745          BEQ    3$
6997 016170 036237 030246 027312  BIT     $LNSEL(R2), LNSEL ;IS THIS ONE OF THE SELECTED LINES?
6998 016176 001002          BNE    41$          ;IF SO, GO ANALYZE THE CHARACTER
6999 016200 104000          ERROR   ;INDICATE SOME KIND OF ERROR
7000 016202 000753          BR      4$          ;GO GET THE NEXT CHARACTER
7001 016204 005262 036252          41$:  INC     RCNT(R2)       ;COUNT THE CHARACTER
7002 016210 020362 027420          CMP     R3, AETAB(R2) ;IS THE DATA CORRECT ??
7003 016214 001420          BEQ    5$          ;BR IF YES
7004
7005 016216 004737 027200          JSR    PC, SAPS      ;SAVE THE ERROR PSW
7006 016222 016204 027420          MOV     AETAB(R2), R4 ;GET THE S/B DATA
7007 016226 062702 027420          ADD    #AETAB, R2    ;GENERATE S/B ADDRESS
7008 016232 062701 000002          ADD    #NRC, R1      ;GENERATE THE WAS ADDRESS
7009 016236 004737 024416          JSR    PC, SUER2A    ;GO SET UP ERROR INFO
7010 016242 004537 024636          JSR    R5, SUNUM     ;PUT LINE NO. IN MESSAGE
7011 016246 001216          $TMP6
7012 016250 032630          EM31+45
7013 016252 104031          ERROR   31          ;DATA COMPARE ERROR
7014 016254 000415          BR      TST52         ;:EXIT TEST ON ERROR
7015
7016 016256 022762 000100 036252  5$:   CMP     #100, RCNT(R2) ;DONE 64. CHARS ON THIS LINE ?
7017 016264 001306          BNE    3$          ;BR IF NOT
7018 016266 013711 001216          MOV     $TMP6, (R1)   ;SELECT LINE IN SCR REG
7019 016272 042761 100000 000004  BIC    #BIT15, LPR(R1) ;TURN OFF A.E. BIT
7020 016300 046237 027520 027560  BIC    LINBIT(R2), LINACT ;ALL LINES INACTIVE ??
7021 016306 001275          BNE    3$          ;BR IF NOT
    
```

7022
7023
7024
7025 016310 000004
7026
7027
7028
7029
7030
7031
7032
7033
7034
7035
7036
7037
7038
7039
7040
7041
7042
7043
7044
7045
7046
7047
7048
7049
7050
7051
7052
7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064
7065
7066
7067
7068
7069
7070
7071
7072
7073
7074
7075
7076
7077

: *TEST 52 BREAK BIT TEST - ALL SELECTED LINES
: *****
TST52: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT THE 'BREAK' FEATURE WORKS PROPERLY
FOR ALL SELECTED LINES. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. RETRIEVE THE CORRECT S/B DATA FROM THE 'BREAK' DATA TABLE AND UPDATE THE POINTER.
3. GO SELECT A LINE TO TEST - GO TO THE NEXT TEST IF DONE ALL SELECTED LINES
4. RESET THE DH11 AND CLEAR THE 'CAR' AND 'BCR' MEMORIES.
5. PRIME SELECTED LINE TO OUTPUT TWO 'NULL' CHARS TO CLEAR UART
6. ACTIVATE THE SELECTED LINE
7. WAIT FOR SILO TO RECEIVE TWO NULLS - IF TIMEOUT REPORT ERROR AND RESTART AT STEP 2
8. IF NO TIMEOUT CLEAR THE SELECTED DH11 AND RESELECT LINE NO.
9. PRIME SELECTED LINE TO OUT PUT 256. CHARS.
10. SET THE SELECTED LINE'S BREAK BIT
11. ACTIVATE THE SELECTED LINE
12. WAIT FOR 'BAR' REG TO CLEAR -F TIMEOUT REPORT ERROR AND RESTART AT STEP 2
13. IF NO TIMEOUT VERIFY THAT THE SILO RECEIVED ONLY ONE CHAR- IF NOT REPORT ERROR AND RESTART AT STEP 2
14. IF SILO RECEIVED ONLY ONE CHAR VERIFY THAT IT WAS A 'BREAK' CHAR - IF NOT REPORT ERROR - AND RESTART AT STEP 2

ERRORS:

1. ERROR 25 IS CALLED TO REPORT ALL ERRORS

SYNC: M7277 SH4 LOAD BCR H FU1

DEBUG:

1. IF ALL LINES FAILED SUSPECT THAT THE M7277 IS NOT GENERATING THE BREAK CONTROL REG LOAD SIGNAL.
2. IF ONLY ONE LINE FAILS SUSPECT THE BREAK CONTROL LOGIC ON THE M7278

KEY SIGNALS:

M7277 SH4 LOAD BCR H FU1

M7278 SH5 THRU SH8

74175 REGISTER CHIPS E51, E38, E67, E60

7400 DRIVERS E45, E46, E75, E76

```

7078
7079
7080
7081 016312 012737 016372 001110 X MOV #2$,SLPERR ;SET UP ERROR LOOP RETURN
7082 016320 012705 027622 MOV #BRKTAB,R5 ;SET UP POINTER TO BREAK DATA TABLE
7083 016324 005037 027316 CLR LMSK1 ;INIT BIT TEST MASK
7084 016330 000261 SEC ;SET BIT MARKER IN 'C'
7085 016332 000401 BR 12$ ;GO SHIFT MASK
7086 016334 000241 1$: CLC ;INIT THE 'C' BIT
7087 016336 006137 027316 12$: ROL LMSK1 ;SHIFT TEST MARKER
7088 016342 001411 BEQ 11$ ;BR IF ALL LINES DONE
7089 016344 012504 MOV (R5)+,R4 ;GET TEST DATA FOR THIS LINE
7090 016346 033737 027316 027312 BIT LMSK1,LINSEL ;LINE SELECTED ?
7091 016354 001767 BEQ 1$ ;BR IF NOT
7092 016356 004737 024544 JSR PC,SELINE ;GO SELECT A LINE TO TEST
7093 016362 000401 BR 11$ ;BR IF DONE ALL SELECTED LINES
7094 016364 000402 BR 2$ ;GO TEST THE SELECTED LINE
7095 016366 000137 017046 11$: JMP 9$ ;GO EXIT TEST
7096 016372 012711 004000 2$: MOV #BIT11,(R1) ;CLEAR THE DH11
7097 016376 004737 024716 JSR PC,CLCABC ;GO CLR THE 'CAR' AND 'BCR' MEMORIES
7098 016402 113711 030322 MOVB LINE,(R1) ;SELECT THE LINE
7099
7100 016406 012761 030356 000006 MOV #TNUL, CAR(R1) ;SET UP TO OUTPUT TWO NULL CHARS
7101 016414 012761 177776 000010 MOV #-2,BCR(R1) ;SET BYTE COUNT TO 2
7102 016422 012761 033503 000004 MOV #33503,LPR(R1) ;SET UP LINE PARAMETERS
7103 016430 013761 027314 000012 MOV LINMSK,BAR(R1) ;ACTIVATE SELECTED LINE
7104
7105 016436 012737 000001 030350 MOV #1,TIMEA ;INIT TIMER A
7106 016444 005037 030352 CLR TIMEB ;INIT TIMER B
7107 016450 122761 000002 000017 3$: CMPB #2,SSR+1(R1) ;TWO CHARS RECEIVED ??
7108 016456 001432 BEQ 4$ ;BR IF YES
7109 016460 004737 027016 JSR PC,TIMEIT ;CALL THE TIMER
7110 016464 000771 BR 3$ ;TIMER ROUTINE WILL MOVE RETURN PC
7111 ;AROUND THIS BRANCH IF TIME OUT OCCURS
7112
7113 016466 004737 027200 JSR PC,SAPS ;SAVE THE ERROR PSW
7114 016472 010437 001204 MOV R4,$TMP1 ;SAVE S/B DATA
7115 016476 010102 MOV R1,R2 ;SET UP REGADR
7116 016500 062702 000016 ADD #SSR,R2
7117 016504 011203 MOV (R2),R3 ;GET THE WAS DATA
7118 016506 042703 100377 BIC #100377,R3 ;CLEAR JUNK
7119 016512 012704 000002 MOV #2,R4 ;SET UP S/B DATA
7120 016516 000304 SWAB R4
7121 016520 004737 024416 JSR PC,SUER2A ;GO SET UP ERROR INFO
7122 016524 004537 024636 JSR R5,SUNUM ;GO PUT LINE NO. IN MESSAGE
7123 016530 030322 LINE
7124 016532 032333 EM25+34
7125 016534 013704 001204 MOV $TMP1,R4 ;RESTORE S/B DATA
7126 016540 104025 ERROR 25 ;TIMED OUT WAITING FOR TWO NULLS
7127 016542 000674 BR 1$ ;GO TRY NEXT LINE
7128
7129 016544 012711 004000 4$: MOV #BIT11,(R1) ;CLEAR THE INTERFACE
7130 016550 113711 030322 MOVB LINE,(R1) ;SELECT THE LINE
7131 016554 012761 037312 000006 MOV #TBUF,CAR(R1) ;SET UP BUS ADDRESS REG FOR XMITTR
7132 016562 012761 177400 000010 MOV #-400,BCR(R1) ;SET BYTE COUNT TO XMIT 256(10) CHARS
7133 016570 012761 033503 000004 MOV #33503,LPR(R1) ;SET UP LINE PARAMETERS
  
```


7134	016576	013761	027314	000014	MOV	LINMSK,BKR(R1)	;SET BREAK BIT FOR ACTIVE LINE
7135	016604	013761	027314	000012	MOV	LINMSK,BAR(R1)	;ACTIVATE THE SELECTED LINE
7136							
7137	016612	012737	000005	030350	MOV	#5,TIMEA	;INIT TIMER A
7138	016620	005037	030352		CLR	TIMEB	;INIT TIMER B
7139	016624	005761	000012		5\$: TST	BAR(R1)	;BAR BIT CLEARED ??
7140	016630	001426			BEQ	6\$;BR IFD YES
7141	016632	004737	027016		JSR	PC,TIMEIT	;CALL THE TIMER
7142	016636	000772			BR	5\$;TIMER ROUTINE WILL MOVE RETURN PC
7143							;AROUND THIS BRANCH IF TIME OUT OCCURS
7144							
7145	016640	004737	027200		JSR	PC,SAPS	;SAVE THE ERROR PSW
7146	016644	010437	001204		MOV	R4,\$TMP1	;SAVE THE S/B DATA
7147	016650	010102			MOV	R1,R2	;SET UP REGADR
7148	016652	062702	000012		ADD	#BAR,R2	
7149	016656	011203			MOV	(R2),R3	;GET THE WAS DATA
7150	016660	005004			CLR	R4	;SET UP S/B DATA
7151	016662	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
7152	016666	004537	024636		JSR	R5,SUNUM	;PUT LINE NO IN MESSAGE
7153	016672	030322			LINE		
7154	016674	032333			EM25+34		
7155	016676	013704	001204		MOV	\$TMP1,R4	;RESTORE THE S/B DATA
7156	016702	104025			ERROR	25	;BAR BIT FAILED TO CLEAR
7157	016704	000613			BR	1\$;GO TRY NEXT LINE
7158							
7159	016706	122761	000001	000017	6\$: CMPB	#1,SSR+1(R1)	;ONE CHAR RECEIVED ?
7160	016714	001430			BEQ	7\$;BR IF YES
7161							
7162	016716	004737	027200		JSR	PC,SAPS	;SAVE THE ERROR PSW
7163	016722	010437	001204		MOV	R4,\$TMP1	;SAVE THE S/B DATA
7164	016726	010102			MOV	R1,R2	;SET UP REGADR
7165	016730	062702	000016		ADD	#SSR,R2	
7166	016734	011203			MOV	(R2),R3	;GET THE WAS DATA
7167	016736	042703	100377		BIC	#100377,R3	;CLEAR JUNK
7168	016742	012704	000001		MOV	#1,R4	;SET UP S/B DATA
7169	016746	000304			SWAB	R4	
7170	016750	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
7171	016754	004537	024636		JSR	R5,SUNUM	;GO PUT LINE NO. IN MESSAGE
7172	016760	030322			LINE		
7173	016762	032333			EM25+34		
7174	016764	013704	001204		MOV	\$TMP1,R4	;RESTORE THE S/B DATA
7175	016770	104025			ERROR	25	;FAILED TO RECEIVE THE ONE CHAR
7176	016772	000137	016334		JMP	1\$;GO TRY NEXT LINE
7177							
7178							
7179	016776	016103	000002		7\$: MOV	NRC(R1),R3	;GET THE WAS DATA
7180	017002	020304			CMP	R3,R4	;WAS IT A BREAK CHAR ?
7181	017004	001002			BNE	8\$;BR IF NOT CORRECT
7182	017006	000137	016334		JMP	1\$;GO TEST NEXT LINE
7183							
7184	017012	004737	027200		8\$: JSR	PC,SAPS	;SAVE THE ERROR PSW
7185	017016	010102			MOV	R1,R2	;SET UP REGADR
7186	017020	062702	000002		ADD	#NRC,R2	
7187	017024	004737	024416		JSR	PC,SUER2A	;GO SET UP ERROR INFO
7188	017030	004537	024636		JSR	R5,SUNUM	;PUT LINE NO IN MESSAGE
7189	017034	030322			LINE		

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78
152

08:05 PAGE 171
BREAK BIT TEST - ALL SELECTED LINES

M 13

SEQ 0169
SEQ 0168

7190 017036 032333
7191 017040 104025
7192 017042 000137 016334
7193 017046 000240
7194

9%:

EM25+34
ERROR 25
JMP 18
NOP

:INCORRECT DATA RECEIVED
:GO TRY NEXT LINE
:EXIT THIS TEST

12
12

7195
7196
7197
7198 017050 000004
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229
7230
7231
7232
7233
7234
7235
7236
7237
7238
7239
7240
7241
7242
7243
7244 017052 012737 017066 001110
7245 017060 004737 024544
7246 017064 000477
7247 017066 012711 004000
7248 017072 004737 024716
7249 017076 153711 030322
7250 017102 012761 037312 000006

```
*****  
: *TEST 53 HALF DUPLEX TEST - ALL SELECTED LINES  
: *****  
TST53: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT THE RECEIVERS ON ALL SELECTED LINES ARE 'BLINDED' WHEN THE HALF-DUPLEX MODE IS ENABLED. THE TEST SEQUENCE IS AS FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. GO SELECT A LINE NO. TO TEST
3. IF DONE ALL SELECTED LINES - GO TO THE NEXT TEST
4. RESET THE DH11 AND CLEAR THE "CAR" AND "BCR" MEMORIES
5. PRIME THE SELECTED DH11 TO XMIT 256. CHARS IN HALF-DUPLEX MODE.
6. ACTIVATE THE SELECTED LINE AND WAIT FOR THE "BAR" REG TO CLEAR
7. IF TIMEOUT - REPORT ERROR AND GO TO STEP 2
8. IF NO TIMEOUT VERIFY THE "CHAR AVAIL" DID NOT SET (RECEIVER BLINDED) - IF ERROR REPORT IT AND GO TO STEP2 - IF NO ERROR GO TO STEP 2

ERRORS:

1. ERROR 26 IS CALLED TO REPORT ALL ERRORS

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. SUSPECT EITHER THE M7289 OR THE M7288 MODULES

KEY LOGIC:

```
M7289 SH5 HALF DUPLEX <15:00> H SIGNALS  
END OF CHAR <15:00> SIGNALS  
  
M7288 SH5 HALF DUPLEX <03:00> H SIGNALS  
SH7 " " <07:00> H SIGNALS  
SH9 " " <11:08> H SIGNALS  
SH11 " " <15:12> H SIGNALS
```

```
%  
1$: MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN  
JSR PC, SELINE  
BR TST54 ;:BR IF ALL LINES TESTED  
2$: MOV #BIT11, (R1) ;CLEAR THE INTERFACE  
JSR PC, CLCABC ;GO CLR THE "CAR" AND "BCR" MEMORIES  
BISB LINE, (R1) ;SELECT THE LINE  
MOV #TBUF, CAR(R1) ;POINT TO XMIT BUFFER
```

```
7251 017110 012761 177400 000010      MOV      #-400,BCR(R1)      ;XMIT 256(10) CHARS
7252 017116 012761 073503 000004      MOV      #73503,LPR(R1)    ;SET UP THE LINE PARAMETERS
7253 017124 013761 027314 000012      MOV      LINMSK,BAR(R1)    ;ACTIVATE THE SELECTED LINE
7254
7255 017132 012737 000001 03C350      MOV      #1,TIMEA          ;INIT TIME A
7256 017140 005037 030352              CLR      TIMEB              ;INIT TIME B
7257 017144 005761 000012      3$:    TST      BAR(R1)        ;WAIT FOR XMITTR TO FINISH
7258 017150 001423              BEQ      4$                 ;BR IF XMITTR FINISHED
7259 017152 004737 027016      JSR      PC,TIMEIT         ;CALL TIMER
7260 017156 000772              BR       3$                 ;TIMER WILL MOVE RETURN PC AROUND
7261
7262
7263 017160 004737 027200      JSR      PC,SAPS           ;SAVE THE ERROR PSW
7264 017164 016103 000012      MOV      BAR(R1),R3        ;GET THE WAS DATA
7265 017170 010102              MOV      R1,R2             ;SET UP REGADR
7266 017172 062702 000012      ADD      #BAR,R2
7267 017176 005004              CLR      R4                 ;SET UP NEW S/B DATA
7268 017200 004737 024416      JSR      PC,SUER2A         ;GO SET UP THE ERROR INFO
7269 017204 004537 024636      JSR      R5,SUNUM          ;PUT LINE NO. IN MESSAGE
7270 017210 030322              LINE
7271 017212 032376              EM26+37
7272 017214 104026              ERROR  26                   ;BAR BIT FAILED TO CLEAR ON TIME
7273 017216 000720              BR       1$                 ;GO TRY NEXT LINE
7274
7275 017220 105711      4$:    TSTB     (R1)            ;CHAR AVAIL SET ??
7276 017222 100316              BPL      1$                 ;BR IF NOT IT SHOULDN'T BE
7277
7278 017224 004737 027200      JSR      PC,SAPS           ;SAVE THE ERROR PSW
7279 017230 010102              MOV      R1,R2             ;SET UP REGADR
7280 017232 011103              MOV      (R1),R3           ;GET WAS DATA
7281 017234 042703 100000      BIC      #BIT15,R3         ;CLEAR JUNK BIT
7282 017240 113704 030322      MOV      LINE,R4           ;SET UP S/B DATA
7283 017244 004737 024416      JSR      PC,SUER2A         ;GO SETUP ERROR INFO
7284 017250 004537 024636      JSR      R5,SUNUM          ;PUT LINE NO. IN MSG
7285 017254 030322              LINE
7286 017256 032376              EM26+37
7287 017260 104026              ERROR  26                   ;HALF DUPLEX FAILED TO BLIND RECVR
7288 017262 000676              BR       1$                 ;GO SELECT NEXT LINE
```

7289
7290
7291
7292 017264 000004
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317
7318
7319
7320
7321
7322
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342
7343
7344

```
*****  
: *TEST 54 VERIFY THAT OVERRUN CAN SET PROPERLY - ALL SELECTED LINES  
: *****  
TST54: SCOPE  
.REM %  
TEST ABSTRACT:  
*****
```

THIS TEST VERIFIES THAT "OVERRUN" SETS PROPERLY FOR ALL LINES THAT ARE SELECTED FOR TEST WHEN THE OVERRUN CONDITION IS FORCED BY THE PROGRAM. THE TEST SFQUENCE IS AS FOLLOWS:

1. SET UP THE ERROR LOOP RETURN
2. SELECT A LINE NO. TO TEST - IF DONE ALL LINES GO TO THE NEXT TEST.
3. PRIME THE SELECTED LINE TO XMIT 68. CHARS
4. ACTIVATE THE SELECTED LINE
5. WAIT FOR "XMIT DONE" TO SET - IF TIMEOUT REPORT ERROR AND RESTART AT STEP 2
6. IF NO TIMEOUT READ 65. CHARS FROM THE SILO AND VERIFY THAT "OVERRUN" IS SET ON THE LAST WORD READ
7. IF NOT REPORT ERROR AND RESTART AT STEP 2

ERRORS:

1. ERROR 50 IS CALLED TO REPORT "XMIT DONE " TIMEOUTS
2. ERROR 56 IS CALLED TO REPORT "OVERRUN" ERROR

SYNC: M7277 SH3 INIT A H EF2

DEBUG:

1. I FAULT APPEARS ON ONLY ONE LINE SUSPECT UART MODULE FOR THE APPROPRIATE LINE IN QUESTION.
2. IF FAULT APPEARS ON ALL LINES SUSPECT THE M7279 MODULE

KEY LOGIC:

```
M7279 SH1 MASTER OR H E12-9  
SH2 MEMORY CHIP (3341) E13-11  
  
M7280 SH2 UC1 OR 2 MASTER OR EN2  
SH2-5 UART PIN 15 (BUF OR LINE NN)
```

%

```
MOV #2$, $LPERR ;SET UP ERROR LOOP RETURN  
1$: JSR PC, SELINE ;GO SELECT A LINE # TO TEST  
BR TST55 ;BR IF DONE ALL SELECTED LINES  
2$: MOV #BIT11, (R1) ;CLEAR OUT THE DH11  
MOVB LINE, (R1) ;SELECT THE LINE TO TEST
```

012737 017302 001110
004737 024544
000512
012711 004000
113711 030322

```

7345 017312 012761 037312 00G006      MOV    #TBUF,CAR(R1)    ;SET UP CURRENT ADDRESS
7346 017320 012761 177674 000010      MOV    #-68.,BCR(R1)   ;SET UP BYTE COUNT REG
7347 017326 012761 033503 000004      MOV    #33503,LPR(R1)  ;DO IT AT 9600 BAUD - 8 BITS
7348 017334 013761 027314 000012      MOV    LINMSK,BAR(R1)  ;ACTIVATE THE SELECTED LINE
7349
7350 017342 012737 000001 030350      MOV    #1,TIMEA        ;INIT TIMERS A AND B
7351 017350 005037 030352                CLR    TIMEB
7352 017354 005711                3$:   TST    (R1)          ;TRANSMITTER DONE ??
7353 017356 100425                BMI    4$              ;BR IF YES
7354 017360 004737 027016                JSR    PC,TIMEIT       ;CALL TIMER
7355 017364 000773                BR     3$              ;BR IF NO TIMEOUT
7356
7357 017366 004737 027200                JSR    PC,SAPS         ;GO SAVE PSW
7358 017372 011103                MOV    (R1),R3         ;GET THE WAS DATA
7359 017374 042703 077760                BIC    #77760,R3      ;CLEAR UNINTERESTING BITS
7360 017400 113704 030322                MOV    LINE,R4        ;SET UP S/B DATA
7361 017404 052704 100000                BIS    #BIT15,R4
7362 017410 010102                MOV    R1,R2          ;SET UP REGADR
7363 017412 004737 024416                JSR    PC,SUER2A      ;GO SET UP ERROR INFO
7364 017416 004537 024636                JSR    R5,SUNUM       ;PUT LINE NO. IN MESSAGE HEADER
7365 017422 030322                LINE
7366 017424 034254                EM50+53
7367 017426 104050                ERROR  50             ;REPORT XMIT DONE TIME OUT
7368 017430 000721                BR     1$             ;GO TRY NEXT LINE
7369
7370 017432 012737 000101 001204  4$:   MOV    #65.,$TMP1     ;SET UP TO READ 65. WORDS FROM SILO
7371 017440 113704 030322                MOV    LINE,R4        ;SET UP S/B DATA
7372 017444 000304                SWAB   R4
7373 017446 152704 000101                BIS    #65.,R4
7374 017452 052704 140000                BIS    #BIT15+BIT14,R4 ;PUT IN OVERRUN AND VALID DATA BITS
7375 017456 016103 000002                5$:   MOV    NRC(R1),R3   ;GET WAS DATA FROM SILO
7376 017462 005337 001204                DEC    $TMP1          ;COUNT ONE WORD READ
7377 017466 001373                BNE    5$             ;BR TIL 65. READ
7378 017470 020304                CMP    R3,R4          ;WAS DATA AND OVERRUN CORRECT ??
7379 017472 001700                BEQ    1$             ;BR IF YES TRY NEXT SELECTED LINE
7380
7381 017474 004737 027200                JSR    PC,SAPS         ;GO SAVE PSW
7382 017500 010102                MOV    R1,R2          ;SET UP REGADR
7383 017502 062702 000002                ADD    #NRC,R2
7384 017506 004737 024416                JSR    PC,SUER2A      ;GO SET UP ERROR INFO
7385 017512 004537 024636                JSR    R5,SUNUM       ;GO PUT LINE NO. IN MSG HDR
7386 017516 030322                LINE
7387 017520 034672                EM56+42
7388 017522 104056                ERROR  56             ;OVERRUN OR DATA INCORRECT
7389 017524 000663                BR     1$             ;GO TEST NEXT SELECTED LINE
7390

```

7391
7392
7393
7394 017526 000004
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417 017530 032777 002000 161402
7418 017536 001402
7419 017540 000137 020576
7420 017544 012700 030144
7421 017550 013737 030320 001202
7422 017556 006337 001202
7423 017562 063700 001202
7424 017566 011037 030310
7425 017572 013737 030310 030312
7426 017600 062737 000002 030312
7427 017606 005737 030310
7428 017612 001004
7429 017614 104401
7430 017616 036216
7431 017620 000137 020576
7432 017624 004737 024544
7433 017630 000473
7434
7435 017632 005077 010452
7436 017636 013737 030310 001202
7437 017644 042737 000340 177776
7438 017652 113701 030322
7439 017656 010137 001206
7440 017662 012777 002000 010420
7441 017670 012702 000020
7442 017674 010177 010410
7443 017700 012777 000001 010404
7444 017706 005077 010376
7445 017712 005005
7446 017714 017704 010372

```

:*****
:*TEST 55      ABBREVIATED MODEM CONTROL DIAGNOSTIC
:*****
TST55: SCOPE
.REM      %
TEST ABSTRACT:
*****

```

THE FOLLOWING 4 TESTS ARE EXTRACTED FROM DZDHK DIAGNOSTIC AND ARE INSERTED HERE SO THAT ALL LEVEL CONVERTERS AND CABLES CAN BE CHECKED WITH JUST ONE PROGRAM (RATHER THAN TWO) USING THE H315 TURNAROUND CONNECTOR.

THIS TEST VERIFIES THAT THE LINE ENABLE FUNCTION FLIP-FLOP CAN BE SET AND CLEARED FOR THE SELECTED LINE.

ERRORS

IF ANY ERRORS OCCUR IN THE FOLLOWING TESTS, THEN DZDHK, THE MODEM CONTROL DIAGNOSTIC SHOULD BE RUN IN ITS ENTIRETY FOR A MORE COMPLETE MODEM CONTROL CHECKOUT.

%

```

                BIT      #BIT10,@SWR      ;CHECK MODEM CONTROL?
                BEQ      2$                ;BRANCH IF YES.
                JMP      ENDA              ;OTHERWISE, GET OUT.
                2$:     MOV      #DMADRS,RO  ;RO POINTS TO BEGINNING OF DM ADDRESS TABLE.
                MOV      DHNUM,$TMP0
                ASL      $TMP0              ;DOUBLE TMP0
                ADD      $TMP0,RO          ;CREATE AN OFFSET.
                MOV      (RO),DHMCSR       ;MOVE THE DM ADDRESS INTO DHMCSR.
                MOV      DHMCSR,DHMLSR
                ADD      #2,DHMLSR         ;SAVE LINE STATUS REGISTER ADDRESS.
                TST      DHMCSR            ;IS THERE A MODEM CONTROL HERE?
                BNE      DOIT11            ;BRANCH IF YES.
                TYPE     MSG5
                JMP      ENDA              ;NO MODEM CONTROL FOUND.
                JSR      PC,SELINE         ;GET OUTTA HERE.
                BR       TST56             ;GO SELECT A LINE.
                ;:BR IF DONE ALL SELECTED LINES

MUX11:  CLR      @DHMCSR                  ;CLEAR CONTROL STATUS REGISTER
        MOV      DHMCSR,$TMP0            ;SAVE DEVICE REGISTER FOR ERROR MESSAGE
        BIC      #340,PS                  ;ENABLE INTERRUPTS
        MOVB    LINE,R1
        MOV      R1,$TMP2                 ;SAVE LINE NUMBER FOR ERRGR MESSAGE
MUX11A: MOV      #CLRMUX,@DHMCSR
        MOV      #16.,R2
        MOV      R1,@DHMCSR              ;SELECT LINE TO BE TESTED
        MOV      #LINENA,@DHMLSR         ;SET LINE ENABLE FUNCTION FLIP-FLOP
        CLR     @DHMCSR                   ;THE STEP BIT WILL BE USED TO FIND RIGHT LINE
MUX11B: CLR      R5
        MOV      @DHMLSR,R4              ;READ LINE STATUS REGISTER

```

```
7447 017720 117703 010364      MOVB   @DHMCSR,R3      ;READ CONTROL STATUS REGISTER
7448 017724 042703 117760      BIC    #177760,R3     ;CLEAR UNWANTED BITS
7449 017730 020103              CMP    R1,R3          ;IF LINE NUMBER=SELECTED LINE NUMBER,
7450 017732 001002              BNE   MUX11C          ;EXCEPT LINE ENABLE FUNCTION FLIP FLOP
7451 017734 012705 000001      MOV    #LINENA,R5
7452                                ;TO BE SET
7453 017740 020504              MUX11C: CMP   R5,R4    ;COMPARE EXPECTED AND RECEIVED
7454 017742 001403              BEQ   MUX11D          ;RESULTS
7455 017744 030504              BIT   R5,R4          ;TEST TO SEE IF THE LINE ENABLE BIT WAS SET
7456 017746 001001              BNE   MUX11D          ;IF SO, EVERYTHING IS OK
7457 017750 104060              ERROR  60            ;INDICATE A MODEM CONTROL ERROR
7458 017752 052777 000400 010330 MUX11D: BIS   #STEP,@DHMCSR ;EXAMINE NEXT LINE
7459 017760 005302              DEC   R2
7460 017762 001353              BNE   MUX11B
7461 017764 005005              CLR   R5
7462 017766 010177 010316      MUX11F: MOV   R1,@DHMCSR
7463 017772 010103              MOV   R1,R3          ;SET LINE COUNTER TO SELECTED LINE
7464 017774 005077 010312      CLR   @DHMLSR        ;CLEAR LINE ENABLE FLIP FLOP
7465 020000 105227 000000      INCB  #0             ;DELAY FOR CABLE
7466 020004 001375              BNE   -.4            ;DITTO
7467 020006 017704 010300      MOV   @DHMLSR,R4     ;READ LINE STATUS REGISTER
7468                                ;WAS LINE ENABLE FUNCTION FLIP FLOP
7469 020012 001704              BEQ   DOIT11         ;CLEARED
7470 020014 104060              ERROR  60            ;MODEM CONRTROL ERROR
7471 020016 000702              BR    DOIT11         ;DO THE NEXT LINE THAT IS SELECTED
7472
```


7473
7474
7475
7476 020020 000004
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491 020022 005077 010264
7492 020026 004737 024544
7493 020032 000471
7494 020034 005077 010250
7495 020040 042737 000340 177776
7496 020046 113701 030322
7497 020052 012777 002000 010230
7498 020060 010137 001206
7499 020064 012702 000020
7500 020070 010177 010214
7501 020074 012777 000003 010210
7502 020102 005077 010202
7503 020106 005005
7504 020110 017704 010176
7505 020114 117703 010170
7506 020120 042703 177760
7507 020124 020103
7508 020126 001002
7509 020130 012705 000143
7510
7511 020134 020405
7512 020136 001401
7513 020140 104060
7514 020142 052777 000400 010140
7515 020150 005302
7516 020152 001355
7517 020154 012705 000001
7518 020160 010103
7519 020162 010177 010122
7520 020166 042777 000002 010116
7521 020174 105227 000000
7522 020200 001375
7523 020202 017704 010104
7524 020206 020504
7525 020210 001704
7526 020212 104060
7527 020214 000702
7528

*TEST 56 MODEM CONTROL DIAGNOSTIC CONTINUED

TST56: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT CLEAR TO SEND AND CARRIER ARE SET
IF "LINE ENABLE" AND TERMINAL ARE SET FOR THE SELECTED LINE.

ERRORS:

IF ANY ERRORS OCCUR, RUN THE MODEM CONTROL DIAGNOSTIC,
DZDHK.

%
DOIT15: CLR @DHMLSR ;CLEAR LINE STATUS REGISTER
JSR PC,SELINE ;GO SELECT A LINE.
BR TST57 ;;BR IF DONE ALL SELECTED LINES
MUX15: CLR @DHMCSR ;CLEAR CONTROL REGISTER
BIC #340,PS ;ENABLE INTERRUPTS
MOVB LINE,R1
MOV #CLRMUX,@DHMCSR ;RESET ALL THE LINE STATUS REGISTERS
MOV R1,\$TMP2 ;SAVE LINE NUMBER FOR ERROR MESSAGE
MUX15A: MOV #16.,R2 ;16 LINES
MOV R1,@DHMCSR ;SELECT A LINE
MOV #LINENA+TRMRDY,@DHMLSR ;SET LINE ENABLE +TRMRDY
CLR @DHMCSR ;CLEAR CONTROL REGISTER
MUX15B: CLR R5 ;CLEAR EXPECTED RESULT
MOV @DHMLSR,R4 ;READ LINE STATUS
MOVB @DHMCSR,R3 ;READ LINE NUMBER
BIC #177760,R3 ;CLEAR UNWANTED BITS
CMP R1,R3 ;IF RECEIVED LINE=SELECTED LINE
BNE MUX15C ;EXPECT LINE ENABLE AND
MOV #LINENA+TRMRDY+CO+CS,R5 ;CLEAR TO SEND AND CARRIER ARE SET
MUX15C: CMP R4,R5 ;COMPARE EXPECTED AND
BEQ MUX15D ;RECEIVED RESULTS
ERROR 60 ;MODEM CONTROL ERROR
MUX15D: BIS #STEP,@DHMCSR ;UPDATE LINE COUNTER
DEC R2 ;CONTINUE IF ALL CHECKS
BNE MUX15B ;ARE NOT DONE FOR THIS LINE
MOV #LINENA,R5 ;EXPECT LINE ENABLE
MUX15E: MOV R1,R3 ;ON SELECTED LINE
MOV R1,@DHMCSR ;SELECT LINE
BIC #TRMRDY,@DHMLSR ;CLEAR TERMINAL
INCB #0 ;DELAY FOR CABLE
BNE .-4 ;DITTO
MOV @DHMLSR,R4 ;READ LINE STATUS REGISTER
CMP R5,R4 ;ONLY LINE ENABLE SHOULD BE
BEQ DOIT15 ;SET ON THIS LINE
ERROR 60 ;MODEM COMNTROL ERROR
BR DOIT15 ;GO DO THE NEXT LINE

7529
7530
7531
7532 020216 000004
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547 020220 005077 010066
7548 020224 004737 024544
7549 020230 000466
7550 020232 005077 010052
7551 020236 042737 000340 177776
7552 020244 113701 030322
7553 020250 010137 001206
7554 020254 012702 000020
7555 020260 010177 010024
7556 020264 012777 000005 010020
7557 020272 005077 010012
7558 020276 005005
7559 020300 017704 010006
7560 020304 117703 010000
7561 020310 042703 177760
7562 020314 020103
7563 020316 001002
7564 020320 012705 000205
7565
7566 020324 020405
7567 020326 001401
7568 020330 104060
7569 020332 052777 000400 007750
7570 020340 005302
7571 020342 001355
7572 020344 012705 000001
7573 020350 010103
7574 020352 010177 007732
7575 020356 042777 000004 007726
7576 020364 105227 000000
7577 020370 001375
7578 020372 017704 007714
7579 020376 020504
7580 020400 001707
7581 020402 104060
7582 020404 000705
7583

*TEST 57 MODEM CONTROL DIAGNOSTIC CONTINUED

TST57: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT RING IS SET IF "LINE ENABLE" AND
REQUEST TO SEND ARE SET FOR THE SELECTED LINE.

ERRORS:

IF ANY ERRORS OCCUR, RUN THE MODEM CONTROL DIAGNOSTIC,
DZDHK.

%
DOIT16: CLR @DHMLSR ;CLEAR LINE STATUS REGISTER
JSR PC,SELIN ;GO SELECT A LINE.
BR TST60 ;:BR IF DONE ALL SELECTED LINES
MUX16: CLR @DHMCSR ;CLEAR CONTROL REGISTER
BIC #340,PS ;ENABLE INTERRUPTS
MOVB LINE,R1
MOV R1,STMP2 ;SAVE LINE NBUMBER FOR ERROR MESSAGE
MUX16A: MOV #16.,R2 ;16 LINES
MOV R1,@DHMCSR ;SELECT A LINE
MOV #LINENA+RS,@DHMLSR ;SET LINE ENABLE +RS
CLR @DHMCSR ;CLEAR CONTROL REGISTER
MUX16B: CLR R5 ;CLEAR EXPECTED RESULT
MOV @DHMLSR,R4 ;READ LINE STATUS
MOVB @DHMCSR,R3 ;READ LINE NUMBER
BIC #177760,R3 ;CLEAR UNWANTED BITS
CMP R1,R3 ;IF RECEIVED LINE=SELECTED LINE
BNE MUX16C ;EXPECT LINE ENABLE AND
MOV #LINENA+RS+RING,R5 ;RING IS SET
MUX16C: CMP R4,R5 ;COMPARE EXPECTED AND
BEQ MUX16D ;RECEIVED RESULTS
ERROR 60 ;MODEM CONTROL ERROR
MUX16D: BIS #STEP,@DHMCSR ;UPDATE LINE COUNTER
DEC R2 ;CONTINUE IF ALL CHECKS
BNE MUX16B ;ARE NOT DONE FOR THIS LINE
MOV #LINENA,R5 ;EXPECT LINE ENABLE
MUX16E: MOV R1,R3 ;ON SELECTED LINE
MOV R1,@DHMCSR ;SELECT LINE
BIC #RS,@DHMLSR ;CLEAR REQUEST TO SEND
INCB #0 ;DELAY FOR CABLE
BNE -4 ;DITTO
MOV @DHMLSR,R4 ;READ LINE STATUS REGISTER
CMP R5,R4 ;ONLY LINE ENABLE SHOULD BE
BEQ DOIT16 ;SET ON THIS LINE
ERROR 60 ;MODEM CONTROL ERROR
BR DOIT16 ;GO DO THE NEXT LINE

7584
7585
7586
7587 020406 000004
7588
7589
7590
7591
7592
7593
7594
7595
7596
7597
7598
7599
7600
7601
7602 020410 005077 007676
7603 020414 004737 024544
7604 020420 000466
7605 020422 005077 007662
7606 020426 042737 000340 177776
7607 020434 113701 030322
7608 020440 010137 001206
7609 020444 012702 000020
7610 020450 010177 007634
7611 020454 012777 000011 007630
7612 020462 005077 007622
7613 020466 005005
7614 020470 017704 007616
7615 020474 117703 007610
7616 020500 042703 177760
7617 020504 020103
7618 020506 001002
7619 020510 012705 000031
7620
7621 020514 020405
7622 020516 001401
7623 020520 104060
7624 020522 052777 000400 007560
7625 020530 005302
7626 020532 001355
7627 020534 012705 000001
7628 020540 010103
7629 020542 010177 007542
7630 020546 042777 000010 007536
7631 020554 105227 000000
7632 020560 001375
7633 020562 017704 007524
7634 020566 020504
7635 020570 001707
7636 020572 104060
7637 020574 000705
7638
7639

*TEST 60 MODEM CONTROL DIAGNOSTIC CONTINUED

TST60: SCOPE
.REM %
TEST ABSTRACT:

THIS TEST VERIFIES THAT SECONDARY RECEIVE IS SET IF 'LINE
ENABLE' AND SECONDARY TRANSMIT ARE SET FOR THE SELECTED LINE.

ERRORS:

IF ANY ERRORS OCCUR, RUN THE MODEM CONTROL DIAGNOSTIC,
DZDHK.

%
DOIT17: CLR @DHMLSR ;CLEAR LINE STATUS REGISTER.
JSR PC,SELINE ;GO SELECT A LINE.
BR ENDA
MUX17: CLR @DHMCSR ;CLEAR CONTROL REGISTER
BIC #340,PS ;ENABLE INTERRUPTS
MOVB LINE,R1
MOV R1,\$TMP2 ;SAVE LINE NUMBER FOR ERROR MESSAGE
MUX17A: MOV #16.,R2 ;16 LINES
MOV R1,@DHMCSR ;SELECT A LINE
MOV #LINENA+SECTX,@DHMLSR ;SET LINE ENABLE +SECTX
CLR @DHMCSR ;CLEAR CONTROL REGISTER
MUX17B: CLR R5 ;CLEAR EXPECTED RESULT
MOV @DHMLSR,R4 ;READ LINE STATUS
MOVB @DHMCSR,R3 ;READ LINE NUMBER
BIC #177760,R3 ;CLEAR UNWANTED BITS
CMP R1,R3 ;IF RECEIVED LINE=SELECTED LINE
BNE MUX17C ;EXPECT LINE ENABLE AND
MOV #LINENA+SECTX+SECRX,R5 ;SECONDARY RECEIVE IS SET
MUX17C: CMP R4,R5 ;COMPARE EXPECTED AND
BEQ MUX17D ;RECEIVED RESULTS
ERROR 60 ;MODEM CONTROL ERROR
MUX17D: BIS #STEP,@DHMCSR ;UPDATE LINE COUNTER
DEC R2 ;CONTINUE IF ALL CHECKS
BNE MUX17B ;ARE NOT DONE FOR THIS LINE
MOV #LINENA,R5 ;EXPECT LINE ENABLE
MUX17E: MOV R1,R3 ;ON SELECTED LINE
MOV R1,@DHMCSR ;SELECT LINE
BIC #SECTX,@DHMLSR ;CLEAR SECONDARY TRANSMIT
INCB #0 ;DELAY FOR CABLE
BNE .-4 ;DITTO
MOV @DHMLSR,R4 ;READ LINE STATUS REGISTER
CMP R5,R4 ;ONLY LINE ENABLE SHOULD BE
BEQ DOIT17 ;SET ON THIS LINE
ERROR 60 ;MODEM CONTROL ERROR
BR DOIT17 ;GO DO THE NEXT LINE

```

7640 020576 000004          ENDA:  SCOPE
7641 020600 012737 000240 020662  MOV   #240,$EOP      ;NOP THE SCOPE AT THE BEGINNING OF EOP
7642 020606 005237 030320          INC   DHNUM          ;GENERATE NEW DH11 NUMBER
7643 020612 062737 000002 030326  ADD   #2,ADPTR      ;UPDATE THE TABLE POINTERS
7644 020620 062737 000002 030330  ADD   #2,VCPTR
7645 020626 062737 000002 030332  ADD   #2,BRPTR
7646 020634 006337 027306          ASL   SELMSK        ;SHIFT MARKER TO TEST NEXT DH11
7647 020640 001410          BEQ   $EOP          ;BR IF TESTED ALL SELECTED DH11'S
7648 020642 033737 027306 027310  BIT   SELMSK,DHSEL ;IS THIS DH11 SELECTED ?
7649 020650 001752          BEQ   ENDA         ;BR IF NOT
7650 020652 105037 001102          CLRB  $STNM        ;INIT TEST NUMBER
7651 020656 000137 002744          JMP   RSTRTA       ;GO TEST THIS DH11

```

.SBTTL END OF PASS ROUTINE

```

;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO START2

```

```

7660 020662          $EOP:  SCOPE
7661 020662 000004          CLR   $STNM        ;;ZERO THE TEST NUMBER
7662 020664 005037 001102          CLR   $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
7663 020670 005037 001222          INC   $PASS       ;;INCREMENT THE PASS NUMBER
7664 020674 005237 001240          BIC   #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
7665 020700 042737 100000 001240  DEC   (PC)+        ;;LOOP?
7666 020706 005327          $EOPCT: .WORD    1
7667 020710 000001          BGT   $DOAGN      ;;YES
7668 020712 003022          MOV   (PC)+,@(PC)+ ;;RESTORE COUNTER
7669 020714 012737          $ENDCT: .WORD    1
7670 020716 000001          $EOPCT
7671 020720 020710          TYPE  , $ENDMG    ;;TYPE "END PASS #"
7672 020722 104401 020767          MOV   $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
7673 020726 013746 001240          TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
7674 020732 104405          TYPE  , $ENULL   ;;TYPE A NULL CHARACTER
7675 020734 104401 020764          $GET42: MOV   @#42,R0 ;;GET MONITOR ADDRESS
7676 020740 013700 000042          BEQ   $DOAGN     ;;BRANCH IF NO MONITOR
7677 020744 001405          RESET ;;CLEAR THE WORLD
7678 020746 000005          $ENDAD: JSR   PC,(R0) ;;GO TO MONITOR
7679 020750 004710          NOP   ;;SAVE ROOM
7680 020752 000240          NOP   ;;FOR
7681 020754 000240          NOP   ;;ACT11
7682 020756 000240          $DOAGN:
7683 020760          JMP   @(PC)+      ;;RETURN
7684 020760 000137          $RTNAD: .WORD   START2
7685 020762 002640          $ENULL: .BYTE  -1,-1,0 ;;NULL CHARACTER STRING
7686 020764 377 377 000          $ENDMG: .ASCIZ <15><12>/END PASS #/
7687 020767 015 042412 042116
7688 020774 050040 051501 020123
7689 021002 000043

```

.SBTTL SCOPE HANDLER ROUTINE

```

;*****
;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>

```

7690
7691
7692
7693
7694
7695

```

7696      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7697      ;*SW14=1      LOOP ON TEST
7698      ;*SW11=1      INHIBIT ITERATIONS
7699      ;*SW09=1      LOOP ON ERROR
7700      ;*SW08=1      LOOP ON TEST IN SWR<7:0>
7701      ;*CALL
7702      ;*      SCOPE      ;;SCOPE=IOT
7703
7704      021004      $SCOPE:
7705      021004      104407      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
7706      021006      005037      030322      CLR      LINE      ;;INIT THE LINE NO. TO ZERO
7707      021012      013701      027302      MOV      DHADR,R1   ;;SET UP DEVADR IN R1
7708      021016      032777      040000      1$:      BIT      #BIT14,@SWR  ;;LOOP ON PRESENT TEST?
7709      021024      001114      BNE      $OVER      ;;YES IF SW14=1
7710      ;#####IART OF CODE FOR THE XOR TESTER#####
7711      021026      000416      $XTSTR: BR      6$      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
7712      ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7713      021030      013746      000004      MOV      @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7714      021034      012737      021054      000004      MOV      #5$,@#ERRVEC  ;;SET FOR TIMEOUT
7715      021042      005737      177060      TST      @#177060     ;;TIME OUT ON XOR?
7716      021046      012637      000004      MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7717      021052      000463      BR      $SVLAD      ;;GO TO THE NEXT TEST
7718      021054      022626      5$:      CMP      (SP)+,(SP)+  ;;CLEAR THE STACK AFTER A TIME OUT
7719      021056      012637      000004      MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7720      021062      000423      BR      7$      ;;LOOP ON THE PRESENT TEST
7721      021064      6$:;#####END OF CODE FOR THE XOR TESTER#####
7722      021064      032777      000400      160046      BIT      #BIT08,@SWR  ;;LOOP ON SPEC. TEST?
7723      021072      001404      BEQ      2$      ;;BR IF NO
7724      021074      127737      160040      001102      CMPB    @SWR,$TSTNM  ;;ON THE RIGHT TEST? SWR<7:0>
7725      021102      001465      BEQ      $OVER      ;;BR IF YES
7726      021104      105737      001103      2$:      TSTB    $ERFLG      ;;HAS AN ERROR OCCURRED?
7727      021110      001421      BEQ      3$      ;;BR IF NO
7728      021112      123737      001115      001103      CMPB    $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
7729      021120      101015      BHI      3$      ;;BR IF NO
7730      021122      032777      001000      160010      BIT      #BIT09,@SWR  ;;LOOP ON ERROR?
7731      021130      001404      BEQ      4$      ;;BR IF NO
7732      021132      013737      001110      001106      7$:      MOV      $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
7733      021140      000446      BR      $OVER
7734      021142      105037      001103      4$:      CLRB    $ERFLG      ;;ZERO THE ERROR FLAG
7735      021146      005037      001222      CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
7736      021152      000415      BR      1$      ;;ESCAPE TO THE NEXT TEST
7737      021154      032777      004000      157756      3$:      BIT      #BIT11,@SWR  ;;INHIBIT ITERATIONS?
7738      021162      001011      BNE      1$      ;;BR IF YES
7739      021164      005737      001240      TST      $PASS      ;;IF FIRST PASS OF PROGRAM
7740      021170      001406      BEQ      1$      ;;INHIBIT ITERATIONS
7741      021172      005237      001104      INC      $ICNT      ;;INCREMENT ITERATION COUNT
7742      021176      023737      001222      001104      CMP      $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
7743      021204      002024      BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
7744      021206      012737      000001      001104      1$:      MOV      #1,$ICNT     ;;REINITIALIZE THE ITERATION COUNTER
7745      021214      013737      021272      001222      MOV      $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
7746      021222      105237      001102      $SVLAD: INCB    $TSTNM  ;;COUNT TEST NUMBERS
7747      021226      113737      001102      001236      MOVB    $TSTNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
7748      021234      011637      001106      MOV      (SP),$LPADR  ;;SAVE SCOPE LOOP ADDRESS
7749      021240      011637      001110      MOV      (SP),$LPERR  ;;SAVE ERROR LOOP ADDRESS
7750      021244      005037      001224      CLR      $ESCAPE     ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
7751      021250      112737      000001      001115      MOVB    #1,$ERMAX    ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST

```

```

7752 021256 013777 001102 157656 $OVER:  MOV    $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
7753 021264 013716 001106          MOV    $LPADR,(SP)  ;;FUDGE RETURN ADDRESS
7754 021270 000002          RTI                ;;FIXES PS
7755 021272 000010          $MXCNT: 10          ;;MAX. NUMBER OF ITERATIONS
7756          .SBTTL  ERROR HANDLER ROUTINE
7757
7758          ;;*****
7759          ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7760          ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7761          ;;*AND GO TO $ERRTYP ON ERROR
7762          ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7763          ;;*SW15=1      HALT ON ERROR
7764          ;;*SW13=1      INHIBIT ERROR TYPEOUTS
7765          ;;*SW09=1      LOOP ON ERROR
7766          ;;*CALL
7767          ;;*      ERROR  N      ;;ERROR=EMT AND N-ERROR ITEM NUMBER
7768
7769          $ERROR:
7770 021274 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
7771 021276 105237 001103 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
7772 021302 001775          BEQ        7$          ;;DON'T LET THE FLAG GO TO ZERO
7773 021304 013777 001102 157630  MOV    $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
7774 021312 005237 001112          INC      $ERTTL      ;;INC THE ERROR COUNT
7775 021316 011637 001116          MOV    (SP),$ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
7776 021322 162737 000002 001116  SUB    #2,$ERRPC
7777 021330 117737 157562 001114  MOVB   @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
7778 021336 032777 020000 157574  BIT    #BIT13,@SWR    ;;SKIP TYPEOUT IF SET
7779 021344 001004          BNE      20$          ;;SKIP TYPEOUTS
7780 021346 004737 021460          JSR    PC,$ERRTYP    ;;GO TO USER ERROR ROUTINE
7781 021352 104401 001227          TYPE   , $CRLF
7782 021356          20$:
7783 021356 122737 000001 001252  CMPB   #APTENV,$ENV   ;;RUNNING IN APT MODE
7784 021364 001007          BNE      2$          ;;NO,SKIP APT ERROR REPORT
7785 021366 113737 001114 021400  MOVB   $ITEMB,21$    ;;SET ITEM NUMBER AS ERROR NUMBER
7786 021374 004737 022566          JSR    PC,$ATY4     ;;REPORT FATAL ERROR TO APT
7787 021400 000          21$:  .BYTE   0
7788 021401 000          .BYTE   0
7789 021402 000777          22$:  BR      22$          ;;APT ERROR LOOP
7790 021404 005777 157530  2$:      TST    @SWR          ;;HALT ON ERROR
7791 021410 100002          BPL      3$          ;;SKIP IF CONTINUE
7792 021412 000000          HALT                ;;HALT ON ERROR!
7793 021414 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
7794 021416 032777 001000 157514  3$:      BIT    #BIT09,@SWR  ;;LOOP ON ERROR SWITCH SET?
7795 021424 001402          BEQ      4$          ;;BR IF NO
7796 021426 013716 001110          MOV    $LPERR,(SP)  ;;FUDGE RETURN FOR LOOPING
7797 021432 005737 001224          4$:      TST    $ESCAPE    ;;CHECK FOR AN ESCAPE ADDRESS
7798 021436 001402          BEQ      5$          ;;BR IF NONE
7799 021440 013716 001224          MOV    $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
7800 021444          5$:
7801 021444 022737 020750 000042  CMP    #SENDAD,@#42  ;;ACT-11 AUTO-ACCEPT?
7802 021452 001001          BNE      6$          ;;BRANCH IF NO
7803 021454 000000          HALT                ;;YES
7804 021456          6$:
7805 021456 000002          RTI                ;;RETURN
7806          .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
7807

```

```

7808
7809
7810
7811
7812
7813 021460
7814 021460 104401 001227
7815 021464 010046
7816 021466 005000
7817 021470 153700 001114
7818 021474 001004
7819
7820 021476 013746 001116
7821
7822 021502 104402
7823 021504 000426
7824 021506 005300
7825 021510 006300
7826 021512 006300
7827 021514 006300
7828 021516 062700 001356
7829 021522 012037 021532
7830 021526 001404
7831 021530 104401
7832 021532 000000
7833 021534 104401 001227
7834 021540 012037 021550
7835 021544 001404
7836 021546 104401
7837 021550 000000
7838 021552 104401 001227
7839 021556 011000
7840 021560 001004
7841 021562 012600
7842 021564 104401 001227
7843 021570 000207
7844 021572
7845 021572 013046
7846 021574 104402
7847 021576 005710
7848 021600 001770
7849 021602 104401 021610
7850 021606 000771
7851 021610 020040 000
7852 021614

```

```

*****
*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

```

```

$ERRTYP:
      TYPE      , $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV       RO, -(SP)        ;; SAVE RO
      CLR       RO                ;; PICKUP THE ITEM INDEX
      BISB      @#$ITEMB, RO
      BNE      1$                ;; IF ITEM NUMBER IS ZERO, JUST
                                ;; TYPE THE PC OF THE ERROR
      MOV       $ERRPC, -(SP)    ;; SAVE $ERRPC FOR TYPEOUT
                                ;; ERROR ADDRESS
      TYPOC                    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      BR       6$                ;; GET OUT
1$:   DEC       RO                ;; ADJUST THE INDEX SO THAT IT WILL
      ASL      RO                ;; WORK FOR THE ERROR TABLE
      ASL      RO
      ASL      RO
      ADD      #$ERRTB, RO       ;; FORM TABLE POINTER
      MOV      (RO)+, 2$        ;; PICKUP "ERROR MESSAGE" POINTER
      BEQ      3$                ;; SKIP TYPEOUT IF NO POINTER
      TYPE     "ERROR MESSAGE" ;; TYPE THE "ERROR MESSAGE"
                                ;; "ERROR MESSAGE" POINTER GOES HERE
2$:   .WORD    0                 ;; "CARRIAGE RETURN" & "LINE FEED"
      TYPE     , $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
3$:   MOV      (RO)+, 4$        ;; PICKUP "DATA HEADER" POINTER
      BEQ      5$                ;; SKIP TYPEOUT IF 0
      TYPE     "DATA HEADER"   ;; TYPE THE "DATA HEADER"
                                ;; "DATA HEADER" POINTER GOES HERE
4$:   .WORD    0                 ;; "CARRIAGE RETURN" & "LINE FEED"
      TYPE     , $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
5$:   MOV      (RO), RO         ;; PICKUP "DATA TABLE" POINTER
      BNE      7$                ;; GO TYPE THE DATA
6$:   MOV      (SP)+, RO        ;; RESTORE RO
      TYPE     , $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
      RTS      PC                ;; RETURN
7$:   MOV      @(RO)+, -(SP)    ;; SAVE @(RO)+ FOR TYPEOUT
      TYPOC                    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      TST      (RO)              ;; IS THERE ANOTHER NUMBER?
      BEQ      6$                ;; BR IF NO
      TYPE     , 8$              ;; TYPE TWO(2) SPACES
      BR       7$                ;; LOOP
8$:   .ASCIZ   / /                ;; TWO(2) SPACES
      .EVEN
      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

```

```

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

```

```

7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863

```

```

7864      ;*                               ;;1=TYPE LEADING ZEROS
7865      ;*                               ;;0=SUPPRESS LEADING ZEROS
7866      ;*
7867      ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
7868      ;*$TYPOS OR $TYPOC
7869      ;*$CALL:
7870      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7871      ;*      TYPON                    ;;CALL FOR TYPEOUT
7872      ;*
7873      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
7874      ;*$CALL:
7875      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
7876      ;*      TYPOC                    ;;CALL FOR TYPEOUT
7877
7878 021614 017646 000000      $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
7879 021620 116637 000001 022037      MOV      1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
7880 021626 112637 022041      MOV      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
7881 021632 062716 000002      ADD      #2, (SP)           ;;ADJUST RETURN ADDRESS
7882 021636 000406      BR      $TYPON
7883 021640 112737 000001 022037      $TYPOC: MOV      #1, $OFILL      ;;SET THE ZERO FILL SWITCH
7884 021646 112737 000006 022041      MOV      #6, $OMODE+1      ;;SET FOR SIX(6) DIGITS
7885 021654 112737 000005 022036      $TYPON: MOV      #5, $OCNT      ;;SET THE ITERATION COUNT
7886 021662 010346      MOV      R3, -(SP)         ;;SAVE R3
7887 021664 010446      MOV      R4, -(SP)         ;;SAVE R4
7888 021666 010546      MOV      R5, -(SP)         ;;SAVE R5
7889 021670 113704 022041      MCVB    $OMODE+1, R4      ;;GET THE NUMBER OF DIGITS TO TYPE
7890 021674 005404      NEG      R4
7891 021676 062704 000006      ADD      #6, R4           ;;SUBTRACT IT FOR MAX. ALLOWED
7892 021702 110437 022040      MOV      R4, $OMODE      ;;SAVE IT FOR USE
7893 021706 113704 022037      MOV      $OFILL, R4      ;;GET THE ZERO FILL SWITCH
7894 021712 016605 000012      MOV      12(SP), R5      ;;PICKUP THE INPUT NUMBER
7895 021716 005003      CLR      R3              ;;CLEAR THE OUTPUT WORD
7896 021720 006105      1$:     ROL      R5        ;;ROTATE MSB INTO 'C'
7897 021722 000404      BR      3$              ;;GO DO MSB
7898 021724 006105      2$:     ROL      R5        ;;FORM THIS DIGIT
7899 021726 006105      ROL      R5
7900 021730 006105      ROL      R5
7901 021732 010503      MOV      R5, R3
7902 021734 006103      3$:     ROL      R3        ;;GET LSB OF THIS DIGIT
7903 021736 105337 022040      DECB    $OMODE          ;;TYPE THIS DIGIT?
7904 021742 100016      BPL     7$              ;;BR IF NO
7905 021744 042703 177770      BIC     #177770, R3     ;;GET RID OF JUNK
7906 021750 001002      BNE     4$              ;;TEST FOR 0
7907 021752 005704      TST     R4              ;;SUPPRESS THIS 0?
7908 021754 001403      BEQ     5$              ;;BR IF YES
7909 021756 005204      4$:     INC      R4        ;;DON'T SUPPRESS ANYMORE 0'S
7910 021760 052703 000060      BIS     #'0, R3        ;;MAKE THIS DIGIT ASCII
7911 021764 052703 000040      5$:     BIS     #' , R3     ;;MAKE ASCII IF NOT ALREADY
7912 021770 110337 022034      MOV      R3, 8$         ;;SAVE FOR TYPING
7913 021774 104401 022034      TYPE    , 8$           ;;GO TYPE THIS DIGIT
7914 022000 105337 022036      7$:     DECB    $OCNT      ;;COUNT BY 1
7915 022004 003347      BGT     2$              ;;BR IF MORE TO DO
7916 022006 002402      BLT     6$              ;;BR IF DONE
7917 022010 005204      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
7918 022012 000744      BR      2$              ;;GO DO THE LAST DIGIT
7919 022014 012605      6$:     MOV     (SP)+, R5   ;;RESTORE R5

```



```

7920 022016 012604          MOV      (SP)+,R4          ;;RESTORE R4
7921 022020 012603          MOV      (SP)+,R3          ;;RESTORE R3
7922 022022 016666 000002 000004  MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
7923 022030 012616          MOV      (SP)+,(SP)
7924 022032 000002          RTI                          ;;RETURN
7925 022034 000          8$:      .BYTE 0          ;;STORAGE FOR ASCII DIGIT
7926 022035 000          .BYTE 0          ;;TERMINATOR FOR TYPE ROUTINE
7927 022036 000          $OCNT: .BYTE 0        ;;OCTAL DIGIT COUNTER
7928 022037 000          $OFILL: .BYTE 0       ;;ZERO FILL SWITCH
7929 022040 000000          $OMODE: .WORD 0        ;;NUMBER OF DIGITS TO TYPE
7930          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7931
7932          ;*****
7933          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7934          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7935          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7936          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7937          ;*REPLACED WITH SPACES.
7938          ;*CALL:
7939          ;*      MOV      NUM,-(SP)          ;;PUT THE BINARY NUMBER ON THE STACK
7940          ;*      TYPDS          ;;GO TO THE ROUTINE
7941
7942          $TYPDS:
7943 022042 010046          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
7944 022044 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
7945 022046 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
7946 022050 010346          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
7947 022052 010546          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
7948 022054 012746 020200          MOV      #20200,-(SP)     ;;SET BLANK SWITCH AND SIGN
7949 022060 016605 000020          MOV      20(SP),R5        ;;GET THE INPUT NUMBER
7950 022064 100004          BPL      1$                ;;BR IF INPUT IS POS.
7951 022066 005405          NEG      R5                ;;MAKE THE BINARY NUMBER POS.
7952 022070 112766 000055 000001          MOVB     #'-,1(SP)        ;;MAKE THE ASCII NUMBER NEG.
7953 022076 005000          1$:      CLR      R0                ;;ZERO THE CONSTANTS INDEX
7954 022100 012703 022256          MOV      #$DBLK,R3        ;;SETUP THE OUTPUT POINTER
7955 022104 112723 000040          MOVB     #'',(R3)+        ;;SET THE FIRST CHARACTER TO A BLANK
7956 022110 005002          2$:      CLR      R2                ;;CLEAR THE BCD NUMBER
7957 022112 016001 022246          MOV      $DTBL(R0),R1     ;;GET THE CONSTANT
7958 022116 160105          3$:      SUB      R1,R5          ;;FORM THIS BCD DIGIT
7959 022120 002402          BLT     4$                ;;BR IF DONE
7960 022122 005202          INC     R2                ;;INCREASE THE BCD DIGIT BY 1
7961 022124 000774          BR      3$
7962 022126 060105          4$:      ADD     R1,R5          ;;ADD BACK THE CONSTANT
7963 022130 005702          TST     R2                ;;CHECK IF BCD DIGIT=0
7964 022132 001002          BNE     5$                ;;FALL THROUGH IF 0
7965 022134 105716          TSTB    (SP)              ;;STILL DOING LEADING 0'S?
7966 022136 100407          BMI     7$                ;;BR IF YES
7967 022140 106316          5$:      ASLB    (SP)          ;;MSD?
7968 022142 103003          BCC     6$                ;;BR IF NO
7969 022144 116663 000001 177777          MOVB     1(SP),-1(R3)     ;;YES--SET THE SIGN
7970 022152 052702 000060          6$:      BIS     #'0,R2        ;;MAKE THE BCD DIGIT ASCII
7971 022156 052702 000040          7$:      BIS     #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7972 022162 110223          MOVB     R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7973 022164 005720          TST     (R0)+            ;;JUST INCREMENTING
7974 022166 020027 000010          CMP     R0,#10          ;;CHECK THE TABLE INDEX
7975 022172 002746          BLT     2$                ;;GO DO THE NEXT DIGIT

```

```

7976 022174 003002          BGT      8$          ;;GO TO EXIT
7977 022176 010502          MOV      R5,R2       ;;GET THE LSD
7978 022200 000764          BR       6$          ;;GO CHANGE TO ASCII
7979 022202 105726          8$: TSTB   (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
7980 022204 100003          BPL     9$          ;;BR IF NO
7981 022206 116663 177777 177776 MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
7982 022214 105013          9$: CLRB   (R3)       ;;SET THE TERMINATOR
7983 022216 012605          MOV     (SP)+,R5     ;;POP STACK INTO R5
7984 022220 012603          MOV     (SP)+,R3     ;;POP STACK INTO R3
7985 022222 012602          MOV     (SP)+,R2     ;;POP STACK INTO R2
7986 022224 012601          MOV     (SP)+,R1     ;;POP STACK INTO R1
7987 022226 012600          MOV     (SP)+,R0     ;;POP STACK INTO R0
7988 022230 104401 022256   TYPE     $DBLK        ;;NOW TYPE THE NUMBER
7989 022234 016666 000002 000004 MOV     2(SP),4(SP)   ;;ADJUST THE STACK
7990 022242 012616          MOV     (SP)+,(SP)
7991 022244 000002          RTI                      ;;RETURN TO USER
7992 022246 023420          $DTBL: 10000.
7993 022250 001750          1000.
7994 022252 000144          100.
7995 022254 000012          10.
7996 022256 000004          $DBLK: .BLKW 4
7997                          .SBTTL TYPE ROUTINE
7998
7999                          ;*****
8000                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8001                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8002                          ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
8003                          ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
8004                          ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
8005                          ;*
8006                          ;*CALL:
8007                          ;*1) USING A TRAP INSTRUCTION
8008                          ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
8009                          ;*OR
8010                          ;*      TYPE
8011                          ;*      MESADR
8012                          ;*
8013
8014 022266 105737 001157   $TYPE: TSTB   $TPFLG          ;;IS THERE A TERMINAL?
8015 022272 100002          BPL     1$          ;;BR IF YES
8016 022274 000000          HALT                    ;;HALT HERE IF NO TERMINAL
8017 022276 000430          BR      3$          ;;LEAVE
8018 022300 010046          1$: MOV     R0,-(SP)     ;;SAVE R0
8019 022302 017600 000002   MOV     @2(SP),R0     ;;GET ADDRESS OF ASCIZ STRING
8020 022306 122737 000001 001252   CMPB   #APTENV,$ENV   ;;RUNNING IN APT MODE
8021 022314 001011          BNE     62$         ;;NO,GO CHECK FOR APT CONSOLE
8022 022316 132737 000100 001253   BITB   #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
8023 022324 001405          BEQ     62$         ;;NO,GO CHECK FOR CONSOLE
8024 022326 010037 022336   MOV     R0,61$        ;;SETUP MESSAGE ADDRESS FOR APT
8025 022332 004737 022556   JSR     PC,$ATY3      ;;SPOOL MESSAGE TO APT
8026 022336 000000          61$: .WORD 0          ;;MESSAGE ADDRESS
8027 022340 132737 000040 001253   62$: BITB   #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
8028 022346 001003          BNE     60$         ;;YES,SKIP TYPE OUT
8029 022350 112046          2$: MOVB   (R0)+,-(SP)  ;;PUSH CHARACTER TO BE TYPED ONTO STACK
8030 022352 001005          BNE     4$          ;;BR IF IT ISN'T THE TERMINATOR
8031 022354 005726          TST    (SP)+         ;;IF TERMINATOR POP IT OFF THE STACK

```

```

8032 022356 012600      60$:  MOV    (SP)+,R0      ;;RESTORE R0
8033 022360 062716 000002 3$:  ADD    #2,(SP)      ;;ADJUST RETURN PC
8034 022364 000002      RTI                    ;;RETURN
8035 022366 122716 000011 4$:  CMPB   #HT,(SP)      ;;BRANCH IF <HT>
8036 022372 001430      BEQ    8$              ;;
8037 022374 122716 000200  CMPB   #CRLF,(SP)    ;;BRANCH IF NOT <CRLF>
8038 022400 001006      BNE    5$              ;;
8039 022402 005726      TST    (SP)+          ;;POP <CR><LF> EQUIV
8040 022404 104401      TYPE                    ;;TYPE A CR AND LF
8041 022406 001227      $CRLF
8042 022410 105037 022544  CLRB   $CHARCNT      ;;CLEAR CHARACTER COUNT
8043 022414 000755      BR     2$              ;;GET NEXT CHARACTER
8044 022416 004737 022500 5$:  JSR    PC,$TYPEC     ;;GO TYPE THIS CHARACTER
8045 022422 123726 001156 6$:  CMPB   $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
8046 022426 001350      BNE    2$              ;;IF NO GO GET NEXT CHAR.
8047 022430 013746 001154  MOV    $NULL,-(SP)   ;;GET # OF FILLER CHARS. NEEDED
8048                                     ;;AND THE NULL CHAR.
8049 022434 105366 000001 7$:  DECB   1(SP)         ;;DOES A NULL NEED TO BE TYPED?
8050 022440 002770      BLT    6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
8051 022442 004737 022500  JSR    PC,$TYPEC     ;;GO TYPE A NULL
8052 022446 105337 022544  DECB   $CHARCNT      ;;DO NOT COUNT AS A COUNT
8053 022452 000770      BR     7$              ;;LOOP
8054
8055                                     ;HORIZONTAL TAB PROCESSOR
8056
8057 022454 112716 000040 8$:  MOVB   #' ,(SP)     ;;REPLACE TAB WITH SPACE
8058 022460 004737 022500 9$:  JSR    PC,$TYPEC     ;;TYPE A SPACE
8059 022464 132737 000007 022544 BITB   #7,$CHARCNT   ;;BRANCH IF NOT AT
8060 022472 001372      BNE    9$              ;;TAB STOP
8061 022474 005726      TST    (SP)+          ;;POP SPACE OFF STACK
8062 022476 000724      BR     2$              ;;GET NEXT CHARACTER
8063 022500 105777 156444 $TYPEC: TSTB   @STPS   ;;WAIT UNTIL PRINTER IS READY
8064 022504 100375      BPL    $TYPEC
8065 022506 116677 000002 156436 MOVB   2(SP),@STPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8066 022514 122766 000015 000002 CMPB   #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
8067 022522 001003      BNE    1$              ;;BRANCH IF NO
8068 022524 105037 022544  CLRB   $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
8069 022530 000406      BR     $TYPEX         ;;EXIT
8070 022532 122766 000012 000002 1$:  CMPB   #LF,2(SP)    ;;IS CHARACTER A LINE FEED?
8071 022540 001402      BEQ    $TYPEX         ;;BRANCH IF YES
8072 022542 105227      INCB   (PC)+         ;;COUNT THE CHARACTER
8073 022544 000000      $CHARCNT: .WORD    0 ;;CHARACTER COUNT STORAGE
8074 022546 000207      $TYPEX: RTS         PC
8075
8076                                     .SBTTL  APT COMMUNICATIONS ROUTINE
8077
8078                                     ;*****
8079 022550 112737 000001 023014 $ATY1: MOVB   #1,$FFLG  ;;TO REPORT FATAL ERROR
8080 022556 112737 000001 023012 $ATY3: MOVB   #1,$MFLG  ;;TO TYPE A MESSAGE
8081 022564 000403      BR     $ATYC
8082 022566 112737 000001 023014 $ATY4: MOVB   #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
8083 022574      $ATYC:
8084 022574 010046      MOV    R0,-(SP)     ;;PUSH R0 ON STACK
8085 022576 010146      MOV    R1,-(SP)     ;;PUSH R1 ON STACK
8086 022600 105737 023012  TSTB   $MFLG        ;;SHOULD TYPE A MESSAGE?
8087 022604 001450      BEQ    5$            ;;IF NOT: BR

```

```

8088 022606 122737 000001 001252      CMPB   #APTENV,$ENV      ;;OPERATING UNDER APT?
8089 022614 001031                    BNE    3$                ;;IF NOT: BR
8090 022616 132737 000100 001253      BITB   #APTSPool,$ENVM   ;;SHOULD SPOOL MESSAGES?
8091 022624 001425                    BEQ    3$                ;;IF NOT: BR
8092 022626 017600 000004                    MOV    @4(SP),RO        ;;GET MESSAGE ADDR.
8093 022632 062766 000002 000004      ADD    #2,4(SP)         ;;BUMP RETURN ADDR.
8094 022640 005737 001232      1$:   TST    $MSGTYPE     ;;SEE IF DONE W/ LAST XMISSION?
8095 022644 001375                    BNE    1$                ;;IF NOT: WAIT
8096 022646 010037 001246      MOV    RO,$MSGAD       ;;PUT ADDR IN MAILBOX
8097 022652 105720      2$:   TSTB   (RO)+        ;;FIND END OF MESSAGE
8098 022654 001376                    BNE    2$
8099 022656 163700 001246      SUB    $MSGAD,RO        ;;SUB START OF MESSAGE
8100 022662 006200                    ASR    RO                ;;GET MESSAGE LNTH IN WORDS
8101 022664 010037 001250      MOV    RO,$MSGGLT      ;;PUT LENGTH IN MAILBOX
8102 022670 012737 000004 001232      MOV    #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
8103 022676 000413                    BR     5$
8104 022700 017637 000004 022724  3$:   MOV    @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
8105 022706 062766 000002 000004      ADD    #2,4(SP)         ;;BUMP RETURN ADDRESS
8106 022714 013746 177776                    MOV    177776,-(SP)    ;;PUSH 177776 ON STACK
8107 022720 004737 022266      JSR    PC,$TYPE        ;;CALL TYPE MACRO
8108 022724 000000      4$:   .WORD    0
8109 022726      5$:
8110 022726 105737 023014      10$:  TSTB   $FFLG         ;;SHOULD REPORT FATAL ERROR?
8111 022732 001416                    BEQ    12$               ;;IF NOT: BR
8112 022734 005737 001252      TST    $ENV            ;;RUNNING UNDER APT?
8113 022740 001413                    BEQ    12$               ;;IF NOT: BR
8114 022742 005737 001232      11$:  TST    $MSGTYPE     ;;FINISHED LAST MESSAGE?
8115 022746 001375                    BNE    11$              ;;IF NOT: WAIT
8116 022750 017637 000004 001234      MOV    @4(SP),$FATAL   ;;GET ERROR #
8117 022756 062766 000002 000004      ADD    #2,4(SP)         ;;BUMP RETURN ADDR.
8118 022764 005237 001232      INC    $MSGTYPE        ;;TELL APT TO TAKE ERROR
8119 022770 105037 023014      12$:  CLRB   $FFLG         ;;CLEAR FATAL FLAG
8120 022774 105037 023013      CLRB   $LFLG          ;;CLEAR LOG FLAG
8121 023000 105037 023012      CLRB   $MFLG         ;;CLEAR MESSAGE FLAG
8122 023004 012601      MOV    (SP)+,R1        ;;POP STACK INTO R1
8123 023006 012600      MOV    (SP)+,RO        ;;POP STACK INTO RO
8124 023010 000207      RTS    PC              ;;RETURN
8125 023012      000      $MFLG: .BYTE    0      ;;MESSG. FLAG
8126 023013      000      $LFLG: .BYTE    0      ;;LOG FLAG
8127 023014      000      $FFLG: .BYTE    0      ;;FATAL FLAG
8128      023016      .EVEN
8129      000200      APTSIZE=200
8130      000001      APTENV=001
8131      000100      APTSPool=100
8132      000040      APTCSUP=040
8133      .SBTTL  TTY INPUT ROUTINE
8134
8135      ;;*****
8136      .ENABL  LSB
8137
8138      ;;*****
8139      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
8140      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
8141      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
8142      ;*WHEN OPERATING IN TTY FLAG MODE.
8143 023016 022737 000176 001140  $CKSWR: CMP    #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?

```

8144	023024	001074		BNE	15\$::BRANCH IF NO
8145	023026	105777	156112	TSTB	@\$TKS	::CHAR THERE?
8146	023032	100071		BPL	15\$::IF NO, DON'T WAIT AROUND
8147	023034	117746	156106	MOVB	@\$TKB,-(SP)	::SAVE THE CHAR
8148	023040	042716	177600	BIC	# C177,(SP)	::STRIP-OFF THE ASCII
8149	023044	022726	000007	CMP	#7,(SP)+	::IS IT A CONTROL G?
8150	023050	001062		BNE	15\$::NO, RETURN TO USER
8151	023052	123727	001134 000001	CMPB	\$AUTOB,#1	::ARE WE RUNNING IN AUTO-MODE?
8152	023060	001456		BEQ	15\$::BRANCH IF YES
8153						
8154	023062	104401	023671	TYPE	,\$CNTLG	::ECHO THE CONTROL-G (G)
8155	023066	104401	023676	\$GTSWR: TYPE	,\$MSWR	::TYPE CURRENT CONTENTS
8156	023072	013746	000176	MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
8157	023076	104402		TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
8158	023100	104401	023707	TYPE	,\$MNEW	::PROMPT FOR NEW SWR
8159	023104	005046		19\$: CLR	-(SP)	::CLEAR COUNTER
8160	023106	005046		CLR	-(SP)	::THE NEW SWR
8161	023110	105777	156030	7\$: TSTB	@\$TKS	::CHAR THERE?
8162	023114	100375		BPL	7\$::IF NOT TRY AGAIN
8163						
8164	023116	117746	156024	MOVB	@\$TKB,-(SP)	::PICK UP CHAR
8165	023122	042716	177600	BIC	# C177,(SP)	::MAKE IT 7-BIT ASCII
8166						
8167						
8168						
8169	023126	021627	000025	9\$: CMP	(SP),#25	::IS IT A CONTROL-U?
8170	023132	001005		BNE	10\$::BRANCH IF NOT
8171	023134	104401	023664	TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (U)
8172	023140	062706	000006	20\$: ADD	#6,SP	::IGNORE PREVIOUS INPUT
8173	023144	000757		BR	19\$::LET'S TRY IT AGAIN
8174						
8175						
8176	023146	021627	000015	10\$: CMP	(SP),#15	::IS IT A <CR>?
8177	023152	001022		BNE	16\$::BRANCH IF NO
8178	023154	005766	000004	TST	4(SP)	::YES, IS IT THE FIRST CHAR?
8179	023160	001403		BEQ	11\$::BRANCH IF YES
8180	023162	016677	000002 155750	MOV	2(SP),@SWR	::SAVE NEW SWR
8181	023170	062706	000006	11\$: ADD	#6,SP	::CLEAR UP STACK
8182	023174	104401	001227	14\$: TYPE	,\$CRLF	::ECHO <CR> AND <LF>
8183	023200	123727	001135 000001	CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
8184	023206	001003		BNE	15\$::BRANCH IF NOT
8185	023210	012777	000100 155726	MOV	#100,@\$TKS	::RE-ENABLE TTY KBD INTERRUPTS
8186	023216	000002		15\$: RTI		::RETURN
8187	023220	004737	022500	16\$: JSR	PC,\$TYPEC	::ECHO CHAR
8188	023224	021627	000060	CMP	(SP),#60	::CHAR < 0?
8189	023230	002420		BLT	18\$::BRANCH IF YES
8190	023232	021627	000067	CMP	(SP),#67	::CHAR > 7?
8191	023236	003015		BGT	18\$::BRANCH IF YES
8192	023240	042726	000060	BIC	#60,(SP)+	::STRIP-OFF ASCII
8193	023244	005766	000002	TST	2(SP)	::IS THIS THE FIRST CHAR
8194	023250	001403		BEQ	17\$::BRANCH IF YES
8195	023252	006316		ASL	(SP)	::NO, SHIFT PRESENT
8196	023254	006316		ASL	(SP)	:: CHAR OVER TO MAKE
8197	023256	006316		ASL	(SP)	:: ROOM FOR NEW ONE.
8198	023260	005266	000002	17\$: INC	2(SP)	::KEEP COUNT OF CHAR
8199	023264	056616	177776	BIS	-2(SP),(SP)	::SET IN NEW CHAR

```

8200 023270 000707          BR      7$          ;;GET THE NEXT ONE
8201 023272 104401 001226 18$:  TYPE  ,9QUES      ;;TYPE ?<CR><LF>
8202 023276 000720          BR      20$         ;;SIMULATE CONTROL-U
8203          .DSABL  LSB
8204
8205
8206          ;*****
8207          ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8208          ;*CALL:
8209          ;*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
8210          ;*      RETURN HERE   ;;CHARACTER IS ON THE STACK
8211          ;*
8212          ;
8213
8214 023300 011646          $RDCHR: MOV    (SP),-(SP)      ;;PUSH DOWN THE PC
8215 023302 016666 000004 000002 MOV    4(SP),2(SP)      ;;SAVE THE PS
8216 023310 105777 155630 1$:  TSTB   @TKS          ;;WAIT FOR
8217 023314 100375          BPL     1$          ;;A CHARACTER
8218 023316 117766 155624 000004 MOVB   @TKB,4(SP)      ;;READ THE TTY
8219 023324 042766 177600 000004 BIC    # C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
8220 023332 026627 000004 000023 CMP    4(SP),#23      ;;IS IT A CONTROL-S?
8221 023340 001013          BNE    3$          ;;BRANCH IF NO
8222 023342 105777 155576 2$:  TSTB   @TKS          ;;WAIT FOR A CHARACTER
8223 023346 100375          BPL     2$          ;;LOOP UNTIL ITS THERE
8224 023350 117746 155572 MOVB   @TKB,-(SP)      ;;GET CHARACTER
8225 023354 042716 177600 BIC    # C177,(SP)    ;;MAKE IT 7-BIT ASCII
8226 023360 022627 000021 CMP    (SP)+,#21      ;;IS IT A CONTROL-Q?
8227 023364 001366          BNE    2$          ;;IF NOT DISCARD IT
8228 023366 000750          BR     1$          ;;YES, RESUME
8229 023370 026627 000004 000140 3$:  CMP    4(SP),#140     ;;IS IT UPPER CASE?
8230 023376 002407          SLT    4$          ;;BRANCH IF YES
8231 023400 026627 000004 000175 CMP    4(SP),#175     ;;IS IT A SPECIAL CHAR?
8232 023406 003003          BGT    4$          ;;BRANCH IF YES
8233 023410 042766 000040 000004 BIC    #40,4(SP)     ;;MAKE IT UPPER CASE
8234 023416 000002          RTI          ;;GO BACK TO USER
8235          ;*****
8236          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
8237          ;*CALL:
8238          ;*      RDLIN          ;;INPUT A STRING FROM THE TTY
8239          ;*      RETURN HERE   ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8240          ;*
8241          ;*      TERMINATOR WILL BE A BYTE OF ALL 0'S
8242 023420 010346          $RDLIN: MOV    R3,-(SP)      ;;SAVE R3
8243 023422 005046          CLR    -(SP)         ;;CLEAR THE RUBOUT KEY
8244 023424 012703 023654 1$:  MOV    #TTYIN,R3      ;;GET ADDRESS
8245 023430 022703 023664 2$:  CMP    #TTYIN+8.,R3   ;;BUFFER FULL?
8246 023434 101456          BLOS   4$          ;;BR IF YES
8247 023436 104410          RDCHR          ;;GO READ ONE CHARACTER FROM THE TTY
8248 023440 112613          MOVB   (SP)+,(R3)    ;;GET CHARACTER
8249 023442 122713 000177 10$:  CMPB   #177,(R3)     ;;IS IT A RUBOUT
8250 023446 001022          BNE    5$          ;;BR IF NO
8251 023450 005716          TST    (SP)         ;;IS THIS THE FIRST RUBOUT?
8252 023452 001007          BNE    6$          ;;BR IF NO
8253 023454 112737 000134 023652 MOVB   #' ,9$        ;;TYPE A BACK SLASH
8254 023462 104401 023652          TYPE  ,9$
8255 023466 012716 177777          MOV    #-1,(SP)     ;;SET THE RUBOUT KEY

```

```

8256 023472 005303          6$: DEC R3          ;;BACKUP BY ONE
8257 023474 020327 023654  CMP R3,#$TTYIN  ;;STACK EMPTY?
8258 023500 103434          BLO 4$          ;;BR IF YES
8259 023502 111337 023652  MOVB (R3),9$    ;;SETUP TO TYPEOUT THE DELETED CHAR.
8260 023506 104401 023652  TYPE ,9$        ;;GO TYPE
8261 023512 000746          BR 2$           ;;GO READ ANOTHER CHAR.
8262 023514 005716          5$: TST (SP)      ;;RUBOUT KEY SET?
8263 023516 001406          BEQ 7$          ;;BR IF NO
8264 023520 112737 000134 023652  MOVB #' ,9$     ;;TYPE A BACK SLASH
8265 023526 104401 023652  TYPE ,9$        ;;
8266 023532 005016          CLR (SP)        ;;CLEAR THE RUBOUT KEY
8267 023534 122713 000025  7$: CMPB #25,(R3)  ;;IS CHARACTER A CTRL U?
8268 023540 001003          BNE 8$          ;;BR IF NO
8269 023542 104401 023664  TYPE ,SCNTLU    ;;TYPE A CONTROL 'U'
8270 023546 000726          BR 1$           ;;GO START OVER
8271 023550 122713 000022  8$: CMPB #22,(R3)  ;;IS CHARACTER A " R"?
8272 023554 001011          BNE 3$          ;;BRANCH IF NO
8273 023556 105013          CLRB (R3)       ;;CLEAR THE CHARACTER
8274 023560 104401 001227  TYPE ,SCRLF     ;;TYPE A "CR" & "LF"
8275 023564 104401 023654  TYPE ,STTYIN    ;;TYPE THE INPUT STRING
8276 023570 000717          BR 2$           ;;GO PICKUP ANOTHER CHACTER
8277 023572 104401 001226  4$: TYPE ,SQUES  ;;TYPE A '?'
8278 023576 000712          BR 1$           ;;CLEAR THE BUFFER AND LOOP
8279 023600 111337 023652  3$: MOVB (R3),9$    ;;ECHO THE CHARACTER
8280 023604 104401 023652  TYPE ,9$        ;;
8281 023610 122723 000015  CMPB #15,(R3)+  ;;CHECK FOR RETURN
8282 023614 001305          BNE 2$          ;;LOOP IF NOT RETURN
8283 023616 105063 177777  CLRB -1(R3)     ;;CLEAR RETURN (THE 15)
8284 023622 104401 001230  TYPE ,SLF       ;;TYPE A LINE FEED
8285 023626 005726          TST (SP)+       ;;CLEAN RUBOUT KEY FROM THE STACK
8286 023630 012603          MOV (SP)+,R3    ;;RESTORE R3
8287 023632 011646          MOV (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
8288 023634 016666 000004 000002  MOV 4(SP),2(SP)  ;; FIRST ASCII CHARACTER ON IT
8289 023642 012766 023654 000004  MOV #STTYIN,4(SP)
8290 023650 000002          RTI           ;;RETURN
8291 023652 000          9$: .BYTE 0    ;;STORAGE FOR ASCII CHAR. TO TYPE
8292 023653 000          .BYTE 0    ;;TERMINATOR
8293 023654 000010          $TTYIN: .BLKB 8.  ;;RESERVE 8 BYTES FOR TTY INPUT
8294 023664 052536 005015 000  $CNTLU: .ASCIZ / U/<15><12>  ;;CONTROL 'U'
8295 023671 136 006507 000012  $CNTLG: .ASCIZ / G/<15><12>  ;;CONTROL 'G'
8296 023676 005015 053523 020122  $MSWR: .ASCIZ <15><12>/SWR = /
8297 023704 020075 000
8298 023707 040 047040 053505  $MNEW: .ASCIZ / NEW = /
8299 023714 036440 000040
8300 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
8301
8302 ;;*****
8303 ;;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
8304 ;;*CHANGE IT TO BINARY.
8305 ;;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
8306 ;;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
8307 ;;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
8308 ;;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
8309 ;;*CALL:
8310 ;;* RDOCT ;;READ AN OCTAL NUMBER
8311 ;;* RETURN HERE ;;LOW ORDER BITS ARE ON TOP OF THE STACK

```

```

8312          ;*                               ;;HIGH ORDER BITS ARE IN $HIOCT
8313
8314 023720 011646 $RDOCT: MOV (SP),-(SP)      ;;PROVIDE SPACE FOR THE
8315 023722 016666 000004 000002 MOV 4(SP),2(SP)  ;;INPUT NUMBER
8316 023730 010046 MOV R0,-(SP)      ;;PUSH R0 ON STACK
8317 023732 010146 MOV R1,-(SP)      ;;PUSH R1 ON STACK
8318 023734 010246 MOV R2,-(SP)      ;;PUSH R2 ON STACK
8319 023736 104411 1$: RDLIN                ;;READ AN ASCIZ LINE
8320 023740 012600 MOV (SP)+,R0      ;;GET ADDRESS OF 1ST CHARACTER
8321 023742 010037 024046 MOV R0,5$      ;;AND SAVE IT
8322 023746 005001 CLR R1                ;;CLEAR DATA WORD
8323 023750 005002 CLR R2
8324 023752 112046 2$: MOV (R0)+,-(SP)      ;;PICKUP THIS CHARACTER
8325 023754 001420 BEQ 3$                ;;IF ZERO GET OUT
8326 023756 122716 000060 CMPB #'0,(SP)      ;;MAKE SURE THIS CHARACTER
8327 023762 003026 BGT 4$                ;;IS AN OCTAL DIGIT
8328 023764 122716 000067 CMPB #'7,(SP)
8329 023770 002423 BLT 4$
8330 023772 006301 ASL R1                ;;*2
8331 023774 006102 ROL R2
8332 023776 006301 ASL R1                ;;*4
8333 024000 006102 ROL R2
8334 024002 006301 ASL R1                ;;*8
8335 024004 006102 ROL R2
8336 024006 042716 177770 BIC #'C7,(SP)      ;;STRIP THE ASCII JUNK
8337 024012 062601 ADD (SP)+,R1      ;;ADD IN THIS DIGIT
8338 024014 000756 BR 2$                ;;LOOP
8339 024016 005726 3$: TST (SP)+          ;;CLEAN TERMINATOR FROM STACK
8340 024020 010166 000012 MOV R1,12(SP)      ;;SAVE THE RESULT
8341 024024 010237 024056 MOV R2,$HIOCT
8342 024030 012602 MOV (SP)+,R2      ;;POP STACK INTO R2
8343 024032 012601 MOV (SP)+,R1      ;;POP STACK INTO R1
8344 024034 012600 MOV (SP)+,R0      ;;POP STACK INTO R0
8345 024036 000002 RTI                ;;RETURN
8346 024040 005726 4$: TST (SP)+          ;;CLEAN PARTIAL FROM STACK
8347 024042 105010 CLRB (R0)          ;;SET A TERMINATOR
8348 024044 104401 TYPE                ;;TYPE UP THRU THE BAD CHAR.
8349 024046 000000 5$: .WORD 0
8350 024050 104401 001226 TYPE , $QUES      ;;'"?' 'CR' & 'LF'
8351 024054 000730 BR 1$                ;;TRY AGAIN
8352 024056 000000 $HIOCT: .WORD 0      ;;HIGH ORDER BITS GO HERE
8353          .SBTTL TRAP DECODER
8354
8355          ;*****
8356          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8357          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8358          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8359          ;*GO TO THAT ROUTINE.
8360
8361 024060 010046 $TRAP: MOV R0,-(SP)      ;;SAVE R0
8362 024062 016600 000002 MOV 2(SP),R0      ;;GET TRAP ADDRESS
8363 024066 005740 TST -(R0)          ;;BACKUP BY 2
8364 024070 111000 MOV (R0),R0      ;;GET RIGHT BYTE OF TRAP
8365 024072 006300 ASL R0                ;;POSITION FOR INDEXING
8366 024074 016000 024114 MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
8367 024100 000200 RTS R0                ;;GO TO ROUTINE

```



```

8368
8369
8370          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8371
8372 024102 011646          $TRAP2: MOV      (SP),-(SP)      ;;MOVE THE PC DOWN
8373 024104 016666 000004 000002  MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
8374 024112 000002          RTI                          ;;RESTORE THE PSW
8375
8376          .SBTTL TRAP TABLE
8377
8378          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8379          ;*BY THE "TRAP" INSTRUCTION.
8380
8381          :          ROUTINE
8382          :          -----
8383 024114 024102          $TRPAD: .WORD      $TRAP2
8384 024116 022266          $TYPE      ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
8385 024120 021640          $TYPOC     ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8386 024122 021614          $TYPOS     ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
8387 024124 021654          $TYPON     ;;CALL=TYPON     TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
8388 024126 022042          $TYPDS     ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
8389
8390 024130 023066          $GTSWR     ;;CALL=GTSWR     TRAP+6(104406) GET SOFT-SWR SETTING
8391
8392 024132 023016          $CKSWR     ;;CALL=CKSWR     TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
8393 024134 023300          $RDCHR     ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
8394 024136 023420          $RDLIN     ;;CALL=RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
8395 024140 023720          $RDOCT     ;;CALL=RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
8396          .SBTTL POWER DOWN AND UP ROUTINES
8397
8398          ;*****
8399          ;POWER DOWN ROUTINE
8400 024142 012737 024306 000024 $PWRDN: MOV      #SILLUP,@PWRVEC ;;SET FOR FAST UP
8401 024150 012737 000340 000026  MOV      #340,@PWRVEC+2 ;;PRIO:7
8402 024156 010046          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
8403 024160 010146          MOV      R1,-(SP)      ;;PUSH R1 ON STACK
8404 024162 010246          MOV      R2,-(SP)      ;;PUSH R2 ON STACK
8405 024164 010346          MOV      R3,-(SP)      ;;PUSH R3 ON STACK
8406 024166 010446          MOV      R4,-(SP)      ;;PUSH R4 ON STACK
8407 024170 010546          MOV      R5,-(SP)      ;;PUSH R5 ON STACK
8408 024172 017746 154742          MOV      @SWR,-(SP)      ;;PUSH @SWR ON STACK
8409 024176 010637 024312          MOV      SP,$SAVR6      ;;SAVE SP
8410 024202 012737 024214 000024  MOV      #SPWRUP,@PWRVEC ;;SET UP VECTOR
8411 024210 000000          HALT
8412 024212 000776          BR      -2          ;;HANG UP
8413
8414          ;*****
8415          ;POWER UP ROUTINE
8416 024214 012737 024306 000024 $PWRUP: MOV      #SILLUP,@PWRVEC ;;SET FOR FAST DOWN
8417 024222 013706 024312          MOV      $SAVR6,SP      ;;GET SP
8418 024226 005037 024312          CLR      $SAVR6        ;;WAIT LOOP FOR THE TTY
8419 024232 005237 024312          1$: INC      $SAVR6        ;;WAIT FOR THE INC
8420 024236 001375          BNE     1$            ;;OF WORD
8421 024240 012677 154674          MOV      (SP)+,@SWR     ;;POP STACK INTO @SWR
8422 024244 012605          MOV      (SP)+,R5      ;;POP STACK INTO R5
8423 024246 012604          MOV      (SP)+,R4      ;;POP STACK INTO R4

```

```
8424 024250 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
8425 024252 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
8426 024254 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
8427 024256 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
8428 024260 012737 024142 000024  MOV      #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
8429 024266 012737 000340 000026  MOV      #340,@#PWRVEC+2 ;;PRIO:7
8430 024274 104401      TYPE                                ;;REPORT THE POWER FAILURE
8431 024276 024314      $PWRMG: .WORD $POWER      ;;POWER FAIL MESSAGE POINTER
8432 024300 012716      MOV      (PC)+,(SP)      ;;RESTART AT RSTRTA
8433 024302 002744      $PWRAD: .WORD RSTRTA      ;;RESTART ADDRESS
8434 024304 000002      RTI
8435 024306 000000      $ILLUP: HALT              ;;THE POWER UP SEQUENCE WAS STARTED
8436 024310 000776      BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
8437 024312 000000      $SAVR6: 0                ;;PUT THE SP HERE
8438 024314 005015 047520 042527  $POWER: .ASCIZ <15><12>'POWER'
8439 024322 000122
8440
8441 024324 004737 027244      .EVEN
      JSR      PC,DCACHE      ;;DISABLE CACHE ;;++D
```

```

8442
8443
8444
8445
8446
8447
8448
8449 024330 012701 037312
8450 024334 005002
8451 024336 110221
8452 024340 005202
8453 024342 022702 000400
8454 024346 001373
8455 024350 000207
8456
8457
8458
8459
8460 024352 004737 027200
8461 024356 113700 001102
8462 024362 010037 001162
8463 024366 010137 001164
8464 024372 010237 001166
8465 024376 010637 001176
8466 024402 062737 000002 001176
8467 024410 000207
8468
8469
8470
8471
8472 024412 004737 027200
8473 024416 113700 001102
8474 024422 010037 001162
8475 024426 010137 001164
8476 024432 010237 001166
8477 024436 010337 001170
8478 024442 010437 001172
8479 024446 010637 001176
8480 024452 062737 000002 001176
8481 024460 000207
8482
8483
8484
8485
8486 024462 010037 001162
8487 024466 010137 001164
8488 024472 010237 001166
8489 024476 000207
8490
8491
8492
8493
8494 024500 005037 001202
8495 024504 113737 030324 001202
8496 024512 113700 001102
8497 024516 010037 001162

```

```

*****
:COMMON DH11 SERVICE ROUTINES
*****

;THIS ROUTINE IS CALLED DURING START UP TO LOAD THE XMITTER
;OUTPUT BUFFER WITH A BINARY COUNT TEST PATTERN

LDTBF1: MOV    #TBUF,R1      ;POINT TO START OF BUFFER
        CLR    R2          ;INIT DATA BYTE GENERATOR
1$:     MOVB   R2,(R1)+     ;LOAD ONE CHAR
        INC    R2          ;GENERATE NEXT CHAR
        CMP    #400,R2     ;LOADED 256(10) BYTES
        BNE   1$          ;BR IF NOT
        RTS    PC         ;RETURN TO START TESTING

;THIS ROUTINE SETS UP THE ERROR INFORMATION REQUIRED BY ANY TEST
;USING A 'DH1' HEADER

SUER1:  JSR    PC,SAPS     ;SAVE THE ERROR PSW
        MOVB   $STNM,R0   ;SAVE THE TEST NO.
        MOV    R0,$REG0    ;SAVE THE TEST NO. FOR ERROR PRINT
        MOV    R1,$REG1    ;SAVE THE DH11 ADDR
        MOV    R2,$REG2    ;SAVE THE REG ADDRESS
        MOV    R6,$REG6    ;SAVE THE SP
        ADD    #2,$REG6    ;CORRECT FOR CALLING JSR
        RTS    PC         ;RETURN TO CALLING ROUTINE

;THIS ROUTINE IS CALLED BY THOSE TESTS USING A 'DH2' HEADER TO
;SAVE THE ERROR INFORMATION IN 'DT2'

SUER2:  JSR    PC,SAPS     ;SAVE THE ERROR PSW
SUER2A: MOVB   $STNM,R0   ;GET THE TEST NO.
        MOV    R0,$REG0    ;SAVE THE REGISTERS-TEST#
        MOV    R1,$REG1    ;SAVE THE DH ADDRESS
        MOV    R2,$REG2    ;SAVE THE REGISTER ADDRESS
        MOV    R3,$REG3    ;SAVE THE WAS DATA
        MOV    R4,$REG4    ;SAVE THE S/B DATA
        MOV    R6,$REG6    ;SAVE THE STACK POINTER
        ADD    #2,$REG6    ;CORRECT FOR CALLING JSR
        RTS    PC         ;RETURN TO REPORT ERROR

;THIS ROUTINE IS CALLED TO SET UP ERROR INFORMATION FOR THE
;BUS ERROR AND RSVD INSTR ERROR ROUTINES

SUER3:  MOV    R0,$REG0    ;SAVE THE REGS
        MOV    R1,$REG1
        MOV    R2,$REG2
        RTS    PC         ;RETURN TO REPORT ERROR

;THIS ROUTINE IS CALLED TO SET UP ERROR INFORMATION FOR THE
;CAR/BCR MEMORY PATTERNS TESTS

SUER4:  CLR    $TMP0      ;SAVE THE LINE NO. WRITTEN
        MOVB   LINEA,$TMP0
        MOVB   $STNM,R0   ;SAVE THE TEST NUMBER
        MOV    R0,$REG0    ;SAVE THE REGISTER INFORMATION

```

```

8498 024522 010137 001164      MOV    R1,$REG1
8499 024526 010237 001166      MOV    R2,$REG2
8500 024532 010337 001170      MOV    R3,$REG3
8501 024536 010437 001172      MOV    R4,$REG4
8502 024542 000207              RTS     PC                ;RETURN TO PATTERNS TEST
8503
8504                               ;THIS ROUTINE IS CALLED TO SELECT A NEW LINE NO. BASED ON THE
8505                               ;VALUE OF THE LINE SELECTION PARAMETER
8506
8507                               ;CALLING SEQUENCE:
8508
8509                               ;JSR    PC,SELINE      ;CALL THE ROUTINE
8510                               ;BR     1$              ;EXIT BRANCH-ROUTINE MOVES THE RETURN
8511                               ;PC AROUND THIS BR IF MORE LINES ARE
8512                               ;YET TO BE TESTED
8513
8514 024544 105737 030323      SELINE: TSTB   LINE+1      ;FIRST TIME THROUGH FOR ANY TEST ?
8515 024550 001010              BNE     1$                ;BR IF NOT
8516 024552 105137 030323      COMB   LINE+1            ;SET ENTRY FLAG
8517 024556 012737 000001 027314  MOV    #1,LINMSK        ;INIT SELECT TEST MASK TO TEST LINE 00
8518 024564 105037 030322      CLR   LINE              ;START WITH LINE #00
8519 024570 000405              BR     2$                ;GO TEST FOR LINE #00
8520 024572 105237 030322      1$:   INCB  LINE         ;GENERATE NEW LINE NO.
8521 024576 006337 027314      ASL   LINMSK            ;SHIFT SELECT MASK TO TEST NXT LINE
8522 024602 001407              BEQ    3$                ;RETURN TO EXIT BRANCH - ALL LINES DONE
8523 024604 033737 027314 027312 2$:   BIT   LINMSK,LINSEL    ;IS THE LINE SELECTED FOR TEST ??
8524 024612 001767              BEQ    1$                ;BR IF NOT
8525 024614 062716 000002      ADD   #2,(SP)          ;MOVE RETURN PC AROUND EXIT BRANCH
8526 024620 000402              BR     4$                ;RETURN TO TEST SELECTED LINE
8527 024622 005037 030322      3$:   CLR   LINE         ;INIT ENTRY FLAG AND LINE NO. TO 000
8528 024626 142777 000017 002446 4$:   BIC   #17,@DHADR      ;INIT LINE SELECT BITS IN "SCR"
8529 024634 000207              RTS     PC                ;RETURN TO CALLING TEST

```

```

8530
8531                               ;THIS ROUTINE IS CALLED TO CONVERT EITHER THE "DH" NUMBER OR THE
8532                               ;"LINE" NUMBER TO TWO ASCII CHARACTERS AND MOVE THEM INTO A
8533                               ;PARTICULAR MESSAGE BUFFER FOR ERROR REPORTING
8534
8535                               ;CALLING SEQUENCE
8536
8537                               ;JSR    R5,SUNUM       ;CALL TO THIS ROUTINE
8538                               ;ADDR1   ;ADDRESS OF THE NUMBER TO BE CONVERTED
8539                               ;ADDR2   ;ADDRESS OF THE MSG BUFFER SLOT
8540
8541 SUNUM:
8542 024636 010046      MOV    R0,-(SP)        ;;PUSH R0 ON STACK
8543 024640 010146      MOV    R1,-(SP)        ;;PUSH R1 ON STACK
8544 024642 010246      MOV    R2,-(SP)        ;;PUSH R2 ON STACK
8545 024644 012500      MOV    (R5)+,R0        ;GET ADDRESS OF NUMBER
8546 024646 012501      MOV    (R5)+,R1        ;GET MSG BUFFER ADDR
8547 024650 111000      MOV   (R0),R0          ;GET NO. TO BE CONVERTED
8548 024652 010002      MOV    R0,R2           ;SAVE IT IN R2
8549 024654 006202      ASR   R2               ;SHIFT MSD TO LSD POSITION
8550 024656 006202      ASR   R2
8551 024660 006202      ASR   R2
8552 024662 042702 177770      BIC   #177770,R2      ;CLR JUNK BITS
8553 024666 062702 000060      ADD   #60,R2          ;MAKE IT ASCII

```

```

8554 024672 110221          MOVB    R2,(R1)+      ;PUT IT IN MSG BUFFER
8555 024674 042700 177770    BIC     #177770,R0    ;CLR JUNK FROM LSD
8556 024700 062700 000060    ADD     #60,R0        ;MAKE IT ASCII
8557 024704 110011          MOVB    R0,(R1)      ;PUT LSD IN THE BUFFER
8558 024706 012602          MOV     (SP)+,R2     ;POP STACK INTO R2
8559 024710 012601          MOV     (SP)+,R1     ;POP STACK INTO R1
8560 024712 012600          MOV     (SP)+,R0     ;POP STACK INTO R0
8561 024714 000205          RTS     R5           ;RETURN TO CALLER
8562
8563                          ;THIS ROUTINE IS CALLED TO CLEAR THE "CAR" AND "BCR" MEMORIES
8564                          ;IT ASSUMES THAT THE ADDRESS OF THE "SCR" IS IN R1
8565
8566 024716 005037 001220    CLCABC: CLR    $TMP7      ;INIT A COUNTER
8567 024722 113711 001220    1$:  MOVB   $TMP7,(R1)    ;SELECT A LINE
8568 024726 005061 000006    CLR    CAR(R1)         ;CLEAR A CAR LOCATION
8569 024732 005061 000010    CLR    BCR(R1)         ;CLEAR A BCR LOCATION
8570 024736 005237 001220    INC    $TMP7           ;GENERATE NEW LINE NO.
8571 024742 022737 000020 001220    CMP    #20,$TMP7      ;DONE ALL LINES ?
8572 024750 001364          BNE    1$             ;BR IF NOT
8573 024752 142711 000017    BICB   #17,(R1)       ;SET "SCR" TO SELECT LINE 00
8574 024756 000207          RTS     PC            ;RETURN TO CALLER
8575
8576                          ;THIS ROUTINE IS CALLED TO LOAD THE "BCR" MEMORY WITH ALL ONES
8577                          ;IT ASSUMES THAT THE ADDRESS OF THE SCR IS IN R1
8578
8579 024760 005037 001220    LDBCR: CLR    $TMP7      ;INIT A COUNTER
8580 024764 113711 001220    1$:  MOVB   $TMP7,(R1)    ;SELECT A LINE
8581 024770 012761 177777 000010    MOV    #-1,BCR(R1)    ;LOAD BCR LOC. WITH 177777
8582 024776 005237 001220    INC    $TMP7           ;GENERATE NEXT LINE NO.
8583 025002 022737 000020 001220    CMP    #20,$TMP7      ;DONE ALL LINES ?
8584 025010 001365          BNE    1$             ;BR IF NOT
8585 025012 142711 000017    BICB   #17,(R1)       ;SET "SCR" TO SELECT LINE 00
8586 025016 000207          RTS     PC            ;RETURN TO CALLER
8587
8588                          ;THIS ROUTINE CALLED TO SET UP FOR PARITY TESTS
8589
8590 025020 012737 000020 001212    SUPPAR: MOV    #20,$TMP4  ;SET UP FOR 16. LINES
8591 025026 105011          CLR    (R1)           ;INIT SCR TO START AT LINE 00
8592 025030 005002          CLR    R2             ;INIT INDEX REGISTER FOR RBUF (EVEN)
8593 025032 012703 000200    MOV    #200,R3        ;SET UP CONSTANT
8594 025036 012704 000001    MOV    #1,R4          ;INIT INDEX REG FOR RBUF (ODD)
8595 025042 012761 037312 000006 1$:  MOV    #TBUF,CAR(R1)   ;LOAD BUS ADDRESS REWG
8596 025050 013761 001216 000010    MOV    $TMP6,BCR(R1)  ;LOAD BYTE COUNT REG
8597 025056 013761 001214 000004    MOV    $TMP5,LPR(R1)  ;LOAD LINE PARAMETERS
8598 025064 105062 036312    CLR    RBUF(R2)       ;INIT DATA BYTE IN RBUF TO START AT 000
8599 025070 110364 036312    MOVB   R5,RBUF(R4)    ;SET CONSTANT IN HIGH BYTE
8600 025074 005211          INC    (R1)           ;SELECT NEXT LINE
8601 025076 005203          INC    R3             ;GENERATE NEW CONSTANT
8602 025100 062702 000002    ADD    #2,R2          ;UPDATE POINTERS TO RBUF (EVEN/ODD)
8603 025104 062704 000002    ADD    #2,R4
8604 025110 005337 001212    DEC    $TMP4          ;COUNT ONE LINE SETUP
8605 025114 001352          BNE    1$             ;BR TILL ALL 16. SET UP
8606 025116 012704 027562    MOV    #MULPTB,R4     ;SET UP TABLE POINTER
8607 025122 013724 001216 2$:  MOV    $TMP6,(R4)+    ;SET UP BYTE COUNT ENTRY
8608 025126 022704 027622    CMP    #MULPTB+40,R4 ;SET UP ALL COUNTS ?
8609 025132 001373          BNE    2$            ;BR IF NOT

```

```
8610 025134 105011          CLR      (R1)          ;INIT SCR TO LINE 00
8611 025136 000207          RTS      PC            ;RETURN TO PA. TEST
8612
8613          ;THIS ROUTINE AUTOSIZES THE SYSTEM TO DETERMINE THE ADDRESSES AND
8614          ;VECTORS OF THE DH11'S AND MODEM CONTROL'S.
8615
8616 025140 010046          AUTOSZ: MOV      R0,-(SP)
8617 025142 005003          CLR      R3
8618 025144 012702 030042          MOV      #DHADRS,R2          ;POINT TO BEGINNING OF TABLE
8619 025150 005022          25$: CLR      (R2)+          ;CLEAR AUTOSIZER TABLES.
8620 025152 005203          INC      R3
8621 025154 020327 000102          CMP      R3,#102          ;HAVE WE CLEARED ALL ENTRIES?
8622 025160 001373          BNE      25$          ;BRANCH IF NOT.
8623 025162 013746 000004          MOV      @#4,-(SP)          ;SAVE TRAP VECTOR.
8624 025166 012737 025274 000004          MOV      #4$,@#4          ;SETUP FOR NON-EXISTENT MEMORY TRAP.
8625 025174 012703 030144          MOV      #DMADRS,R3          ;SETUP DM ADDRESS TABLE POINTER.
8626 025200 012702 030042          MOV      #DHADRS,R2          ;SET UP DH ADDRESS TABLE POINTER.
8627
8628 025204 012701 160020          MOV      #160020,R1          ;R1=FIRST ADDRESS TO BE TESTED.
8629
8630 025210 005711          1$: TST      (R1)          ;SEE IF ADDRESS IN R1 RESPONDS.
8631 025212 005761 000016          TST      16(R1)          ;CHECK TO SEE IF DEVICE IS MODULO 20.
8632 025216 052711 004000          BIS      #4000,(R1)          ;IF IT IS, CONTINUE
8633          ;AND CHECK TO SEE
8634 025222 052711 001000          BIS      #1000,(R1)          ;IF THIS ADDRESS CONTAINS
8635 025226 052711 002000          BIS      #2000,(R1)          ;A DH-11.
8636 025232 032711 003000          BIT      #3000,(R1)          ;CHECK TO INSURE THESE BITS SET.
8637 025236 001410          BEQ      3$          ;IF NOT, BRANCH.
8638          ;SET THE MAINTENANCE BIT, THE NON-
8639 025240 052711 000400          BIS      #400,(R1)          ;EXISTENT MEMORY BIT AND THE CLEAR
8640          ;NON-EXISTENT MEMORY INTERRUPT BIT.
8641 025244 032711 002400          BIT      #2400,(R1)          ;IS THIS A DH-11? (BITS 8 AND 10 SHOULD
8642          ;CLEAR IF THIS IS A DH11.)
8643
8644 025250 001003          BNE      3$          ;IF NOT, CHECK TO SEE IF THIS IS A MODEM CONTROL.
8645 025252 042711 001000          BIC      #1000,(R1)          ;CLEAR MAINTENANCE BIT.
8646 025256 010122          MOV      R1,(R2)+          ;SAVE THE ADDRESS IN THE DH ADR TABLE.
8647
8648
8649 025260 020127 163760          3$: CMP      R1,#163760          ;HAVE WE REACHED THE TOP OF THE FLOATING ADDRESSES.
8650 025264 001406          BEQ      5$          ;IF YES, GET OUT.
8651 025266 062701 000020          ADD      #20,R1          ;IF NOT, UPDATE ADDRESS AND
8652 025272 000746          BR       1$          ;GO CHECK IT.
8653
8654 025274 012716 025260          4$: MOV      #3$,(SP)          ;IF DH ADDRESS DOES NOT RESPOND, GO TO 3$.
8655 025300 000002          RTI
8656
8657          ;TEST FOR MODEM CONTROL ADDRESS
8658
8659 025302 012737 025334 000004          5$: MOV      #6$,@#4          ;SETUP FOR NON-EXISTENT MEMORY TRAP.
8660 025310 012701 170500          MOV      #170500,R1          ;R1=FIRST ADDRESS TO BE TESTED.
8661 025314 005711          21$: TST      (R1)          ;SEE IF ADDRESS RESPONDS.
8662 025316 010123          MOV      R1,(R3)+          ;IF IT DOES, THIS IS A MODEM CONTROL,
8663          ;SO SAVE THE ADDRESS.
8664 025320 020127 170670          23$: CMP      R1,#170670          ;HAVE WE REACHED THE TOP OF THE MODEM ADDRESSES?
8665 025324 001406          BEQ      22$          ;IF YES, GET OUT.
```

```
8666 025326 062701 000010      ADD    #10,R1      ;IF NOT, UPDATE ADDRESS AND
8667 025332 000770      BR     21$        ;GO CHECK IT.
8668
8669 025334 012716 025320      6$:   MOV    #73$,(SP) ;IF DM ADDRESS DOES NOT RESPOND, GO TO 23$.
8670 025340 000002      RTI
8671
8672 025342 012637 000004      22$:  MOV    (SP)+,@#4   ;RESTORE TRAP VECTOR.
8673 025346 162702 030042      SUB    #DHADRS,R2  ;HAVE WE FOUND ANY DH11'S AT ALL?
8674 025352 001003      BNE    7$        ;IF YES, BRANCH
8675 025354 104401 035604      TYPE  ,MSG1      ;NO DH11'S WERE FOUND,
8676 025360 000000      HALT
8677
8678 025362 006202      7$:   ASR    R2        ;R2 NOW CONTAINS THE NUMBER
8679 025364 005000      CLR    R0        ;OF DH'S FOUND.
8680 025366 006100      8$:   ROL    R0        ;FILL R0 WITH 1'S
8681 025370 005200      INC    R0        ;CORRESPONDING TO
8682 025372 005302      DEC    R2        ;THE NUMBER OF DH'S
8683 025374 005702      TST    R2        ;FOUND.
8684 025376 001373      BNE    8$
8685 025400 010037 030314      MOV    R0,$DHSEL ;$DHSEL CONTAINS THE DH SELECTION PARAMETER.
8686                                     ;IE. ALL DH'S FOUND WILL BE TESTED.
8687
8688                                     ;FIND DH VECTOR:
8689 025404 012702 030042      MOV    #DHADRS,R2 ;SETUP POINTER TO BEGINNING OF DH
8690 025410 012705 030104      MOV    #DHVEC,R5  ;ADDRESS TABLE AND VECTOR TABLE.
8691 025414 012737 000340 000022      MOV    #340,@#IOTVEC+2 ;SET IOT TRAP PRIORITY TO 7.
8692 025422 012737 025532 000020      MOV    #12$,@#IOTVEC ;SETUP IOT TRAP VECTOR.
8693 025430 012703 000300      MOV    #300,R3    ;START OF FLOATING VECTORS
8694 025434 012704 000302      MOV    #302,R4    ;PC OF IOT INSTR.
8695
8696 025440 010423      9$:   MOV    R4,(R3)+  ;FILL VECTOR AREA WITH ADDRESS
8697                                     ;OF NEXT INSTR (.+2)
8698 025442 012724 000004      MOV    #4,(R4)+   ;NEXT INSTRUCTION IS AN IOT TRAP.
8699 025446 022324      CMP    (R3)+,(R4)+ ;UPDATE R3+R4.
8700 025450 020427 001000      CMP    R4,#1000   ;HAVE WE REACHED TO TOP OF THE
8701                                     ;VECTOR SPACE?
8702 025454 101771      BLOS   9$        ;IF NOT, REPEAT PROCESS.
8703
8704 025456 005712      10$:  TST    (R2)      ;HAVE WE CHECK ALL DH'S?
8705 025460 001441      BEQ    13$      ;IF YES, GET OUT + CHECK FOR MODEM CONTROL'S VECTORS.
8706
8707 025462 005037 177776      CLR    PS        ;ZERO CPU PRIORITY.
8708 025466 052772 001000 000000      BIS    #1000,@(R2) ;SET MAINTENANCE BIT
8709 025474 052772 000300 000000      BIS    #300,@(R2) ;ATTEMPT TO CAUSE RECEIVER
8710                                     ;INTERRUPT.
8711 025502 005000      CLR    R0
8712
8713 025504 005200      11$:  INC    R0        ;WAIT...
8714 025506 001376      BNE    11$
8715 025510 104401 035634      TYPE  ,MSG2      ;ERROR MSG-NO DH RECEIVER INTERRUPT OCCURRED.
8716 025514 052772 004000 000000      BIS    #4000,@(R2) ;DO A MASTER CLEAR
8717 025522 042772 001000 000000      BIC    #1000,@(R2) ;CLEAR MAINTENANCE BIT
8718 025530 000752      BR     10$
8719
8720 025532 011601      12$:  MOV    (SP),R1
8721 025534 042701 000007      BIC    #7,R1     ;CLEAR GARBAGE.
```

```
8722 025540 010125          MOV    R1,(R5)+      ;SAVE VECTOR ADDRESS.
8723 025542 022626          CMP    (SP)+,(SP)+  ;POP STACK
8724 025544 012716 025456    MOV    #10$(,SP)    ;SETUP FOR RETURN.
8725 025550 052772 004000 000000  BIS    #4000,@(R2)  ;DO A MASTER CLEAR
8726 025556 042732 001000          BIC    #1000,@(R2)+ ;CLEAR MAINTENANCE BIT.
8727 025562 000002          RTI
8728
8729                          ;FIND MODEM CONTROL VECTORS:
8730
8731 025564 012702 030144    13$:  MOV    #DMADRS,R2   ;SET POINTERS TO BEGINNING OF
8732 025570 012705 030206    MOV    #DMVEC,R5    ;ADR TABLE & VECTOR TABLE.
8733 025574 012737 025656 000020  MOV    #16$,@#IOTVEC ;SET IOT TRAP VECTOR.
8734
8735 025602 005712          14$:  TST    (R2)      ;HAVE WE CHECKED ALL DM'S?
8736 025604 001441          BEQ    17$          ;IF YES, GET OUT.
8737 025606 005037 177776    CLR    PS           ;ZERO CPU PRIORITY
8738 025612 052772 001000 000000  BIS    #1000,@(R2)  ;SET MAINTENANCE BIT.
8739 025620 052772 000300 000000  BIS    #300,@(R2)  ;ATTEMPT TO CAUSE INTERRUPT.
8740 025626 005000          CLR    R0
8741
8742 025630 005200          15$:  INC    R0           ;WAIT....
8743 025632 001376          BNE    15$
8744 025634 104401 035701    TYPE  ,MSG3        ;ERROR MSG - NO MODEM CONTROL INTERRUPT OCCURRED.
8745 025640 052772 004000 000000  BIS    #4000,@(R2)  ;CLEAR BITS PREVIOUSLY SET.
8746 025646 042772 001000 000000  BIC    #1000,@(R2)  ;CLEAR MAINTENANCE BIT.
8747 025654 000752          BR    14$
8748
8749 025656 011601          16$:  MOV    (SP),R1      ;CALCULATE VECTOR ADDRESS.
8750 025660 162701 000004    SUB    #4,R1        ;SAVE VECTOR ADDRESS.
8751 025664 010125          MOV    R1,(R5)+    ;POP STACK.
8752 025666 022626          CMP    (SP)+,(SP)+ ;SETUP FOR RETURN.
8753 025670 012716 025602    MOV    #14$(,SP)   ;CLEAR BITS PREVIOUSLY SET.
8754 025674 052772 004000 000000  BIS    #4000,@(R2)  ;CLEAR MAINTENANCE BIT AND
8755 025702 042732 001000          BIC    #1000,@(R2)+ ;POINT TO NEXT MODEM CONTROL ADDRESS.
8756
8757 025706 000002          RTI
8758
8759 025710 012737 021004 000020  17$:  MOV    #$$SCOPE,@#IOTVEC ;RESTORE IOT VECTOR FOR SCOPE ROUTINE.
8760 025716 012600          MOV    (SP)+,R0    ;RESTORE R0.
8761 025720 012703 000300          MOV    #300,R3     ;START OF FLOATING VECTORS.
8762 025724 012704 000302          MOV    #302,R4
8763
8764 025730 010423          18$:  MOV    R4,(R3)+    ;FILL VECTOR AREA WITH ADDRESS OF NEXT
8765                          ;INSTRUCTION (.+2).
8766 025732 012724 000000    MOV    #0,(R4)+    ;NEXT INSTRUCTION IS A HALT.
8767 025736 022324          CMP    (R3)+,(R4)+ ;UPDATE R3 & R4.
8768 025740 020427 001000    CMP    R4,#1000    ;ARE WE DONE?
8769 025744 101771          BLOS  18$         ;IF NOT, REPEAT UNTIL ADDRESSES
8770                          ;377 TO 777 ARE DONE.
8771 025746 013701 030104    MOV    @#DHVEC,R1   ;LET R1 POINT TO 1ST DH VECTOR ADDRESS.
8772 025752 005737 030106    TST    @#DHVEC+2    ;IS THERE MORE THAN ONE VECTOR?
8773 025756 001403          BEQ    26$         ;BRANCH IF NOT.
8774 025760 163701 030106    SUB    @#DHVEC+2,R1 ;DETERMINE NUMBER OF ADDRESSES
8775                          ;BETWEEN DH VECTORS (10(8) OR 20(8)).
8776 025764 005401          NEG    R1           ;MAKE R1 POSITIVE.
8777 025766 010137 030306    26$:  MOV    R1,ADRVEC   ;SAVE THAT NUMBER.
```



```

8778 025772 032777 000002 153140      BIT      #BIT1,@SWR      ;SHOULD DEVICE MAP BE TYPED OUT?
8779 026000 001442                      BEQ      20$          ;IF NOT, RETURN.
8780 026002 104401                      TYPE                                ;TYPEOUT MAP OF DH & MODEM CONTROL'S
8781 026004 035754                      DEVMAP                               ;FOUND.
8782 026006 012701 030042              MOV      #DHADRS,R1    ;R1=BEGINNING OF DH ADDRESS TABLE.
8783 026012 012702 030104              MOV      #DHVEC,R2    ;R2=BEGINNING OF DH VECTOR TABLE.
8784 026016 012703 030144              MOV      #DMADRS,R3   ;R3=BEGINNING OF MODEM CONTROL ADDRESS TABLE.
8785 026022 012704 030206              MOV      #DMVEC,R4    ;R4=BEGINNING OF MODEM CONTROL VECTOR TABLE.
8786
8787 026026 012146                      19$:    MOV      (R1)+,-(SP) ;MOVE DATA TO BE TYPED
8788 026030 104403                      TYPOS                                ;TYPE DATA
8789 026032 006                          .BYTE 6
8790 026033 001                          .BYTE 1
8791 026034 012246                      MOV      (R2)+,-(SP) ;MOVE DATA TO BE TYPED
8792 026036 104403                      TYPOS                                ;TYPE DATA
8793 026040 005                          .BYTE 5
8794 026041 000                          .BYTE 0
8795 026042 104401 035750              TYPE      ,SPACE
8796 026046 012346                      MOV      (R3)+,-(SP) ;MOVE DATA TO BE TYPED.
8797 026050 104403                      TYPOS                                ;TYPE DATA.
8798 026052 006                          .BYTE 6
8799 026053 001                          .BYTE 1
8800 026054 104401 035750              TYPE      ,SPACE
8801 026060 012446                      MOV      (R4)+,-(SP) ;MOVE DATA TO BE TYPED.
8802 026062 104403                      TYPOS                                ;TYPE DATA.
8803 026064 005                          .BYTE 5
8804 026065 000                          .BYTE 0
8805 026066 104401                      TYPE                                ;TYPE A CARRIAGE RETURN & LINE FEED.
8806 026070 001227                      $CRLF
8807 026072 005711                      TST      (R1)          ;HAVE WE TYPED ALL DH ENTRIES?
8808 026074 001354                      BNE      19$          ;IF NOT, DO IT AGAIN.
8809 026076 005713                      TST      (R3)          ;HAVE WE TYPED ALL DM ENTRIES?
8810 026100 001352                      BNE      19$          ;IF NOT - ONE MORE TIME.
8811 026102 104401 001227              TYPE      , $CRLF
8812 026106 000207                      20$:    RTS      PC      ;IF YES, GO BACK TO MAIN PROGRAM.
8813
8814 ;THIS ROUTINE IS USED TO ACCEPT INPUT PARAMETERS FROM THE CONSOLE
8815 ;TELETYPE
8816
8817 026110 104401                      INPARA: TYPE
8818 026112 035507                      VCWC                                ;"ASK FOR NO. ADDRESSES BETWEEN VECTORS"
8819 026114 104412                      RDOCT                               ;READ OCTAL NO. FM TTY
8820 026116 012600                      MOV      (SP)+,RO      ;GET THE NO. HE TYPED
8821 026120 001407                      BEQ      3$           ;BR IF HE TYPED <CR>
8822 026122 022700 000010              CMP      #10,RO       ;10(8) ADDRESSES BETWEEN VECTORS ?
8823 026126 001406                      BEQ      4$           ;BR IF YES
8824 026130 022700 000020              CMP      #20,RO       ;20(8) ADDRESSES BETWEEN VECTORS ??
8825 026134 001403                      BEQ      4$           ;BR IF YES
8826 026136 000764                      BR      INPARA        ;ASK ALL OVER AGAIN
8827 026140 012700 000020              3$:    MOV      #20,RO  ;SET UP CONSTANT FOR 20(8) ADDRESSES
8828 026144 000207                      4$:    RTS      PC      ;RETURN TO CALLER
8829
8830
8831
8832 026146 004737 027244              INPARC: JSR      PC,DCACHE ;DISABLE CACHE ;:++D
8833 026152 012700 177777              MOV      #-1,RO      ;SET FLAG IN RO

```

```
8834 026156 000137 0C2164          JMP      BEGINA          ;GO ASK FOR SELECT PARAMETER
8835
8836 026162 004737 027244          INPARX: JSR      PC,DCACHE ;DISABLE CACHE ;:++D
8837 026166 012737 177777 030000    MOV      #-1,VCFLG      ;SET SETUP FLAG
8838 026174 000137 002164          JMP      BEGINA          ;GO START UP
8839
8840 026200 013701 030042          INPAR:  MOV      @#DHADRS,R1 ;MOVE ADDRESS OF FIRST DH INTO R1.
8841 026204 032777 000001 152726    BIT      #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
8842 026212 001405                    BEQ      2$              ;BRANCH IF NOT.
8843 026214 104401          1$:      TYPE                    ;ASK FOR DEVICE ADDRESS
8844 026216 035064                    INMSG1
8845 026220 104412                    RDOCT
8846 026222 012601                    MOV      (SP)+,R1        ;READ IN WHAT IS TYPED
8847 026224 001403                    BEQ      INPAR1          ;GET THE NO. HE TYPED
8848 026226 004737 026412          2$:      JSR      PC,CHKADR  ;BR IF DEFAULT
8849 026232 000770                    BR       1$              ;GO CHECK VALIDITY OF THE ADDR
8850                                     ;ERROR BRANCH
8851 026234 013701 030104          INPAR1: MOV      @#DHVEC,R1 ;MOVE FIRST DH VECTOR INTO R1.
8852 026240 032777 000001 152672    BIT      #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
8853 026246 001405                    BEQ      2$              ;BRANCH IF NOT.
8854 026250 104401          1$:      TYPE                    ;ASK FOR VECTOR ADDRESS
8855 026252 035130                    INMSG2
8856 026254 104412                    RDOCT
8857 026256 012601                    MOV      (SP)+,R1        ;READ IN WHAT HE TYPES
8858 026260 001403                    BEQ      INPAR3          ;GET THE ADDRESS
8859 026262 004737 026526          2$:      JSR      PC,CHKVCT ;BR IF DEFAULT
8860 026266 000770                    BR       1$              ;GO CHECK VALIDITY OF VECTOR
8861                                     ;ERROR BRANCH
8862 026270 013701 030314          INPAR3: MOV      @#$HSEL,R1 ;MOVE DEVICE SELECTION PARAMETER INTO R1.
8863 026274 005700                    TST      R0              ;DID WE START AT 210?
8864 026276 100404                    BMI      2$              ;BRANCH IF YES.
8865 026300 032777 000001 152632    BIT      #BIT0,@SWR      ;IS PARAMETER TO BE INPUT MANUALLY?
8866 026306 001405                    BEQ      1$              ;BRANCH IF NOT.
8867 026310 104401          2$:      TYPE                    ;ASK FOR DEVICE SELECTION PARAMETER
8868 026312 035177                    INMSG3
8869 026314 104412                    RDOCT
8870 026316 012601                    MOV      (SP)+,R1        ;READ IN WHAT HE TYPES
8871 026320 001402                    BEQ      INPAR4          ;GET THE SELECT PARAMETER
8872 026322 010137 027310          1$:      MOV      R1,DHSEL    ;BR IF DEFAULT
8873                                     ;SET UP DH11 SELECTION PARAMETER
8874 026326 005700          INPAR4: TST      R0              ;DID WE START AT 210?
8875 026330 100404                    BMI      3$              ;BRANCH IF YES.
8876 026332 032777 000001 152600    BIT      #BIT0,@SWR      ;IS LINE SELECT PARAMETER TO BE INPUT MANUALLY?
8877 026340 001410                    BEQ      1$              ;BRANCH IF NO.
8878 026342 104401          3$:      TYPE                    ;ASK FOR LINE SELECT PARAMETER
8879 026344 035375                    INMSG6
8880 026346 104412                    RDOCT
8881 026350 012601                    MOV      (SP)+,R1        ;GET WHAT HE TYPES
8882 026352 001403                    BEQ      1$              ;GET PARAMETER
8883 026354 010137 027312          1$:      MOV      R1,LINSEL    ;BR IF DEFAULT
8884 026360 000403                    BR       2$              ;SET UP LINE SELECT PARAMETER
8885 026362 012737 177777 027312    1$:      MOV      #-1,LINSEL ;CONTINUE
8886 026370 032777 000400 152542    2$:      BIT      #BIT8,@SWR ;SET UP DEFAULT (ALL LINES)
8887 026376 001403                    BEQ      EXPAR          ;HALT AFTER SET UP ??
8888 026400 104401                    TYPE                    ;BR IF NOT
8889 026402 035437                    INMSG7                    ;TYPE CONTINUE MESSAGE PRIOR TO HALTING
```

```
8890 026404 000000          HALT          ;DEPRESS CONTINUE TO RESUME TESTING
8891 026406 000137 002640  EXPAR:  JMP      START2      ;GO START UP THE PROGRAM
8892
8893
8894 026412 020127 160020  CHKADR: CMP      R1,#160020      ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
8895 026416 002001          BGE      1$          ;BR IF YES
8896 026420 000437          BR       4$          ;BR IF NOT
8897 026422 020127 160420  1$:  CMP      R1,#160420      ;IS IT BELOW THE HIGH LIMIT?
8898 026426 002401          BLT      2$          ;BR IF YES
8899 026430 000433          BR       4$          ;BR IF NOT
8900 026432 032701 000017  2$:  BIT      #17,R1          ;CORRECT BOUNDARY ?
8901 026436 001030          BNE      4$          ;BR IF NOT
8902 026440 062716 000002          ADD      #2,(SP)        ;MOVE RETURN PC AROUND ERROR BRANCH
8903 026444 012702 027700          MOV      #DHADTB,R2     ;POINT TO BEGIN OF ADDR TABLE
8904 026450 032777 000001 152462 BIT      #BIT0,@SWR      ;ARE WE AUTOSIZING?
8905 026456 001011          BNE      3$          ;BRANCH IF NOT.
8906 026460 012703 030042          MOV      #DHADRS,R3     ;POINT TO BEGINNING OF AUTOSIZER
8907                                ;DH ADDRESS TABLE.
8908 026464 013704 030314          MOV      $DHSEL,R4
8909 026470 012322  6$:  MOV      (R3)+,(R2)+     ;MOVE CONTENTS OF AUTOSIZER DH TABLE
8910                                ;TO THE TABLE USED BY PROGRAM.
8911 026472 006204          ASR      R4
8912 026474 005704          TST      R4
8913 026476 001374          BNE      6$          ;HAVE WE MOVED ALL TABLE ENTRIES?
8914 026500 000411          BR       5$          ;BRANCH IF NOT--ONE MORE TIME.
8915 026502 010122  3$:  MOV      R1,(R2)+        ;RETURN TO INPUT ROUTINES.
8916 026504 062701 000020          ADD      #20,R1         ;SET UP A TABLE ENTRY
8917 026510 022702 027740          CMP      #DHADTB+40,R2  ;GENERATE NEXT DH11 ADDR
8918 026514 001372          BNE      3$          ;END OF TABLE ?
8919 026516 000402          BR       5$          ;BR IF NOT
8920 026520 104401  4$:  TYPE     INMSG4        ;RETURN TO INPUT ROUTINES
8921 026522 035250          ;TELL HIM HE GOOFED
8922 026524 000207  5$:  RTS      PC           ;RETURN TO INPUT ROUTINES
8923
8924 026526 020127 000300  CHKVCT: CMP      R1,#300      ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
8925 026532 002001          BGE      1$          ;BR IF YES
8926 026534 000436          BR       4$          ;BR IF NOT
8927 026536 020127 001000  1$:  CMP      R1,#1000      ;IS IT BELOW THE HIGH LIMIT?
8928 026542 002401          BLT      2$          ;BR IF YES
8929 026544 000432          BR       4$          ;BR IF NOT
8930 026546 032701 000007  2$:  BIT      #7,R1          ;CORRECT BOUNDARY ?
8931 026552 001027          BNE      4$          ;BR IF NOT
8932 026554 062716 000002          ADD      #2,(SP)        ;MOVE RETURN PC AROUND ERROR BRANCH
8933 026560 012702 027740          MOV      #DHVCTB,R2     ;POINT TO BEGIN OF VECTOR TABLE
8934 026564 032777 000001 152346 BIT      #BIT0,@SWR      ;ARE WE AUTOSIZING?
8935 026572 001011          BNE      3$          ;BRANCH IF NOT.
8936 026574 012703 030104          MOV      #DHVEC,R3     ;POINT TO BEGINING OF AUTOSIZER
8937                                ;DH VECTOR TABLE.
8938 026600 013704 030314          MOV      $DHSEL,R4
8939 026604 012322  6$:  MOV      (R3)+,(R2)+     ;MOVE CONTENTS OF AUTOSIZER VECTOR
8940                                ;TABLE TO TABLE USED BY PROGRAM.
8941 026606 006204          ASR      R4
8942 026610 005704          TST      R4
8943 026612 001374          BNE      6$          ;HAVE WE MOVED ALL TABLE ENTRIES?
8944 026614 000410          BR       5$          ;BRANCH IF NOT--ONE MORE TIME.
8945 026616 010122  3$:  MOV      R1,(R2)+        ;RETURN TO INPUT ROUTINES.
                                ;SET UP A TABLE ENTRY
```

```

8946 026620 060001          ADD    R0,R1          ;GENERATE NEXT DH11 ADDR
8947 026622 022702 030000  CMP    #DHVCTB+40,R2 ;END OF TABLE ?
8948 026626 001373          BNE    3$             ;BR IF NOT
8949 026630 000402          BR     5$             ;RETURN TO INPUT ROUTINES
8950 026632 104401          4$:   TYPE           ;TELL HIM HE GOOFED
8951 026634 035321          INMSG5
8952 026636 000207          5$:   RTS    PC       ;RETURN TO INPUT ROUTINES
8953
8954          ;THESE TWO ROUTINES SERVICE UNEXPECTED BUS ERROR AND RSVD INSTR TRAPS
8955
8956 026640 012737 000340 001202 BUSER:  MOV    #340,$TMP0    ;SAVE THE PSW
8957 026646 010637 001176          MOV    SP,$REG6      ;SAVE THE SP
8958 026652 012601          MOV    (SP)+,R1      ;GET THE TRAP PC
8959 026654 012602          MOV    (SP)+,R2      ;GET THE TRAP PSW
8960 026656 113700 001102          MOV    $STNM,R0      ;GET TEST NO.
8961 026662 012706 001100          MOV    #STACK,SP     ;RESET THE STACK POINTER
8962 026666 004737 024462          JSR    PC,SUER3       ;GO SET UP ERROR INFO
8963 026672 012737 026702 001110          MOV    #1$,$LPERR    ;ALWAYS COME BACK TO 1$
8964 026700 104027          ERROR 27            ;UNEXPECTED BUS ERROR TRAP
8965 026702 000005          1$:   RESET          ;PREPARE TO RESTART
8966 026704 004737 027150          JSR    PC,CHPS1      ;GO CLEAR PSW
8967 026710 000137 002734          JMP    REST1         ;GO RESTART THE PROGRAM
8968
8969 026714 012737 000340 001202 RESERR: MOV    #340,$TMP0    ;SAVE THE PSW
8970 026722 010637 001176          MOV    SP,$REG6      ;SAVE THE SP
8971 026726 012601          MOV    (SP)+,R1      ;GET THE TRAP PC
8972 026730 012602          MOV    (SP)+,R2      ;GET THE TRAP PSW
8973 026732 113700 001102          MOV    $STNM,R0      ;GET TEST NO.
8974 026736 012706 001100          MOV    #STACK,SP     ;RESET THE STACK POINTER
8975 026742 004737 024462          JSR    PC,SUER3       ;GO SET UP ERROR INFO
8976 026746 012737 026756 001110          MOV    #1$,$LPERR    ;ALWAYS COME BACK TO 1$
8977 026754 104030          ERROR 30            ;UNEXPECTED RSVD INSTR ERROR TRAP
8978 026756 000005          1$:   RESET          ;PREPARE TO RESTART
8979 026760 004737 027150          JSR    PC,CHPS1      ;GO CLEAR PSW
8980 026764 000137 002734          JMP    REST1         ;GO RESTART THE PROGRAM
8981
8982          ;THIS ROUTINE IS CALLED WHEN A TEST NEEDS TO RESTORE THE TRAP
8983          ;CATCHER IN THE DH11 VECTOR
8984
8985 026770 013703 027304          RESTRP: MOV    DHVCT,R3      ;GET VECTOR ADDRESS
8986 026774 010313          MOV    R3,(R3)       ;RESTORE THE TRAP CATCHER
8987 026776 062723 000002          ADD    #2,(R3)+
8988 027002 005023          CLR    (R3)+
8989 027004 010313          MOV    R3,(R3)
8990 027006 062723 000002          ADD    #2,(R3)+
8991 027012 005023          CLR    (R3)+
8992 027014 000207          RTS    PC             ;RETURN TO CALLING TEST
8993
8994          ;THIS ROUTINE CALLED BY ANY TEST THAT NEEDS A TIMING WAIT LOOP
8995          ;"TIMEA" IS INITIALIZED BY THE CALLING ROUTINE TO THE MINIMUM REQUIRED
8996          ;VALUE AND "TIMEB" IS CLEARED TO 000000. IF A TIME OUT OCCURS THIS
8997          ;ROUTINE WILL MOVE THE RETURN PC AROUND THE "LOOP" BRANCH BACK IN
8998          ;THE ROUTINE THAT CALLED IT TO ALLOW REPORTING AN ERROR MESSAGE
8999
9000 027016 005237 030352          TIMEIT: INC    TIMEB      ;COUNT B
9001 027022 001005          BNE    1$            ;BR IF NOT ZERO

```

```

9002 027024 005337 030350          DEC    TIMEA      ;COUNT TIME A
9003 027030 001002                   BNE    1$         ;BR IF NO TIMEOUT
9004 027032 062716 000002          ADD    #2,(SP)    ;MOVE RETURN PC TO ALLOW ERROR REPORT
9005 027036 000207          1$:    RTS    PC      ;RETURN TO THE CALLING TEST
9006
9007                                ;THIS ROUTINE CALLED BY THE AUTO ECHO TEST TO SET UP FOR TRANSFERRING
9008                                ;A BINARY COUNT TEST PATTERN ON ALL LINES
9009
9010 027040 012737 000020 001216  SETALL: MOV    #20,$TMP6      ;SET UP SIXTEEN LINES
9011 027046 005002                   CLR    R2         ;INIT A TABLE INDEX REG
9012 027050 012703 000200          MOV    #200,R3    ;SET UP TO GENERATE HI BYTE OF EXPECTED DATA
9013 027054 012704 000001          MOV    #1,R4      ;SET UP INDEX REG TO ODD BYTES
9014 027060 005011                   CLR    (R1)       ;START WITH LINE 00
9015 027062 012761 037312 000006  1$:    MOV    #TBUF,CAR(R1) ;SET UP BUS ADDR REG
9016 027070 012761 177400 000010  MOV    #-400,BCR(R1) ;SET UP BYTE COUNT REG
9017 027076 012761 031403 000004  MOV    #31403,LPR(R1) ;SET UP FOR 4800 BAUD/8 BIT CHARS
9018 027104 105062 036312          CLR    RBUF(R2)   ;START WITH DATA CHAR OF 000
9019 027110 110364 036312          MOV    R3,RBUF(R4) ;SET UP HIGH BYTE OF EXPECTED DATA
9020 027114 005211                   INC    (R1)       ;GEN NEW LINE NO. IN SCR
9021 027116 005203                   INC    R3         ;UPDATE THE POINTERS AND DATA
9022 027120 062702 000002          ADD    #2,R2
9023 027124 062704 000002          ADD    #2,R4
9024 027130 005337 001216          DEC    $TMP6      ;COUNT ONE LINE DONE
9025 027134 001352                   BNE    1$         ;BR TIL ALL 16 SET UP
9026 027136 013737 027312 027560  MOV    LINSEL,LINACT ;SET SOFTWARE FLAG FOR ALL LINES ACTIVE
9027 027144 005011                   CLR    (R1)       ;PUT SCR REG BACK TO LINE 00
9028 027146 000207          RTS    PC      ;RETURN TO AUTO ECHO TEST
9029
9030
9031                                ;THIS ROUTINE IS CALLED TO SET PSW PRIORITY TO 000 IN ORDER
9032                                ;TO BE LSI11 COMPATIBLE
9033
9034 027150 012746 000000          CHPS1: MOV    #0,-(SP)      ;NEW PSW
9035 027154 012746 027162          MOV    #1$,-(SP)     ;NEW PC
9036 027160 000002                   RTI                     ;CHANGE PSW
9037 027162 000207          1$:    RTS    PC      ;RETURN TO CALLING TEST
9038
9039                                ;THIS ROUTINE DOES THE SAME THING EXCEPT IT SET THE PSW
9040                                ;PRIORITY TO 340 (LEVEL 7 ) TO LOCK OUT INTRs
9041
9042 027164 012746 000340          CHPS2: MOV    #340,-(SP)   ;NEW PSW
9043 027170 012746 027176          MOV    #1$,-(SP)     ;NEW PC
9044 027174 000002                   RTI                     ;CHANGE THE PSW
9045 027176 000207          1$:    RTS    PC      ;RETURN TO CALLING TEST
9046
9047                                ;THIS ROUTINE IS ALSO FOR LSI11 COMPATIBILITY AND IT IS CALLED
9048                                ;TO SAVE THE PSW IN "$TMP0"
9049
9050 027200 005046          SAPS:  CLR    -(SP)      ;TEMP STORAGE TO SAVE PSW
9051 027202 013746 000034          MOV    34,-(SP)     ;SAVE TRAP VECTOR POINTER
9052 027206 012737 027216 000034  MOV    #1$,34      ;GO TO 1$ ON TRAP
9053 027214 104400          TRAP                    ;GO TO IT
9054 027216 016666 000002 000006  1$:    MOV    2(SP),6(SP)   ;GET PSW SAVED
9055 027224 012716 027232          MOV    #2$,(SP)    ;GO TO 2$ ON RTI
9056 027230 000002          RTI
9057 027232 012637 000034          2$:    MOV    (SP)+,34  ;RESTORE VECTOR

```

```

9058 027236 012637 001202          MOV    (SP)+,$TMP0    ;FINALLY SAVE PSW IN $TMP0
9059 027242 000207                   RTS    PC
9060
9061
9062          ;SUBROUTINE TO SIZE FOR AN 11/70 CENTRAL PROCESSOR      ;:++D
9063          ; IF IT IS AN 11/70 CPU, CACHE WILL BE DISABLED
9064          ; IF NOT AN 11/70 CPU, NO ACTION TAKEN
9065
9066          ;CALLED BY          JSR    PC,DCACHE
9067          ;                   NO ARGUEMENTS PASSED
9068
9069
9070 027244 013746 000004          DCACHE: MOV    @#4,-(SP)    ;SAVE TRAP INFO
9071 027250 012737 027272 000004          MOV    #1$,@#4        ;SETUP FOR TIMEOUT
9072 027256 005737 177746          TST    @#177746       ;TEST FOR CACHE
9073 027262 012737 000014 177746          MOV    #14,@#177746   ;DISABLE CACHE
9074 027270 000401                   BR     2$              ;EXIT, CACHE DISABLED
9075 027272 022626          1$:  CMP    (SP)+,(SP)+    ;CLEAN UP STACK
9076 027274 012637 000004          2$:  MOV    (SP)+,@#4
9077 027300 000207                   RTS    PC              ;RETURN
9078
9079
9080          ;:*****
9081          ;:ADDITIONAL PROGRAM CONSTANTS AND VARIABLES
9082          ;:*****
9083          000002          NRC=2          ;INDEX CONST. TO ACCESS NEXT RCVD CHAR REG
9084          000004          LPR=4          ;INDEX CONST. TO ACCESS LINE PARAMETER REG.
9085          000006          CAR=6          ;INDEX CONST. TO ACCESS CURRENT ADDRESS REG.
9086          000010          BCR=10         ;INDEX CONST. TO ACCESS BYTE COUNT REG.
9087          000012          BAR=12         ;INDEX CONST. TO ACCESS BUFFER ACTIVE REG.
9088          000014          BKR=14         ;INDEX CONST. TO ACCESS BREAK CONTROL REG.
9089          000016          SSR=16         ;INDEX CONST. TO ACCESS SILO STATUS REG.
9090
9091 027302 000000          DHADR: 0          ;HOLDS THE "SCR" ADDRESS OF THE DH11 UNDER TEST
9092 027304 000000          DHVCT: 0          ;HOLDS THE 1ST VECTOR ADDRESS OF THE DH11 UNDER TEST
9093 027306 000000          SELMSK: 0        ;BIT TST MARKER FOR SELECTING DH11'S
9094 027310 000003          DHSSEL: 3         ;SPECIFIES DH11'S SELECTED FOR TEST
9095 027312 177777          LINSEL: 177777    ;SPECIFIES LINES TO TEST
9096 027314 000000          LINMSK: 0         ;MARKER USED TO TEST FOR LINES TO TEST
9097 027316 000000          LMSK1: 0          ;ALTERNATE MARKER TO SUPPORT THE
9098          ;SELECT LINES FEATURE
9099 027320 000004          MSTCLR: .BLKW 4    ;FOUR WORD ADDRESS TABLE USED BY THE TEST THAT
9100          ;CHECKS OPERATION OF "MASTER CLR"
9101
9102 027330 177777          PATRNA: 177777    ;BIT PATTERNS USED WITH "CAR" AND "BCR" TESTS
9103 027332 125252          125252
9104 027334 052525          052525
9105 027336 000000          000000          ;TABLE TERMINATOR
9106
9107 027340 000060          PATRNB: 60          ;BIT PATTERNS USED IN "CAR" MEM EXT BIT TEST
9108 027342 000300          300
9109 027344 000020          20
9110 027346 000100          100
9111 027350 000040          40
9112 027352 000200          200
9113 027354 000000          0          ;TABLE TERMINATOR

```

```

9114 027356 000000          0          ;TABLE TERMINATOR
9115
9116          ;THIS TABLE STORES THE BYTE COUNT AND LINE PARAMETERS FOR THE
9117          ;8 SUBTESTS IN THE MULTILINE PARITY/DATA TEST
9118
9119 027360 177400          PRTYTB: -400          ;256 CHARS
9120 027362 027363          27363          ;2400 BAUD - ODD PARITY - 8 BITS
9121 027364 177400          -400          ;256 CHARS
9122 027366 027323          27323          ;2400 BAUD - EVEN PARITY - 8 BITS
9123 027370 177600          -200          ;128 CHARS
9124 027372 027362          27362          ;2400 BAUD - ODD PARITY - 7 BITS
9125 027374 177600          -200          ;128 CHARS
9126 027376 027322          27322          ;2400 BAUD - EVEN PARITY - 7 BITS
9127 027400 177700          -100          ;64 CHARS
9128 027402 027361          27361          ;2400 BAUD - ODD PARITY - 6 BITS
9129 027404 177700          -100          ;64 CHARS
9130 027406 027321          27321          ;2400 BAUD - EVEN PARITY - 6 BITS
9131 027410 177740          -40          ;32 CHARS
9132 027412 027360          27360          ;2400 BAUD - ODD PARITY - 5 BITS
9133 027414 177740          -40          ;32 CHARS
9134 027416 027320          27320          ;2400 BAUD - EVEN PARITY - 5 BITS
9135
9136          ;THIS 16 WORD TABLE CONTAINS THE TEST DATA USED BY THE AUTO ECHO
9137          ;TEST (ALL 1'S DATA TABLE)
9138
9139 027420 100377          AETAB: 100377          ;TEST DATA FOR LINE 00
9140 027422 100777          100777          ;TEST DATA FOR LINE 01
9141 027424 101377          101377
9142 027426 101777          101777
9143 027430 102377          102377
9144 027432 102777          102777
9145 027434 103377          103377
9146 027436 103777          103777
9147 027440 104377          104377
9148 027442 104777          104777
9149 027444 105377          105377
9150 027446 105777          105777
9151 027450 106377          106377
9152 027452 106777          106777
9153 027454 107377          107377
9154 027456 107777          107777          ;TEST DATA FOR LINE 17
9155
9156          ;THIS 16 WORD TABLE CONTAINS THE TEST DATA USED BY THE AUTO ECHO
9157          ;TEST (ALL 0'S DATA TABLE)
9158
9159 027460 100000          AETAB0: 100000          ;TEST DATA FOR LINE 00
9160 027462 100400          100400          ;TEST DATA FOR LINE 01
9161 027464 101000          101000
9162 027466 101400          101400
9163 027470 102000          102000
9164 027472 102400          102400
9165 027474 103000          103000
9166 027476 103400          103400
9167 027500 104000          104000
9168 027502 104400          104400
9169 027504 105000          105000

```

```
9170 027506 105400          105400
9171 027510 106000          106000
9172 027512 106400          106400
9173 027514 107000          107000
9174 027516 107400          107400          ;TEST DATA FOR LINE 17
9175
9176          ;THIS TABLE USED BY THE AUTO ECHO TEST 2 TO RESET ACTIVE BIT WHEN A
9177          ;LINE IS DONE
9178
9179 027520 000001          LINBIT: BIT00          ;DEACTIVATE LINE 00
9180 027522 000002          BIT01          ;DEACTIVATE LINE 01
9181 027524 000004          BIT02
9182 027526 000010          BIT03
9183 027530 000020          BIT04
9184 027532 000040          BIT05
9185 027534 000100          BIT06
9186 027536 000200          BIT07
9187 027540 000400          BIT08
9188 027542 001000          BIT09
9189 027544 002000          BIT10
9190 027546 004000          BIT11
9191 027550 010000          BIT12
9192 027552 020000          BIT13
9193 027554 040000          BIT14
9194 027556 100000          BIT15          ;DEACTIVATE LINE 17
9195
9196 027560 000000          LINACT: 0          ;MAINTAINS STATUS OF ACTIVE LINES
9197          ;DURING AUTO ECHO TEST 2
9198
9199
9200          ;THIS TABLE CONTAINS 16. COUNTERS USED BYN THE MULTI-LINE
9201          ;PARITY TEST TO KEEP TRACK OF TOTAL CHARS RECEIVED
9202
9203 027562 000020          MULPTB: .BLKW 16.          ;SIXTEEN WORD COUNTERS TABLE
9204
9205          ;THIS 16 WORD TABLE CONTAINS THE TEST DATA USED BY THE BREAK BIT
9206          ;TEST
9207
9208 027622 120000          BRKTAB: 120000          ;TEST DATA FOR LINE 00
9209 027624 120400          120400          ;TEST DATA FOR LINE 01
9210 027626 121000          121000
9211 027630 121400          121400
9212 027632 122000          122000
9213 027634 122400          122400
9214 027636 123000          123000
9215 027640 123400          123400
9216 027642 124000          124000
9217 027644 124400          124400
9218 027646 125000          125000
9219 027650 125400          125400
9220 027652 126000          126000
9221 027654 126400          126400
9222 027656 127000          127000
9223 027660 127400          127400          ;TEST DATA FOR LINE 17
9224
9225 027662 131177          RGMSK1: 131177          ;MASK TO SPECIFY R/W BITS FOR NORMAL "SCR" REG TEST
```



```

9226 027664 046600      RGMSK2: 46600          ;MASK TO SPECIFY READ ONLY BITS IN "SCR" FOR NORMAL MODE TEST
9227 027666 177767      RGMSK3: 177767        ;MASK TO SPECIFY R/W BITS IN "LPR"
9228 027670 177777      RGMSK4: 177777        ;MASK TO SPECIFY R/W BITS IN "BKR"
9229 027672 100077      RGMSK5: 100077        ;MASK TO SPECIFY R/W BITS IN "SSR"
9230 027674 042200      RGMSK6: 42200         ;MASK TO SPECIFY READ ONLY BITS IN "SCR" FOR MAINT. MODE TEST
9231 027676 030100      INTMSK: 30100         ;MASK USED TO SELECT INTR BITS TO TEST
9232
9233                      ;DH11 ADDRESS TABLE - THIS TABLE CONTAINS THE "SCR" ADDRESS FOR UP TO
9234                      ;SIXTEEN DH11'S
9235
9236 027700 160020      DHADTB: 160020        ;ADDRESS OF FIRST DH11
9237 027702 160040      160040               ;ADDRESS OF SECOND DH11
9238 027704 160060      160060
9239 027706 160100      160100
9240 027710 160120      160120
9241 027712 160140      160140
9242 027714 160160      160160
9243 027716 160200      160200
9244 027720 160220      160220
9245 027722 160240      160240
9246 027724 160260      160260
9247 027726 160300      160300
9248 027730 160320      160320
9249 027732 160340      160340
9250 027734 160360      160360
9251 027736 160400      160400               ;ADDRESS OF THE LAST DH11
9252
9253                      ;DH11 VECTOR TABLE - THIS TABLE CONTAINS THE VECTOR ADDRESSES FOR UP
9254                      ;TO SIXTEEN DH11'S
9255
9256 027740 000330      DHVCTB: 330           ;ADDRESS OF VECTOR FOR FIRST DH11
9257 027742 000350      350                  ;ADDRESS OF VECTOR FOR SECOND DH11
9258 027744 000370      370
9259 027746 000410      410
9260 027750 000430      430
9261 027752 000450      450
9262 027754 000470      470
9263 027756 000510      510
9264 027760 000530      530
9265 027762 000550      550
9266 027764 000570      570
9267 027766 000610      610
9268 027770 000630      630
9269 027772 000650      650
9270 027774 000670      670
9271 027776 000710      710               ;ADDRESS OF VECTOR FOR LAST DH11
9272
9273 030000 000000      VCFLG: 0              ;VECTOR SET UP FLAGG
9274
9275
9276                      ;BR PRIORITY LEVEL TABLE - THIS TABLE CONTAINS THE PRIORITY LEVELS
9277                      ;FOR UP TO SIXTEEN DH11'S - THE RCVR LEVEL IS STORED IN THE LOW BYTE
9278                      ;AND THE XMTR LEVEL IN THE HIGH BYTE
9279
9280 030002 120240      BRLVL: 120240         ;BRLEVELS FOR FIRST DH11
9281 030004 120240      120240               ;BR LEVELS FOR SECOND DH11

```

9282	030006	120240	120240
9283	030010	120240	120240
9284	030012	120240	120240
9285	030014	120240	120240
9286	030016	120240	120240
9287	030020	120240	120240
9288	030022	120240	120240
9289	030024	120240	120240
9290	030026	120240	120240
9291	030030	120240	120240
9292	030032	120240	120240
9293	030034	120240	120240
9294	030036	120240	120240
9295	030040	120240	120240

;BR LEVELS FOR LAST DH11

;THIS DH ADDRESS TABLE IS FILLED BY THE AUTOSIZER.

9299	030042		DHADRS:
9300	030042	000000	.WORD 0
9301	030044	000000	.WORD 0
9302	030046	000000	.WORD 0
9303	030050	000000	.WORD 0
9304	030052	000000	.WORD 0
9305	030054	000000	.WORD 0
9306	030056	000000	.WORD 0
9307	030060	000000	.WORD 0
9308	030062	000000	.WORD 0
9309	030064	000000	.WORD 0
9310	030066	000000	.WORD 0
9311	030070	000000	.WORD 0
9312	030072	000000	.WORD 0
9313	030074	000000	.WORD 0
9314	030076	000000	.WORD 0
9315	030100	000000	.WORD 0
9316	030102	000000	.WORD 0

;THIS DH VECTOR TABLE IS FILLED BY THE AUTOSIZER.

9317			
9318			
9319			
9320	030104		DHVEC:
9321	030104	000000	.WORD 0
9322	030106	000000	.WORD 0
9323	030110	000000	.WORD 0
9324	030112	000000	.WORD 0
9325	030114	000000	.WORD 0
9326	030116	000000	.WORD 0
9327	030120	000000	.WORD 0
9328	030122	000000	.WORD 0
9329	030124	000000	.WORD 0
9330	030126	000000	.WORD 0
9331	030130	000000	.WORD 0
9332	030132	000000	.WORD 0
9333	030134	000000	.WORD 0
9334	030136	000000	.WORD 0
9335	030140	000000	.WORD 0
9336	030142	000000	.WORD 0
9337			

9338 ;THIS DM ADDRESS TABLE IS FILLED BY THE AUTOSIZER.

9339
9340 030144 DMADRS:
9341 030144 000000 .WORD 0
9342 030146 000000 .WORD 0
9343 030150 000000 .WORD 0
9344 030152 000000 .WORD 0
9345 030154 000000 .WORD 0
9346 030156 000000 .WORD 0
9347 030160 000000 .WORD 0
9348 030162 000000 .WORD 0
9349 030164 000000 .WORD 0
9350 030166 000000 .WORD 0
9351 030170 000000 .WORD 0
9352 030172 000000 .WORD 0
9353 030174 000000 .WORD 0
9354 030176 000000 .WORD 0
9355 030200 000000 .WORD 0
9356 030202 000000 .WORD 0
9357 030204 000000 .WORD 0

9358
9359 ;THIS DM VECTOR TABLE IS FILLED BY THE AUTOSIZER.

9360
9361 030206 DMVEC:
9362 030206 000000 .WORD 0
9363 030210 000000 .WORD 0
9364 030212 000000 .WORD 0
9365 030214 000000 .WORD 0
9366 030216 000000 .WORD 0
9367 030220 000000 .WORD 0
9368 030222 000000 .WORD 0
9369 030224 000000 .WORD 0
9370 030226 000000 .WORD 0
9371 030230 000000 .WORD 0
9372 030232 000000 .WORD 0
9373 030234 000000 .WORD 0
9374 030236 000000 .WORD 0
9375 030240 000000 .WORD 0
9376 030242 000000 .WORD 0
9377 030244 000000 .WORD 0

9378
9379
9380 030246 \$LNSEL:
9381 030246 000001 .WORD 1
9382 030250 000002 .WORD 2
9383 030252 000004 .WORD 4
9384 030254 000010 .WORD 10
9385 030256 000020 .WORD 20
9386 030260 000040 .WORD 40
9387 030262 000100 .WORD 100
9388 030264 000200 .WORD 200
9389 030266 000400 .WORD 400
9390 030270 001000 .WORD 1000
9391 030272 002000 .WORD 2000
9392 030274 004000 .WORD 4000
9393 030276 010000 .WORD 10000

```
9394 030300 020000      .WORD 20000
9395 030302 040000      .WORD 40000
9396 030304 100000      .WORD 100000
9397
9398 030306 000000      ADRVEC: 0           ;ADDRESSES BETWEEN VECTORS - FILLED BY THE AUTOSIZER
9399 030310 000000      DHMCSR: 0          ;MODEM CONTROL CONTROL AND STATUS REGISTER.
9400 030312 000000      DHMLSR: 0         ;MODEM CONTROL LINE STATUS REGISTER.
9401 030314 000000      $DHSEL: 0         ;DEVICE SELECT PARAMETER - FILLED BY THE AUTOSIZER.
9402 030316      000      DHRLVL: .BYTE 0    ;BR LEVEL FOR RCVR
9403 030317      000      DHTLVL: .BYTE 0    ;BR LEVEL FOR XMITTER
9404 030320 000000      DHNUM: 0          ;CONTAINS NUMBER OF THE DH11 UNDER TEST
9405 030322 000000      LINE: 0           ;CONTINES NUMBER OF THE LINE UNDER TEST
9406 030324 000000      LINEA: 0          ;LOCATION TO SAVE LINE NUMBER
9407
9408      ;ADDRESS POINTERS TO SET UP TABLES WHEN INPUTTING PARAMETERS
9409 030326 000000      ADPTR: 0          ;POINTS TO ADDRESS TABLE
9410 030330 000000      VCPTR: 0          ;POINTS TO VECTOR TABLE
9411 030332 000000      BRPTR: 0         ;POINTS TO BR LEVEL TABLE
9412
9413      ;THE FOLLOWING TEN CONSTANTS ARE USED BY THE MODEM CONTROL PORTION OF THIS PROGRAM.
9414
9415      000400      STEP=400
9416      000001      LINENA=1
9417      000002      TRMRDY=2
9418      000004      RS=4
9419      000010      SECTX=10
9420      000020      SECRIX=20
9421      000040      CS=40
9422      000100      CO=100
9423      000200      RING=200
9424      002000      CLRMUX=2000
9425
9426 030334 000000      TDATA1: 0         ;DATA BUFFER FOR BASIC DATA TEST
9427 030336 000037      TDATA2: 37        ;TEST DATA FOR FIVE BIT CHAR
9428 030340 000077      77              ;TEST DATA FOR SIX BIT CHAR
9429 030342 000177      177            ;TEST DATA FOR SEVEN BIT CHAR
9430 030344 000377      377            ;TEST DATA FOR EIGHT BIT CHAR
9431 030346 000000      TITFLG: 0        ;FLAG TO ALLOW PRINTING TITLE ONLY ONCE
9432 030350 000000      TIMEA: 0         ;GENERAL PURPOSE TIMERS
9433 030352 000000      TIMEB: 0
9434 030354 000000      TIMEC: 0         ;TIMER FOR TIMING TESTS
9435 030356 000000      TNULL: 0        ;CONTAINS TWO NULL CHARS USED BY BREAK TEST
9436
9437
9438
```

9439
9440
9441
9442
9443
9444
9445
9446
9447
9448
9449
9450
9451
9452
9453
9454
9455
9456
9457
9458
9459
9460
9461
9462
9463
9464
9465
9466
9467
9468
9469
9470
9471
9472
9473
9474
9475
9476
9477
9478
9479
9480
9481
9482
9483
9484
9485
9486
9487
9488
9489
9490
9491
9492
9493
9494

030360 044104 030461 051040
030366 043505 051511 042524
030374 020122 042522 042506
030402 042522 041516 020105
030410 040503 051525 042105
030416 052040 046511 047505
030424 052125 000
030427 040 050050 024503
030434 020040 020040 050050
030442 024523 020040 020040
030450 051450 024520 020040
030456 020040 042524 052123
030464 020040 042040 053105
030472 042101 020122 051040
030500 043505 042101 000122

030506 001116 001202 001176
030514 001162 001164 001166
030522 000000

030524 054523 052123 046505
030532 041440 047117 051124
030540 046117 051040 043505
030546 051511 042524 020122
030554 051105 047522 000122
030562 024040 041520 020051
030570 020040 024040 051520
030576 020051 020040 024040
030604 050123 020051 020040
030612 052040 051505 020124
030620 020040 042504 040526
030626 051104 020040 042522
030634 040507 051104 020040
030642 053440 051501 020040
030650 020040 051440 041057
030656 000
030660
030660 001116 001202 001176
030666 001162 001164 001166
030674 001170 001172 000000

030702 044104 030461 046440
030710 051501 042524 020122
030716 046103 040505 020122
030724 040506 046111 042105
030732 052040 020117 046103

: ERROR MESSAGE INFORMATION - MESSAGE BUFFERS AND POINTERS
: *****

: INFORMATION FOR MESSAGE 1

EM1: .ASCIZ 'DH11 REGISTER REFERENCE CAUSED TIMEOUT'

DH1: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR REGADR'

.EVEN
DT1: .WORD \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,0

: INFORMATION FOR MESSAGE 2

EM2: .ASCIZ 'SYSTEM CONTROL REGISTER ERROR'

DH2: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B'

.EVEN
DT2: .WORD \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0

: INFORMATION FOR MESSAGE 3

EM3: .ASCIZ 'DH11 MASTER CLEAR FAILED TO CLR SPECIFIED REG'

9495	030740	020122	050123	041505	
9496	030746	043111	042511	020104	
9497	030754	042522	000107		
9498					
9499					; INFORMATION FOR MESSAGE 4
9500					
9501	030760	044514	042516	050040	EM4: .ASCIZ 'LINE PARAMETER REGISTER ERROR'
9502	030766	051101	046501	052105	
9503	030774	051105	051040	043505	
9504	031002	051511	042524	020122	
9505	031010	051105	047522	000122	
9506					
9507					; INFORMATION FOR MESSAGE 5
9508					
9509	031016	051102	040505	020113	EM5: .ASCIZ 'BREAK CONTROL REGISTER ERROR'
9510	031024	047503	052116	047522	
9511	031032	020114	042522	044507	
9512	031040	052123	051105	042440	
9513	031046	051122	051117	000	
9514					
9515					; INFORMATION FOR MESSAGE 6
9516					
9517	031053	123	046111	020117	EM6: .ASCIZ 'SILO STATUS REGISTER ERROR'
9518	031060	052123	052101	051525	
9519	031066	051040	043505	051511	
9520	031074	042524	020122	051105	
9521	031102	047522	000122		
9522					
9523					; INFORMATION FOR MESSAGE 7
9524					
9525	031106	052503	051122	047105	EM7: .ASCIZ 'CURRENT ADDRESS REGISTER ERROR - LINE #XX'
9526	031114	020124	042101	051104	
9527	031122	051505	020123	042522	
9528	031130	044507	052123	051105	
9529	031136	042440	051122	051117	
9530	031144	026440	046040	047111	
9531	031152	020105	054043	000130	
9532					
9533					; INFORMATION FOR MESSAGE 10
9534					
9535	031160	054502	042524	041440	EM10: .ASCIZ 'BYTE COUNTER REGISTER ERROR - LINE #XX'
9536	031166	052517	052116	051105	
9537	031174	051040	043505	051511	
9538	031202	042524	020122	051105	
9539	031210	047522	020122	020055	
9540	031216	044514	042516	021440	
9541	031224	054130	000		
9542					
9543					; INFORMATION FOR MESSAGE 11
9544					
9545	031227	125	042516	050130	EM11: .ASCIZ 'UNEXPECTED DH11 RCVR INTERRUPT'
9546	031234	041505	042524	020104	
9547	031242	044104	030461	051040	
9548	031250	053103	020122	047111	
9549	031256	042524	051122	050125	
9550	031264	000124			

```
9551
9552           ; INFORMATION FOR MESSAGE 12
9553
9554 031266 047125 054105 042520 EM12: .ASCIZ 'UNEXPECTED DH11 XMITTR INTERRUPT'
9555 031274 052103 042105 042040
9556 031302 030510 020061 046530
9557 031310 052111 051124 044440
9558 031316 052116 051105 052522
9559 031324 052120      000
9560
9561           ; INFORMATION FOR MESSAGE 13
9562
9563 031327      103 040510 020122 EM13: .ASCIZ 'CHAR AVAILABLE FAILED TO GENERATE RCVR INTERRUPT'
9564 031334 053101 044501 040514
9565 031342 046102 020105 040506
9566 031350 046111 042105 052040
9567 031356 020117 042507 042516
9568 031364 040522 042524 051040
9569 031372 053103 020122 047111
9570 031400 042524 051122 050125
9571 031406 000124
9572
9573           ; INFORMATION FOR MESSAGE 14
9574
9575 031410 051124 047101 046523 EM14: .ASCIZ 'TRANSMITTER NPR LOGIC ERROR - LINE # '
9576 031416 052111 042524 020122
9577 031424 050116 020122 047514
9578 031432 044507 020103 051105
9579 031440 047522 020122 020055
9580 031446 044514 042516 021440
9581 031454 020040      000
9582
9583           ; INFORMATION FOR MESSAGE 15
9584
9585 031457      130 044515 052124 EM15: .ASCIZ 'XMITTR FAILED TO INTERRUPT - LINE # '
9586 031464 020122 040506 046111
9587 031472 042105 052040 020117
9588 031500 047111 042524 051122
9589 031506 050125 020124 020055
9590 031514 044514 042516 021440
9591 031522 020040      000
9592
9593           ; INFORMATION FOR MESSAGE 16
9594
9595 031525      122 053103 020122 EM16: .ASCIZ 'RCVR FAILED TO INTERRUPT'
9596 031532 040506 046111 042105
9597 031540 052040 020117 047111
9598 031546 042524 051122 050125
9599 031554 000124      .
9600
9601           ; INFORMATION FOR MESSAGE 17
9602
9603 031556 051124 047101 046523 EM17: .ASCIZ 'TRANSMITTER TIMING ERROR - LINE # '
9604 031564 052111 042524 020122
9605 031572 044524 044515 043516
9606 031600 042440 051122 051117
```

CZDMM-D-0
CZDHMD.P11

MACY11 30A(1052)
09-MAR-78 15:32

10-MAR-78 08:05 PAGE 217
POWER DOWN AND UP ROUTINES

SEQ 0215
SEQ 0214

9607	031606	026440	046040	047111
9608	031614	020105	020043	000040
9609	031622	024040	041520	020051
9610	031630	020040	024040	051520
9611	031636	020051	020040	024040
9612	031644	050123	020051	020040
9613	031652	052040	051505	020124
9614	031660	020040	042504	040526
9615	031666	051104	020040	050123
9616	031674	042505	020104	020040
9617	031702	044524	042515	020102
9618	031710	020040	044524	042515
9619	031716	000103		

DH6: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR SPEED TIMEB TIMEC'

9620				
9621				
9622				
9623	031720	042522	042503	053111
9624	031726	051105	052040	046511
9625	031734	047111	020107	051105
9626	031742	047522	020122	020055
9627	031750	044514	042516	021440
9628	031756	020040	000	

;INFORMATION FOR MESSAGE 20

EM20: .ASCIZ 'RECEIVER TIMING ERROR - LINE # '

9629				
9630				
9631				
9632	031761	122	053103	020122
9633	031766	040506	046111	042105
9634	031774	052040	020117	047111
9635	032002	042524	051122	050125
9636	032010	020124	020055	044514
9637	032016	042516	021440	020040
9638	032024	000		

;INFORMATION FOR MESSAGE 21

EM21: .ASCIZ 'RCVR FAILED TO INTERRUPT - LINE # '

9639				
9640				
9641				
9642	032025	103	040510	020122
9643	032032	053101	044501	020114
9644	032040	040506	046111	042105
9645	032046	052040	020117	042523
9646	032054	020124	047117	052040
9647	032062	046511	020105	020055
9648	032070	044514	042516	021440
9649	032076	020040	000	

;INFORMATION FOR MESSAGE 22

EM22: .ASCIZ 'CHAR AVAIL FAILED TO SET ON TIME - LINE # '

9650				
9651				
9652				
9653	032101	102	051501	041511
9654	032106	042040	052101	020101
9655	032114	042524	052123	042440
9656	032122	051122	051117	026440
9657	032130	046040	047111	020105
9658	032136	020043	000040	
9659	032142	024040	041520	020051
9660	032150	020040	024040	051520
9661	032156	020051	020040	024040
9662	032164	050123	020051	020040

;INFORMATION FOR MESSAGE 23

EM23: .ASCIZ 'BASIC DATA TEST ERROR - LINE # '

DH7: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR CHRLNG WAS S/B'

9663 032172 052040 051505 020124
9664 032200 020040 042504 040526
9665 032206 051104 020040 044103
9666 032214 046122 043516 020040
9667 032222 053440 051501 020040
9668 032230 020040 051440 041057
9669 032236 000

9670
9671 ; INFORMATION FOR MESSAGE 24

9672
9673 032237 101 052125 020117 EM24: .ASCIZ 'AUTO ECHO TEST ERROR - LINE # '
9674 032244 041505 047510 052040
9675 032252 051505 020124 051105
9676 032260 047522 020122 020055
9677 032266 044514 042516 021440
9678 032274 020040 000

9679
9680 ; INFORMATION FOR MESSAGE 25

9681
9682 032277 102 042522 045501 EM25: .ASCIZ 'BREAK BIT TEST ERROR - LINE # '
9683 032304 041040 052111 052040
9684 032312 051505 020124 051105
9685 032320 047522 020122 020055
9686 032326 044514 042516 021440
9687 032334 020040 000

9688
9689 ; INFORMATION FOR MESSAGE 26

9690
9691 032337 110 046101 026506 EM26: .ASCIZ 'HALF-DUPLEX TEST ERROR - LINE # '
9692 032344 052504 046120 054105
9693 032352 052040 051505 020124
9694 032360 051105 047522 020122
9695 032366 020055 044514 042516
9696 032374 021440 020040 000

9697
9698 ; INFORMATION FOR MESSAGE 27

9699
9700 032401 125 042516 050130 EM27: .ASCIZ 'UNEXPECTED BUS ERROR TRAP'
9701 032406 041505 042524 020104
9702 032414 052502 020123 051105
9703 032422 047522 020122 051124
9704 032430 050101 000

9705 032433 040 050050 024503 DH3: .ASCIZ '(PC) (PS) (SP) TEST TRPPC TRPPS '
9706 032440 020040 020040 050050
9707 032446 024523 020040 020040
9708 032454 051450 024520 020040
9709 032462 020040 042524 052123
9710 032470 020040 052040 050122
9711 032476 041520 020040 052040
9712 032504 050122 051520 000040

9713
9714 032512 001116 001202 001176 .EVEN
9715 032520 001162 001164 001166 DT3: .WORD \$ERRPC,\$TMP0,\$REG6,\$REG0,\$REG1,\$REG2,0
9716 032526 000000

9717
9718 ; INFORMATION FOR MESSAGE 30

```

9719
9720 032530 047125 054105 042520 EM30: .ASCIZ 'UNEXPECTED RSVD INSTR TRAP'
9721 032536 052103 042105 051040
9722 032544 053123 020104 047111
9723 032552 052123 020122 051124
9724 032560 050101 000
9725
9726 ;INFORMATION FOR MESSAGE 31
9727
9728 032563 101 052125 020117 EM31: .ASCIZ 'AUTO ECHO DATA COMPARE FRROR - LINE # '
9729 032570 041505 047510 042040
9730 032576 052101 020101 047503
9731 032604 050115 051101 020105
9732 032612 051105 047522 020122
9733 032620 020055 044514 042516
9734 032626 021440 020040 000
9735 032633 040 050050 024503 DH4: .ASCIZ ' (PC) (PS) (SP) TEST WASADR SBADR WAS S/B'
9736 032640 020040 020040 050050
9737 032646 024523 020040 020040
9738 032654 051450 024520 020040
9739 032662 020040 042524 052123
9740 032670 020040 053440 051501
9741 032676 042101 020122 051440
9742 032704 040502 051104 020040
9743 032712 020040 040527 020123
9744 032720 020040 020040 027523
9745 032726 000102
9746
9747 ;INFORMATION FOR MESSAGE 32
9748
9749 032730 052501 047524 042440 EM32: .ASCIZ 'AUTO ECHO TEST TIMEOUT - LINE # '
9750 032736 044103 020117 042524
9751 032744 052123 052040 046511
9752 032752 047505 052125 026440
9753 032760 046040 047111 020105
9754 032766 020043 000040
9755 032772 024040 041520 020051 DH5: .ASCIZ ' (PC) (LPRG) TEST'
9756 033000 020040 046050 051120
9757 033006 024507 020040 052040
9758 033014 051505 000124
9759 .EVEN
9760 033020 001116 001202 001206 DT4: .WORD $ERRPC,$TMP0,$TMP2,0
9761 033026 000000
9762
9763 ;INFORMATION FOR MESSAGE 33
9764
9765 033030 040520 044522 054524 EM33: .ASCIZ 'PARITY LOGIC TEST ERROR - LINE # '
9766 033036 046040 043517 041511
9767 033044 052040 051505 020124
9768 033052 051105 047522 020122
9769 033060 020055 044514 042516
9770 033066 021440 020040 000
9771
9772 ;INFORMATION FOR MESSAGE 34
9773
9774 033073 115 046125 044524 EM34: .ASCIZ 'MULTI-LINE PARITY DATA TEST ERROR - LINE # - SUBTEST # '

```

9775 033100 046055 047111 020105
9776 033106 040520 044522 054524
9777 033114 042040 052101 020101
9778 033122 042524 052123 042440
9779 033130 051122 051117 026440
9780 033136 046040 047111 020105
9781 033144 020043 020040 020055
9782 033152 052523 052102 051505
9783 033160 020124 020043 000040

9784

9785

9786

9787 033166 052515 052114 026511
9788 033174 044514 042516 050040
9789 033202 051101 052111 020131
9790 033210 040504 040524 052040
9791 033216 051505 020124 044524
9792 033224 042515 052517 000124
9793 033232 024040 041520 020051
9794 033240 020040 046050 051120
9795 033246 024507 020040 041501
9796 033254 046124 047111 000
9797 033262 001116 001202 001210
9798 033270 000000

; INFORMATION FOR MESSAGE 35

EM35: .ASCIZ 'MULTI-LINE PARITY DATA TEST TIMEOUT'

DH14: .ASCIZ '(PC) (LPRG) ACTLIN'

.EVEN

DT6: .WORD \$ERRPC,\$TMP0,\$TMP3,0

9800

9801

9802

9803 033272 044103 051101 040440
9804 033300 040526 046111 041101
9805 033306 042514 052040 046511
9806 033314 047505 052125 000

; INFORMATION FOR MESSAGE 36

EM36: .ASCIZ 'CHAR AVAILABLE TIMEOUT'

9807

9808

9809

9810 033321 104 052101 020101
9811 033326 047503 050115 051101
9812 033334 020105 051105 047522
9813 033342 020122 020055 044514
9814 033350 042516 021440 020040
9815 033356 000

; INFORMATION FOR MESSAGE 37

EM37: .ASCIZ 'DATA COMPARE ERROR - LINE # '

9816

9817

9818

9819 033357 102 043125 042506
9820 033364 020122 041501 044524
9821 033372 042526 051040 043505
9822 033400 042440 051122 051117
9823 033404 026440 046040 047111
9824 033414 020105 020043 000040

; INFORMATION FOR MESSAGE 40

EM40: .ASCIZ 'BUFFER ACTIVE REG ERROR - LINE # '

9825

9826

9827

9828 033422 041522 051126 043040
9829 033430 046101 042523 044440
9830 033436 052116 051105 052522

; INFORMATION FOR MESSAGE 41

EM41: .ASCIZ 'RCVR FALSE INTERRUPT'

```
9831 033444 052120 000
9832
9833 ;INFORMATION FOR MESSAGE 42
9834
9835 033447 123 046111 020117 EM42: .ASCIZ 'SILO OVERFLOW ERROR'
9836 033454 053117 051105 046106
9837 033462 053517 042440 051122
9838 033470 051117 000
9839
9840 ;INFORMATION FOR MESSAGE 43
9841
9842 033473 123 046111 020117 EM43: .ASCIZ 'SILO OVERFLOW FAILED TO GENERATE RCVR INTERRUPT'
9843 033500 053117 051105 046106
9844 033506 053517 043040 044501
9845 033514 042514 020104 047524
9846 033522 043440 047105 051105
9847 033530 052101 020105 041522
9848 033536 051126 044440 052116
9849 033544 051105 052522 052120
9850 033552 000
9851
9852 ;INFORMATION FOR MESSAGE 44
9853
9854 033553 116 047117 042440 EM44: .ASCIZ 'NON EX MEMORY FAILED TO GENERATE XMITTR INTERRUPT'
9855 033560 020130 042515 047515
9856 033566 054522 043040 044501
9857 033574 042514 020104 047524
9858 033602 043440 047105 051105
9859 033610 052101 020105 046530
9860 033616 052111 051124 044440
9861 033624 052116 051105 052522
9862 033632 052120 000
9863
9864 ;INFORMATION FOR MESSAGE 45
9865
9866 033635 130 044515 020124 EM45: .ASCIZ 'XMIT DONE FAILED TO GENERATE XMITTR INTERRUPT'
9867 033642 047504 042516 043040
9868 033650 044501 042514 020104
9869 033656 047524 043440 047105
9870 033664 051105 052101 020105
9871 033672 046530 052111 051124
9872 033700 044440 052116 051105
9873 033706 052522 052120 000
9874
9875 ;INFORMATION FOR MESSAGE 46
9876
9877 033713 103 051125 042522 EM46: .ASCIZ 'CURRENT ADDRESS MEMORY PATTERNS TEST ERROR - LINE # '
9878 033720 052116 040440 042104
9879 033726 042522 051523 046440
9880 033734 046505 051117 020131
9881 033742 040520 052124 051105
9882 033750 051516 052040 051505
9883 033756 020124 051105 047522
9884 033764 020122 020055 044514
9885 033772 042516 021440 020040
9886 034000 000
```

9887	034001	040	050050	024503	DH10: .ASCIZ ' (PC) LINEWR PATRN TEST DEVADR REGADR WAS S/B'
9888	034006	020040	046040	047111	
9889	034014	053505	020122	050040	
9890	034022	052101	051124	020116	
9891	034030	020040	042524	052123	
9892	034036	020040	042040	053105	
9893	034044	042101	020122	051040	
9894	034052	043505	042101	020122	
9895	034060	020040	040527	020123	
9896	034066	020040	020040	027523	
9897	034074	000102			
9898					.EVEN
9899	034076	001116	001202	001204	DT5: .WORD \$ERRPC,\$TMP0,\$TMP1,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0
9900	034104	001162	001164	001166	
9901	034112	001170	001172	000000	
9902					
9903					;INFORMATION FOR MESSAGE 47
9904					
9905	034120	054502	042524	041440	EM47: .ASCIZ 'BYTE COUNT MEMORY PATTERNS TEST ERROR - LINE # '
9906	034126	052517	052116	046440	
9907	034134	046505	051117	020131	
9908	034142	040520	052124	051105	
9909	034150	051516	052040	051505	
9910	034156	020124	051105	047522	
9911	034164	020122	020055	044514	
9912	034172	042516	021440	020040	
9913	034200	000			
9914					;INFORMATION FOR MESSAGE 50
9915					
9916	034201	124	051505	020124	EM50: .ASCIZ 'TEST TIMEOUT WAITING FOR XMIT DONE - LINE # '
9917	034206	044524	042515	052517	
9918	034214	020124	040527	052111	
9919	034222	047111	020107	047506	
9920	034230	020122	046530	052111	
9921	034236	042040	047117	020105	
9922	034244	020055	044514	042516	
9923	034252	021440	020040	000	
9924					
9925					;INFORMATION FOR MESSAGE 51
9926					
9927	034257	116	051120	046040	EM51: .ASCIZ 'NPR LOGIC TEST 2 ERROR'
9928	034264	043517	041511	052040	
9929	034272	051505	020124	020062	
9930	034300	051105	047522	000122	
9931	034306	024040	041520	020051	DH11: .ASCIZ ' (PC) LINACT LINCHK TEST DEVADR REGADR WAS S/B'
9932	034314	020040	044514	040516	
9933	034322	052103	020040	044514	
9934	034330	041516	045510	020040	
9935	034336	052040	051505	020124	
9936	034344	020040	042504	040526	
9937	034352	051104	020040	042522	
9938	034360	040507	051104	020040	
9939	034366	053440	051501	020040	
9940	034374	020040	051440	041057	
9941	034402	000			
9942					

9943
9944
9945
9946
9947
9948
9949
9950

; INFORMATION FOR MESSAGE 52

9951
9952

; INFORMATION FOR MESSAGE 53

9945 034403 102 051501 041511
9946 034410 042040 052101 020101
9947 034416 047503 050115 051101
9948 034424 020105 051105 047522
9949 034432 000122
9953 034434 024040 041520 020051
9954 034442 020040 050123 042505
9955 034450 020104 020040 024040
9956 034456 050123 020051 020040
9957 034464 052040 051505 020124
9958 034472 020040 042504 040526
9959 034500 051104 020040 042522
9960 034506 040507 051104 020040
9961 034514 053440 051501 020040
9962 034522 020040 051440 041057
9963 034530 000

EM52: .ASCIZ 'BASIC DATA COMPARE ERROR'

DH12: .ASCIZ ' (PC) SPEED (SP) TEST DEVADR REGADR W'AS S/B'

9964
9965

; INFORMATION FOR MESSAGE 55

9967 034531 040 050050 024503
9968 034536 020040 020040 050050
9969 034544 024523 020040 020040
9970 034552 051450 024520 020040
9971 034560 020040 042524 052123
9972 034566 020040 042040 053105
9973 034574 042101 020122 041440
9974 034602 051110 047114 020107
9975 034610 051440 051103 040527
9976 034616 020123 051440 051103
9977 034624 027523 000102

DH13: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR CHRLNG SCRWAS SCRS/B'

9978
9979

; INFORMATION FOR MESSAGE 56

9981 034630 053117 051105 052522
9982 034636 020116 044502 020124
9983 034644 040506 046111 042105
9984 034652 052040 020117 042523
9985 034660 020124 020055 044514
9986 034666 042516 021440 020040
9987 034674 000

EM56: .ASCIZ 'OVERRUN BIT FAILED TO SET - LINE # '

9988
9989

; INFORMATION FOR MESSAGE 57

9991 034675 123 047524 040522
9992 034702 042507 047440 042526
9993 034710 043122 047514 020127
9994 034716 044502 020124 040506
9995 034724 046111 042105 026440
9996 034732 046040 047111 020105
9997 034740 020043 000040

EM57: .ASCIZ 'STORAGE OVERFLOW BIT FAILED - LINE # '

9998

```

9999 ; INFORMATION FOR MESSAGE 6C
10000
10001 034744 020040 024040 041520 DH15: .ASCIZ / (PC) DEVADR LINE /
10002 034752 020051 042040 053105
10003 034760 042101 020122 020040
10004 034766 046040 047111 020105
10005 034774 020040 000
10006 035000 ; .EVEN
10007 ; MISCELLANEOUS MESSAGES
10008
10009 035000 005015 055103 044104 TITLE: .ASCIZ <15><12>'CZDHM-D-0 DH11 DIAGNOSTIC'<15><12>
10010 035006 026515 026504 020060
10011 035014 042040 030510 020061
10012 035022 044504 043501 047516
10013 035030 052123 041511 005015
10014 035036 000
10015 035037 015 052012 051505 TITLE2: .ASCIZ <15><12>'TESTING DH11 # '<15><12>
10016 035044 044524 043516 042040
10017 035052 030510 020061 020043
10018 035060 006440 000012
10019 035064 005015 054524 042520 INMSG1: .ASCIZ <15><12>'TYPE SCR ADDRESS FOR FIRST DH11'<15><12>
10020 035072 051440 051103 040440
10021 035100 042104 042522 051523
10022 035106 043040 051117 043040
10023 035114 051111 052123 042040
10024 035122 030510 006461 000012
10025 035130 005015 054524 042520 INMSG2: .ASCIZ <15><12>'TYPE VECTOR ADDRESS FOR FIRST DH11'<15><12>
10026 035136 053040 041505 047524
10027 035144 020122 042101 051104
10028 035152 051505 020123 047506
10029 035160 020122 044506 051522
10030 035166 020124 044104 030461
10031 035174 005015 000
10032 035177 015 052012 050131 INMSG3: .ASCIZ <15><12>'TYPE DH11 DEVICE SELECTION PARAMETER'<15><12>
10033 035204 020105 044104 030461
10034 035212 042040 053105 041511
10035 035220 020105 042523 042514
10036 035226 052103 047511 020116
10037 035234 040520 040522 042515
10038 035242 042524 006522 000012
10039 035250 005015 047111 040526 INMSG4: .ASCIZ <15><12>'INVALID DH11 SCR ADDRESS - TRY AGAIN'<15><12>
10040 035256 044514 020104 044104
10041 035264 030461 051440 051103
10042 035272 040440 042104 042522
10043 035300 051523 026440 052040
10044 035306 054522 040440 040507
10045 035314 047111 005015 000
10046 035321 015 044412 053116 INMSG5: .ASCIZ <15><12>'INVALID DH11 VECTOR ADDRESS - TRY AGAIN'<15><12>
10047 035326 046101 042111 042040
10048 035334 030510 020061 042526
10049 035342 052103 051117 040440
10050 035350 042104 042522 051523
10051 035356 026440 052040 054522
10052 035364 040440 040507 047111
10053 035372 005015 000
10054 035375 015 052012 050131 INMSG6: .ASCIZ <15><12>'TYPE LINE SELECTION PARAMETER'<15><12>

```

10055	035402	020105	044514	042516
10056	035410	051440	046105	041505
10057	035416	044524	047117	050040
10058	035424	051101	046501	052105
10059	035432	051105	005015	000
10060	035437	015	042012	050105
10061	035444	042522	051523	021040
10062	035452	047503	052116	047111
10063	035460	042525	020042	047524
10064	035466	051440	040524	052122
10065	035474	052040	051505	044524
10066	035502	043516	005015	000
10067				
10068	035507	015	052012	050131
10069	035514	020105	047516	020056
10070	035522	043117	040440	042104
10071	035530	042522	051523	051505
10072	035536	024040	041517	040524
10073	035544	024514	041040	052105
10074	035552	042527	047105	053040
10075	035560	041505	047524	051522
10076	035566	024040	030061	047440
10077	035574	020122	030062	006451
10078	035602	000012		
10079	035604	047516	042040	030510
10080	035612	023461	020123	042527
10081	035620	042522	043040	052517
10082	035626	042116	006456	000012
10083	035634	047516	042040	020110
10084	035642	042522	042503	053111
10085	035650	051105	044440	052116
10086	035656	051105	052522	052120
10087	035664	047440	041503	051125
10088	035672	042522	027104	005015
10089	035700	000		
10090	035701	116	020117	047515
10091	035706	042504	020115	047503
10092	035714	052116	047522	020114
10093	035722	047111	042524	051122
10094	035730	050125	020124	041517
10095	035736	052503	051122	042105
10096	035744	006456	000012	
10097	035750	020040	000040	
10098	035754	005015	044104	030461
10099	035762	020054	047515	042504
10100	035770	020115	047503	052116
10101	035776	047522	020114	042504
10102	036004	044526	042503	046440
10103	036012	050101	006472	012
10104	036017	015	042012	030510
10105	036024	020061	020040	044104
10106	036032	030461	020040	046440
10107	036040	042117	046505	041440
10108	036046	047117	051124	046117
10109	036054	020040	046440	042117
10110	036062	046505	041440	047117

INMSG7: .ASCIZ <15><12>'DEPRESS "CONTINUE" TO START TESTING'<15><12>

VCWC: .ASCIZ <15><12>'TYPE NO. OF ADDRESSES (OCTAL) BETWEEN VECTORS (10 OR 20)'<15><1

MSG1: .ASCIZ /NO DH11'S WERE FOUND./<15><12>

MSG2: .ASCIZ 'NO DH RECEIVER INTERRUPT OCCURRED.'<15><12>

MSG3: .ASCIZ 'NO MODEM CONTROL INTERRUPT OCCURRED.'<15><12>

SPACE: .ASCIZ ' '
DEVMAP: .ASCII <15><12>'DH11, MODEM CONTROL DEVICE MAP:'<15><12>

.ASCII <15><12>'DH11 DH11 MODEM CONTROL MODEM CONTROL'


```
10111 036070 051124 046117
10112 036074 005015 042101 051522      .ASCIZ <15><12>'ADRS VECT ADRS VECT'<15><12><15><12>
10113 036102 020040 053040 041505
10114 036110 020124 020040 040440
10115 036116 051104 020123 020040
10116 036124 020040 053040 041505
10117 036132 006524 006412 000012
10118 036140 005015 047515 042504 MSG4: .ASCIZ <15><12>'MODEM CONTROL ERROR, RUN DZDHK DIAGNOSTIC'<15><12>
10119 036146 020115 047503 052116
10120 036154 047522 020114 051105
10121 036162 047522 026122 051040
10122 036170 047125 042040 042132
10123 036176 045510 042040 040511
10124 036204 047107 051517 044524
10125 036212 006503 000012
10126 036216 005015 047516 046440 MSG5: .ASCIZ <15><12>/NO MODEM CONTROL FOUND/<15><12>
10127 036224 042117 046505 041440
10128 036232 047117 051124 046117
10129 036240 043040 052517 042116
10130 036246 005015 000
10131 036252 .EVEN
10132 ;SIXTEEN CHAR COUNTERS USED BY THE AUTO ECHO TEST #3
10133
10134 036252 000020 RCNT: .BLKW 16.
10135
10136 ;256. WORD RECEIVER INPUT BUFFER
10137
10138 036312 000400 RBUF: .BLKW 256.
10139
10140
10141 ;256(10) BYTE TRANSMITTER OUTPUT DATA BUFFER
10142
10143 .EVEN
10144 037312 000400 TBUF: .BLKB 256.
10145
10146 000001 .END
```

ABASE = 000000	2437	2478						
ACDW1 = 000000	2437	2480						
ACDW2 = 000000	2437	2481						
ACPUOP= 000000	2437	2452						
ADDW0 = 000000	2437	2482						
ADDW1 = 000000	2437	2483						
ADDW10= 000000	2437	2492						
ADDW11= 000000	2437	2493						
ADDW12= 000000	2437	2494						
ADDW13= 000000	2437	2495						
ADDW14= 000000	2437	2496						
ADDW15= 000000	2437	2497						
ADDW2 = 000000	2437	2484						
ADDW3 = 000000	2437	2485						
ADDW4 = 000000	2437	2486						
ADDW5 = 000000	2437	2487						
ADDW6 = 000000	2437	2488						
ADDW7 = 000000	2437	2489						
ADDW8 = 000000	2437	2490						
ADDW9 = 000000	2437	2491						
ADEVCT= 000000	2437	2443						
ADEVN = 000000	2437	2479						
ADPTR 030326	2938*	2945*	2953	7643*	9409#			
ADRVEC 030306	2927	8777*	9398#					
AENV = 000000	2437	2448						
AENVN = 000000	2437	2449						
AETAB 027420	6725	6832	6960	7002	7006	7007	9139#	
AETAB0 027460	6795	9159#						
AFATAL= 000000	2437	2440						
AMADR1= 000000	2437	2465						
AMADR2= 000000	2437	2469						
AMADR3= 000000	2437	2472						
AMADR4= 000000	2437	2475						
AMAMS1= 000000	2437	2459						
AMAMS2= 000000	2437	2467						
AMAMS3= 000000	2437	2470						
AMAMS4= 000000	2437	2473						
AMSGAD= 000000	2437	2445						
AMSGLG= 000000	2437	2446						
AMSGTY= 000000	2437	2439						
AMTYP1= 000000	2437	2460						
AMTYP2= 000000	2437	2468						
AMTYP3= 000000	2437	2471						
AMTYP4= 000000	2437	2474						
APASS = 000000	2437	2442						
APRIOR= 000000	2437							
APTCSU= 000040	8027	8132#						
APTENV= 000001	7783	8020	8088	8130#				
APTSIZ= 000200	2896	8129#						
APTSPO= 000100	8022	8090	8131#					
ASWREG= 000000	2437	2450						
AESTN= 000000	2437	2441						
AUNIT = 000000	2437	2444						
AUSWR = 000000	2437	2451						
AUTOSL 025140	2922	8616#						
AVECT1= 000000	2437	2476						

\$TSTM	000220	2257#																
\$TSTNM	001102	2382#	6563	6855	6970	7650*	7662*	7695	7724	7746*	7747	7752	7756	7773				
		7806	8461	8473	8496	8960	8973											
\$TTYIN	023654	8244	8245	8257	8275	8289	8293#											
\$TYPBN=	***** U	8389																
\$TYPDS	022042	7942#	8388															
\$TYPE	022266	8014#	8107	8376	8384													
\$TYPEC	022500	8044	8051	8058	8063#	8064	8187											
\$TYPEX	022546	8069	8071	8074#														
\$TYPOC	021640	7883#	8385															
\$TYPON	021654	7882	7885#	8387														
\$TYPOS	021614	7878#	8386															
\$UNIT	001244	2444#																
\$UNITM	000224	2259#																
\$USWR	001256	2451#																
\$VECT1	001302	2476#																
\$VECT2	001304	2477#																
\$XTSTR	021026	7711#																
\$GET4=	000000	7678#																
\$OFILL	022037	7879*	7883*	7893	7928#													
\$LOCAT=	***** U	7708	7780															
.	= 037712	2217#	2221#	2233	2234#	2236#	2238#	2244	2245#	2247#	2249#	2379#	2433	2862				
		2877	2878	2963	7466	7522	7577	7632	7686	7690	7755	7756	7806	7852#				
		7996#	8076	8128#	8136	8293#	8294	8300	8353	8412	8436	9099#	9203#	9483#				
		9797#	10006#	10131#	10134#	10138#	10144#											
.\$ASTA=	***** U	8080	8083															
.\$X	= 000214	2244#	2249															

. ABS. 037712 000

ERRORS DETECTED: 0

DSKZ:CZDHMD,DSKZ:CZDHMD.SEQ=DSKZ:CZDHMD.SML,DHMMAD.P11,CZDHMD.P11

RUN-TIME: 32 38 1 SECONDS
RUN-TIME RATIO: 254/72=3.5
CORE USED: 49K (97 PAGES)

DOCUMENT PAGES: 237