



4 T2 W  
A ::

SEQ 000

1  
CVDH8DO DHV11-M FUNC TST PART 2 MACRO Y05.02 Monday 01-Apr-85 07:53 Page 2  
PROGRAM DOCUMENT

.REM &amp;

IDENTIFICATION  
-----

PRODUCT CODE: AC T654D-MC  
PRODUCT NAME: CVDH8DO DHV11-M FUNC TST PART 2  
PRODUCT DATE: 29 MARCH 1985  
MAINTAINER: Bruce Ribolini - MK Diagnostics Group  
AUTHOR: Bert Kleinschmidt  
Tony Grimshaw  
MODIFIED BY: Bert Kleinschmidt  
Anthony Hart  
Peter ONeil

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 CORPORATION 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) 1983, 1984, 1985 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL  
DECPDP  
DECUSUNIBUS  
DECTAPE

MASSBUS

PROGRAM DOCUMENT

## \*\*\*\*\* MODIFICATION HISTORY \*\*\*\*\*

Original release: 31-OCT-83 (EDITED 11-JUL-83)  
Bert Kleinschmidt

Version B0 09-OCT-83 Bert Kleinschmidt  
Fixed typographical errors.  
Moved tests from this program to CVDHA (Part 1):  
Old CVDHB (version A) tests 2 through 8 are  
now New CVDHA (version B) tests 20 through 26.  
Moved test from CVDHC (Part 3) into this program:  
Old CVDHC (version A) tests 4 through 6 are  
now New CVDHB (version B) tests 13 through 15.

Version C0 16-DEC-83 Bert Kleinschmidt  
Fixed typographical errors discovered to date.  
Added 2 tests to allow this program to be used for a  
complete checkout of cables and distribution panels.  
New test 24 - Tests continuity of TX/RX lines.  
New test 25 - Detects TX line interactions.

Version D0 17-Jul-84 Bert Kleinschmidt  
Fixed typographical errors discovered to date.  
Modified control of processor priority and LTC throughout  
the program to guarantee a less than 2 second  
response to a Break request while running under APT.  
Fixed bug in IAUTO Active test which caused an XON  
character to arrive late on processors with cache.

Version D0 28-Sep-84 Peter O'Neil  
Modified clean up section to turn off clock if clock  
had been turned on.

29 Mar 85 Howard Marshall  
Did formal release of version CVDH8D0.

PROGRAM DOCUMENT

TABLE OF CONTENTS

1.0 GENERAL PROGRAM CONSIDERATIONS . . . . . 4

1.1 PROGRAM ABSTRACT . . . . . 4

1.2 SYSTEM REQUIREMENTS . . . . . 4

1.3 RELATED DOCUMENTS AND STANDARDS . . . . . 4

1.4 DIAGNOSTIC HIERARCY PREREQUISITES . . . . . 5

2.0 OPERATING INSTRUCTIONS . . . . . 5

2.1 COMMANDS . . . . . 5

2.2 SWITCHES . . . . . 6

2.3 FLAGS . . . . . 7

2.4 EXTENDED COMMAND SYNTAX . . . . . 8

2.4.1 START COMMAND . . . . . 8

2.4.1.1 Tests Switch (/TESTS:<TEST-LIST>) . . . . . 8

2.4.1.2 Pass Switch (/PASS:<PASS-CNT>) . . . . . 8

2.4.1.3 Flags Switch (/FLAGS:<FLAG-LIST>) . . . . . 8

2.4.1.4 End Of Pass Switch (/EOP:<INCR>) . . . . . 8

2.4.1.5 Effect Of Start Command . . . . . 8

2.4.2 Restart Command . . . . . 10

2.4.2.1 Tests, Pass, And Flags Switches . . . . . 10

2.4.2.2 Units Switch (/UNITS:<UNIT-LIST>) . . . . . 10

2.4.2.3 Effect Of Restart Command . . . . . 10

2.4.3 Continue Command . . . . . 10

2.4.3.1 Flag Switch (/FLAGS:<FLAG-LIST>) . . . . . 10

2.4.3.2 Effect Of Continue Command . . . . . 10

2.4.4 Proceed Command . . . . . 11

2.4.4.1 Flags Switch (/FLAGS:<FLAG-LIST>) . . . . . 11

2.4.4.2 Effect Of Proceed Command . . . . . 11

2.4.5 Add Command . . . . . 11

2.4.6 EFFECT OF ADD COMMAND . . . . . 11

2.4.7 Drop Command . . . . . 12

2.4.8 EFFECT OF DROP COMMAND . . . . . 12

2.4.9 Print Command . . . . . 12

2.4.9.1 Effect Of Print Command . . . . . 12

2.4.10 Display Command . . . . . 12

2.4.10.1 Effect Of Display Command . . . . . 12

2.4.11 Flags Command . . . . . 12

2.4.11.1 Effect Of Flags Command . . . . . 12

2.4.12 Zflags Command . . . . . 13

2.4.13 Zflags Command . . . . . 13

2.4.14 Control Characters . . . . . 13

2.5 HARDWARE QUESTIONS . . . . . 13

2.6 SOFTWARE QUESTIONS . . . . . 14

2.7 EXTENDED P-TABLE DIALOGUE . . . . . 15

2.8 QUICK START-UP PROCEDURE (XXDP+) . . . . . 17

3.0 ERROR INFORMATION . . . . . 17

3.1 TYPES OF ERROR MESSAGES . . . . . 17

3.2 ERROR MESSAGES . . . . . 18

4.0 PERFORMANCE AND PROGRESS REPORTS . . . . . 18

5.0 TEST SUMMARIES . . . . . 19

6.0 EXAMPLE ERROR FREE PASS . . . . . 21

## PROGRAM DOCUMENT

## 1.0 GENERAL PROGRAM CONSIDERATIONS

## 1.1 PROGRAM ABSTRACT

CVDHB is part two of the DHV11-M functional verification test. This part of the test verifies that the major communication functions of the board are functioning correctly. This program does not perform extensive data transmission and reception tests.

THIS DIAGNOSTIC HAS BEEN WRITTEN FOR USE WITH THE DIAGNOSTIC RUNTIME SERVICES SOFTWARE (SUPERVISOR). THESE SERVICES PROVIDE THE INTERFACE TO THE OPERATOR AND TO THE SOFTWARE ENVIRONMENT. THIS PROGRAM CAN BE USED WITH XXDP+, ACT, APT, SLIDE AND PAPER TAPE. FOR A COMPLETE DESCRIPTION OF THE RUNTIME SERVICES, REFER TO THE XXDP+ USER'S MANUAL. THERE IS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES IN THE OPERATING INSTRUCTIONS-COMMANDS OF THIS DOCUMENT.

## 1.2 SYSTEM REQUIREMENTS

The following hardware is required to run the DHV FVT:

- o LSI-11 processor with at least 32 Kbytes of RAM.
- o DHV11 boards installed on the Q-bus.
- o Appropriate program load device supporting XXDP+ media or a down line loading system.

## 1.3 RELATED DOCUMENTS AND STANDARDS

- o DHV11-M Hardware Manual - This manual describes the functions and uses of the DHV11-M device.
- o XXDP+ User's Manual - Describes the running of diagnostics under the XXDP+ monitor.

PROGRAM DOCUMENT

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

The LSI-11 processor, the Q-BUS, the system memory, the console terminal, and the load media are assumed to have been tested and found working before th's program is run.

2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES. FOR DETAILED INFORMATION, REFER TO THE XXDP+ USER'S MANUAL (CHQUS).

2.1 COMMANDS

THERE ARE ELEVEN LEGAL COMMANDS FOR THE DIAGNOSTIC RUNTIME SERVICES (SUPERVISOR). THIS SECTION LISTS THE COMMANDS AND GIVES A VERY BRIEF DESCRIPTION OF THEM. THE XXDP+ USER'S MANUAL HAS MORE DETAILS.

COMMAND	EFFECT
START	START THE DIAGNOSTIC FROM AN INITIAL STATE
RESTART	START THE DIAGNOSTIC WITHOUT INITIALIZING
CONTINUE	CONTINUE AT TEST THAT WAS INTERRUPTED (AFTER +C)
PROCEED	CONTINUE FROM AN ERROR HALT
EXIT	RETURN TO XXDP+ MONITOR (XXDP+ OPERATION ONLY!)
ADD	ACTIVATE A UNIT FOR TESTING (ALL UNITS ARE CONSIDERED TO BE ACTIVE AT START TIME)
DROP	DEACTIVATE A UNIT
PRINT	PRINT STATISTICAL INFORMATION (IF IMPLEMENTED BY THE DIAGNOSTIC - SEE PERFORMANCE AND PROGRESS REPORTS SECTION OF THIS DOCUMENT)
DISPLAY	TYPE A LIST OF ALL DEVICE INFORMATION
FLAGS	TYPE THE STATE OF ALL FLAGS (SEE FLAGS SECTION)
ZFLAGS	CLEAR ALL FLAGS (SEE FLAGS SECTION)

A COMMAND CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. SO YOU MAY, FOR EXAMPLE, TYPE "STA" INSTEAD OF "START". MORE INFORMATION CAN BE FOUND WITHIN THE SECTION LABELLED EXTENDED COMMAND SYNTAX

## PROGRAM DOCUMENT

## 2.2 SWITCHES

THERE ARE SEVERAL SWITCHES WHICH ARE USED TO MODIFY SUPERVISOR OPERATION. THESE SWITCHES ARE APPENDED TO THE LEGAL COMMANDS. ALL OF THE LEGAL SWITCHES ARE TABULATED BELOW WITH A BRIEF DESCRIPTION OF EACH. IN THE DESCRIPTIONS BELOW, A DECIMAL NUMBER IS DESIGNATED BY "DDDDD".

SWITCH	EFFECT
/TESTS:LIST	EXECUTE ONLY THOSE TESTS SPECIFIED IN THE LIST. LIST IS A STRING OF TEST NUMBERS, FOR EXAMPLE - /TESTS:1:5:7-10. THIS LIST WILL CAUSE TESTS 1,5,7,8,9,10 TO BE RUN. ALL OTHER TESTS WILL NOT BE RUN.
/PASS:DDDDD	EXECUTE DDDDD PASSES (DDDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS. SEE THE FLAGS SECTION OF THIS DOCUMENT.
/EOP:DDDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY. (DDDDD = 1 TO 64000)
/UNITS:LIST	TEST/ADD/DROP ONLY THOSE UNITS SPECIFIED IN THE LIST. LIST EXAMPLE - /UNITS:0:5:10 12 USE UNITS 0,5,10,11,12 (UNIT NUMBERS = 0 63)

## EXAMPLE OF SWITCH USAGE:

START/TESTS:1-5/PASS:1000/EOP:100

THE EFFECT OF THIS COMMAND WILL BE: 1) TESTS 1 THROUGH 5 WILL BE EXECUTED, 2) ALL UNITS WILL TESTED 1000 TIMES AND 3) THE END OF PASS MESSAGES WILL BE PRINTED AFTER EACH 100 PASSES ONLY. A SWITCH CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. YOU MAY, FOR EXAMPLE, TYPE "/TES:1-5" INSTEAD OF "/TESTS:1-5".

BELOW IS A TABLE THAT SPECIFIES WHICH SWITCHES CAN BE USED BY EACH COMMAND.

	TESTS	PASS	FLAGS	EOP	UNITS
START	X	X	X	X	X
RESTART	X	X	X	X	X
CONTINUE		X	X	X	
PROCEED			X		
DROP					X
ADD					X
PRINT					
DISPLAY					X
FLAGS					
ZFLAGS					
EXIT					

## PROGRAM DOCUMENT

## 2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
HOE	HALT ON ERROR - CONTROL IS RETURNED TO RUNTIME SERVICES COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
IBR*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)
IXR*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
PNT	PRINT TEST NUMBER AS TEST EXECUTES
BOE	"BELL" ON ERROR
UAM	UNATTENDED MODE (NO MANUAL INTERVENTION)
ISR	INHIBIT STATISTICAL REPORTS (DOES NOT APPLY TO DIAGNOSTICS WHICH DO NOT SUPPORT STATISTICAL REPORTING)
IDR	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE
LOT	LOOP ON TEST
EVL	EXECUTE EVALUATION (ON DIAGNOSTICS WHICH HAVE EVALUATION PORT)

\*SEE THE ERROR INFORMATION SECTION OF THIS DOCUMENT.

SEE THE XXDP+ USER'S MANUAL FOR MORE DETAILS ON FLAGS. YOU MAY SPECIFY MORE THAN ONE FLAG WITH THE FLAG SWITCH. FOR EXAMPLE, TO CAUSE THE PROGRAM TO LOOP ON ERROR, INHIBIT ERROR REPORTS AND TYPE A "BELL" ON ERROR, YOU MAY USE THE FOLLOWING STRING:

/FLAGS:LOE:IER:BOE



PROGRAM DOCUMENT

2.4 EXTENDED COMMAND SYNTAX

2.4.1 START COMMAND -

```

*****
STA(RT)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:
<FLAG-LIST>/EOP:<INCR>
*****

```

2.4.1.1 Tests Switch (/TESTS:<TEST-LIST>) -

<TEST-LIST> Is a sequence of decimal numbers (1:2 etc.) or ranges of decimal numbers (1-5:8-10 etc.), seperated by colons, that specify the tests to be executed. Tests will be executed in numerical order regardless of the order of specification. The default is to execute all tests. On this and all switches, the angle brackets <> are punctuation used in the definition only, and are not to be typed by the operator. See example at end of "Effect of Start Command" section.

2.4.1.2 Pass Switch (/PASS:<PASS-CNT>) - :

<PASS-CNT> Is a decimal number indicating the desired number of passes. A pass is defined as the execution of the full diagnostic (all selected tests). The default is non-ending execution. In this case, exit from the program is accomplished either by typing a control/C or by occurrence of an error with the halt on error flag being set. The exit is a return to command mode. See example at end of "Effect of Start Command" section.

2.4.1.3 Flags Switch (/FLAGS:<FLAG-LIST>) -

<FLAG-LIST> is a sequence of elements of the form <FLAG>, <FLAG=1>, or <FLAG=0>, separated by colons, where <FLAG> has one of the following values:

- HOE Halt on error, causing command mode to be entered when an error is encountered.
- LOE Loop on error, causing the diagnostic to loop continuously within the smallest defined block of coding (segment, subtest, or test) containing the error.
- IER Inhibit error reporting.
- IBE Inhibit basic error reports.
- IXE Inhibit extended error reports.
- PRI Direct all messages to a line printer.
- PNT Print number of test being executed.
- BOE Bell on error.
- UAM Run in unattended mode, bypassing manual

## PROGRAM DOCUMENT

intervention.  
 ISR Inhibit statistical reports.  
 IDU Inhibit dropping of units by diagnostic.  
 LOT Loop on test.

The flags named or equated to 1 are set, those equated to 0 are cleared. A flag not specified is cleared. If the flags switch is not given all flags are cleared. See example at end of "Effect of Start Command" section.

## 2.4.1.4 End Of Pass Switch (/EOP:&lt;INCR&gt;) -

<INCR> Is a decimal number indicating how often (in terms of passes) it is desired that the end of pass message be printed. The default is at the end of every pass. See example at end of "Effect of Start Command" section.

## 2.4.1.5 Effect Of Start Command -

The effect of the start command is to initiate the hardware parameter dialogue, the software parameter dialogue, the initialization questions, and then the diagnostic commences testing.

The hardware parameter dialogue commences with the question "# UNITS (D) ?" to which the operator should reply with the number of units to be tested. Following this are the questions whereby the P-Tables themselves are built. Each P Table is a core-resident table containing all the hardware information for one complete unit. Each question is followed by the response radix (D for decimal, B for binary, O for octal, L for Yes/No) in parentheses and the default value after the parentheses. For the actual Hardware P-Table questions see the "Hardware Parameters" section.

Following the hardware questions are the software questions to build the software tables, which define operating parameters of the diagnostic program. These Questions are described in the "Software Parameters" section.

## EXAMPLE:

STA/TESTS:1:3-4:/PASS:3/FLAGS:IER:HOE=1

This command will cause three passes to be made, with each pass consisting of tests 1,3, and 4. There is no difference between saying <FLAG> and saying <FLAG=1>. The notation <FLAG=0> is meaningful only on a command other than start to clear a flag that was previously set. Note that on all commands only the first three letters are scanned.

PROGRAM DOCUMENT

2.4.2 Restart Command -

```

*****
RES(TART)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:
<FLAG-LIST>/UNITS:<UNIT-LIST>
*****

```

2.4.2.1 Tests, Pass, And Flags Switches -

<TEST-LIST>, <PASS-CNT>, and <FLAG-LIST> are as in the start command.

2.4.2.2 Units Switch (/UNITS:<UNIT-LIST>) - <UNIT LIST> IS A SEQUENCE OF DECIMAL NUMBERS (0,1 ETC.) OR RANGES OF DECIMAL NUMBERS (0-5, 8-10 ETC.) THAT SPECIFY THE UNITS TO BE TESTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS MAY RANGE FROM 0 THRU N-1 (N IS THE NUMBER OF UNITS SPECIFIED IN THE PREVIOUS START COMMAND). THE NUMBER INDICATES THE POSITION OF THE P-TABLE AS THE DATA WAS ENTERED DURING THE HARDWARE DIAGLOGUE. THE UNITS WHICH ARE SELECTED MUST NOT HAVE BEEN DROPPED BY THE DROP COMMAND. SEE THE DISCUSSION OF ADD AND DROP COMMANDS BELOW. DEFAULT IS TO TEST ALL UNITS WHICH HAVE NOT BEEN DROPPED BY A DROP COMMAND.

2.4.2.3 Effect Of Restart Command -

The restart command differs from the start command in that the P-Tables from the previous start command (there must have been one) are used, instead of new ones being built. The software dialogue may optionally be reexecuted (operator will be asked). The command can be used after command mode has been reentered in any of the three normal ways: a) the requested number of passes have been made, b) an error was encountered with the halt on error flag set, or c) a control/C was entered by the operator.

2.4.3 Continue Command -

```

*****
CON(TINUE)/PASS:<PASS-CNT>/FLAGS:<FLAG-LIST>
*****

```

PROGRAM DOCUMENT

2.4.3.1 Flag Switch (/FLAGS:<FLAG-LIST>) -

<FLAG-LIST> Is same as in the start command, but unspecified flags retain their current value.

2.4.3.2 Effect Of Continue Command -

Continue must follow a start or restart, and command mode must have been entered due to a halt on error or a control/C. The effect of the command is to go to the beginning of the test that was being executed when the halt or control/C took place. Software dialogue may optionally be reexecuted. Hardware parameters may not be changed.

2.4.4 Proceed Command -

\*\*\*\*\*  
PRO(CEED)/FLAGS:<FLAG-LIST>  
\*\*\*\*\*

2.4.4.1 Flags Switch (/FLAGS:<FLAG-LIST>) -

<FLAG-LIST> Is as in the start command, but unspecified flags retain their current value.

2.4.4.2 Effect Of Proceed Command -

Proceed must follow a start, restart, or continue. Command mode must have been entered via a halt on error. The effect of the command is to begin execution at the location following the error call. Neither hardware nor software parameters may be altered.

2.4.5 Add Command -

\*\*\*\*\*  
ADD/UNITS:<UNIT-LIST>  
\*\*\*\*\*

2.4.6 EFFECT OF ADD COMMAND - THE UNITS SPECIFIED ARE ADDED TO THE TEST SEQUENCE. EACH UNIT MUST HAVE A P-TABLE IN MEMORY DUE TO AN EARLIER HARDWARE DIALOGUE. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR CONTINUE. THE UNITS SWITCH MUST BE SPECIFIED. THE ADD COMMAND IS MEANINGFUL ONLY FOR UNITS THAT WERE PREVIOUSLY DROPPED.

PROGRAM DOCUMENT

2.4.7 Drop Command -

\*\*\*\*\*  
DRO(P)/UNITS:<UNIT-LIST>  
\*\*\*\*\*

2.4.8 EFFECT OF DROP COMMAND - THE UNITS SPECIFIED WILL BE DROPPED FROM TESTING. THE UNITS WILL BE RESELECTED ONLY BY THE EXECUTION OF AN ADD OR START COMMAND. THE UNITS SWITCH MUST BE ENTRED. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR A CONTINUE COMMAND.

2.4.9 Print Command -

\*\*\*\*\*  
PRI(NT)  
\*\*\*\*\*

2.4.9.1 Effect Of Print Command - Error summary reporting is not implemented in this diagnostic, so this command has no effect.

2.4.10 Display Command -

\*\*\*\*\*  
DIS(PLAY)/UNITS:<UNIT-LIST>  
\*\*\*\*\*

2.4.10.1 Effect Of Display Command -

The hardware P-Tables for all units are printed in the format in which they were entered.

2.4.11 Flags Command -

\*\*\*\*\*  
FLA(GS)  
\*\*\*\*\*

2.4.11.1 Effect Of Flags Command -

The current settings of all flags are printed.

## PROGRAM DOCUMENT

## 2.4.12 Zflags Command -

```
*****
ZFL(AGS)
*****
```

## 2.4.13 Zflags Command -

All flags are cleared.

## 2.4.14 Control Characters -

- C A control/C (C) entered during the execution of a diagnostic causes a return to command mode.
- Z A control/Z (Z) entered during one of the two operator dialogues-- hardware P-Table dialogue or software P-Table dialogue causes the defaults to be taken for the remainder of that dialogue.
- O A control/O (O) entered during the execution of a diagnostic causes all teletype output to be suppressed for the remainder of the diagnostic or until another control/O is typed, which restores normal teletype output.

## 2.5 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE RUNTIME SERVICES WILL PROMPT THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?" YOU MUST ANSWER "Y" AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN "PRELOADED" USING THE SETUP UTILITY (SEE CHAPTER 6 OF THE XXDP-USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A "Y", THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL). YOU WILL THEN BE ASKED THE FOLLOWING QUESTIONS FOR EACH UNIT:

1. CSR ADDRESS - This question requests the CSR address of the specified DHV11. The default answer for this question is the lowest address in the PDP-11 floating address space in which a DHV11-M can be placed (160460 Octal).
2. INTERRUPT VECTOR ADDRESS - This question requests the interrupt vector address of the specified DHV11.
3. ACTIVE LINES BIT MAP - This question requests an octal bit map of the serial communication lines on the DHV11 which are being selected for testing. If the bit in the bit map is set which corresponds to a particular line (i.e. bit 3 for line

## PROGRAM DOCUMENT

3) that line will be tested by the FVT. With staggered loopback a pair of lines with the specified transmit line and another receive line will be tested. Therefore, to guarantee that both the transmitter and receiver of a specified line are tested when using the staggered loopback connector, both the intended line AND its mate must be selected (ie. to test line 1, select both line 1 and line 3). In nonstaggered testing, a bit in the active lines bit map selects the transmitter and receiver for the same line.

4. TYPE OF LOOPBACK (1=INTERNAL, 2=H3277, 3=H325) - This question requests the type of loopback to be used in testing the DHV11. The following types of loopback are supported:
  - o INTERNAL - Only internal UART loopback is to be used in testing the DHV.
  - o H3277 - Staggered Berg connector(s) are installed at the end of the 40 wire cables in place of the DHV11 distribution panels.
  - o H325 - Single line, 25 pin loopback connectors (type H325) are installed on the lines to be tested. These connectors can be installed on the distribution panel or on the end of the terminal or modem cable. The H325 connectors must have the removable jumpers installed.
  
5. BR Level - This question requests the interrupt BR level of the DHV11.

## 2.6 SOFTWARE QUESTIONS

AFTER YOU HAVE ANSWERED THE HARDWARE QUESTIONS OR AFTER A RESTART OR CONTINUE COMMAND, THE RUNTIME SERVICES WILL ASK FOR SOFTWARE PARAMETERS. THESE PARAMETERS WILL GOVERN SOME DIAGNOSTIC SPECIFIC OPERATION MODES. YOU WILL BE PROMPTED BY "CHANGE SW (L) ?" IF YOU WISH TO CHANGE ANY PARAMETERS, ANSWER BY TYPING "Y". The following Software P-Table questions are asked by the program if the operator indicates that the Software Parameters are to be changed:

1. REPORT UNIT NUMBER AS EACH UNIT IS TESTED - This question asks whether the program should report the number of the unit which it is testing as it begins to test each unit. The unit number will only be reported if more than one unit is being tested.
  
2. NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE - This question asks for the number of data errors which should be reported individually by this program for each line for each transmission test. Errors which are not reported individually are reported in summary error reports.

## PROGRAM DOCUMENT

## 2.7 EXTENDED P-TABLE DIALOGUE

WHEN YOU ANSWER THE HARDWARE QUESTIONS, YOU ARE BUILDING ENTRIES IN A TABLE THAT DESCRIBES THE DEVICES UNDER TEST. THE SIMPLEST WAY TO BUILD THIS TABLE IS TO ANSWER ALL QUESTIONS FOR EACH UNIT TO BE TESTED. IF YOU HAVE A MULTIPLEXED DEVICE SUCH AS A MASS STORAGE CONTROLLER WITH SEVERAL DRIVES OR A COMMUNICATION DEVICE WITH SEVERAL LINES, THIS BECOMES TEDIOUS SINCE MOST OF THE ANSWERS ARE REPETITIOUS.

TO ILLUSTRATE A MORE EFFICIENT METHOD, SUPPOSE YOU ARE TESTING A FICTIONAL DEVICE, THE XY11. SUPPOSE THIS DEVICE CONSISTS OF A CONTROL MODULE WITH EIGHT UNITS (SUB-DEVICES) ATTACHED TO IT. THESE UNITS ARE DESCRIBED BY THE OCTAL NUMBERS 0 THROUGH 7. THERE IS ONE HARDWARE PARAMETER THAT CAN VARY AMONG UNITS CALLED THE Q-FACTOR. THIS Q-FACTOR MAY BE 0 OR 1. BELOW IS A SIMPLE WAY TO BUILD A TABLE FOR ONE XY11 WITH EIGHT UNITS.

```
# UNITS (0) ? 8<CR>
```

```
UNIT 1
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 0<CR>
```

```
Q-FACTOR (0) 0 ? 1<CR>
```

```
UNIT 2
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 1<CR>
```

```
Q-FACTOR (0) 1 ? 0<CR>
```

```
UNIT 3
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 2<CR>
```

```
Q-FACTOR (0) 0 ? <CR>
```

```
UNIT 4
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 3<CR>
```

```
Q FACTOR (0) 0 ? <CR>
```

```
UNIT 5
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 4<CR>
```

```
Q-FACTOR (0) 0 ? <CR>
```

```
UNIT 6
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 5<CR>
```

```
Q-FACTOR (0) 0 ? <CR>
```

```
UNIT 7
```

```
CSR ADDRESS (0) ? 160000<CR>
```

```
SUB-DEVICE # (0) ? 6<CR>
```

```
Q-FACTOR (0) 0 ? 1<CR>
```

```
UNIT 8
```

```
CSR ADDRESS (0) 160000<CR>
```

```
SUB-DEVICE # (0) ? 7<CR>
```



PROGRAM DOCUMENT

Q FACTOR (0) 1 ? &lt;CR&gt;

NOTICE THAT THE DEFAULT VALUE FOR THE Q-FACTOR CHANGES WHEN A NON-DEFAULT RESPONSE IS GIVEN. BE CAREFUL WHEN SPECIFYING MULTIPLE UNITS!

AS YOU CAN SEE FROM THE ABOVE EXAMPLE, THE HARDWARE PARAMETERS DO NOT VARY SIGNIFICANTLY FROM UNIT TO UNIT. THE PROCEDURE SHOWN IS NOT VERY EFFICIENT.

THE RUNTIME SERVICES CAN TAKE MULTIPLE UNIT SPECIFICATIONS HOWEVER. LET'S BUILD THE SAME TABLE USING THE MULTIPLE SPECIFICATION FEATURE.

# UNITS (0) ? 8&lt;CR&gt;

UNIT 1  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 0,1<CR>  
Q-FACTOR (0) 0 ? 1,0<CR>

UNIT 3  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 2-5<CR>  
Q-FACTOR (0) 0 ? 0<CR>

UNIT 7  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 6,7<CR>  
Q-FACTOR (0) 0 ? 1<CR>

AS YOU CAN SEE IN THE ABOVE DIALOGUE, THE RUNTIME SERVICES WILL BUILD AS MANY ENTRIES AS IT CAN WITH THE INFORMATION GIVEN IN ANY ONE PASS THROUGH THE QUESTIONS. IN THE FIRST PASS, TWO ENTRIES ARE BUILT SINCE TWO SUB-DEVICES AND Q-FACTORS WERE SPECIFIED. THE SERVICES ASSUME THAT THE CSR ADDRESS IS 160000 FOR BOTH SINCE IT WAS SPECIFIED ONLY ONCE. IN THE SECOND PASS, FOUR ENTRIES WERE BUILT. THIS IS BECAUSE FOUR SUB-DEVICES WERE SPECIFIED. THE " " CONSTRUCT TELLS THE RUNTIME SERVICES TO INCREMENT THE DATA FROM THE FIRST NUMBER TO THE SECOND. IN THIS CASE, SUB-DEVICES 2, 3, 4 AND 5 WERE SPECIFIED. (IF THE SUB DEVICE WERE SPECIFIED BY ADDRESSES, THE INCREMENT WOULD BE BY 2 SINCE ADDRESSES MUST BE ON AN EVEN BOUNDARY.) THE CSR ADDRESSES AND Q-FACTORS FOR THE FOUR ENTRIES ARE ASSUMED TO BE 160000 AND 0 RESPECTIVELY SINCE THEY WERE ONLY SPECIFIED ONCE. THE LAST TWO UNITS ARE SPECIFIED IN THE THIRD PASS.

THE WHOLE PROCESS COULD HAVE BEEN ACCOMPLISHED IN ONE PASS AS SHOWN BELOW.

# UNITS (0) ? 8&lt;CR&gt;

UNIT 1  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 0-7<CR>  
Q FACTOR (0) 0 ? 0,1,0,...,1,1<CR>

## PROGRAM DOCUMENT

AS YOU CAN SEE FROM THIS EXAMPLE, NULL REPLIES (COMMAS ENCLOSING A NULL FIELD) TELL THE RUNTIME SERVICES TO REPEAT THE LAST REPLY.

## 2.8 QUICK START-UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXDP+
2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ (IF THERE IS A CLOCK AND THE QUESTION IS ASKED) QUESTIONS
3. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
4. TYPE "START"
5. ANSWER THE "CHANGE HW" QUESTION WITH "Y"
6. ANSWER ALL THE HARDWARE QUESTIONS
7. ANSWER THE "CHANGE SW" QUESTION WITH "N"

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS AND SOFTWARE PARAMETERS. FOR DEFAULT INFORMATION SEE THE SECTIONS WITHIN THIS DOCUMENT ON FLAGS, AND HARDWARE QUESTIONS.

## 3.0 ERROR INFORMATION

### 3.1 TYPES OF ERROR MESSAGES

THERE ARE THREE LEVELS OF ERROR MESSAGES THAT MAY BE ISSUED BY A DIAGNOSTIC: GENERAL, BASIC AND EXTENDED. GENERAL ERROR MESSAGES ARE ALWAYS PRINTED UNLESS THE "IER" FLAG IS SET (SEE THE FLAGS SECTION OF THIS DOCUMENT).

THE GENERAL ERROR MESSAGE IS OF THE FORM:

```
NAME TYPE NUMBER ON UNIT NUMBER TST NUMBER PC:XXXXXX  
ERROR MESSAGE
```

.WHERE; NAME = DIAGNOSTIC NAME  
TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)  
NUMBER = ERROR NUMBER  
UNIT NUMBER = 0 - N (N IS LAST UNIT IN PTABLE)  
TST NUMBER = TEST AND SUBTEST WHERE ERROR OCCURRED  
PC:XXXXXX = ADDRESS OF ERROR MESSAGE CALL

BASIC ERROR MESSAGES ARE MESSAGES THAT CONTAIN SOME ADDITIONAL INFORMATION ABOUT THE ERROR. THESE ARE ALWAYS PRINTED UNLESS THE "IER" OR "IBR" FLAGS ARE SET (SEE THE FLAGS SECTION OF THIS

## PROGRAM DOCUMENT

DOCUMENT).  
THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL MESSAGE.

EXTENDED ERROR MESSAGES CONTAIN SUPPLEMENTARY ERROR INFORMATION SUCH AS REGISTER CONTENTS OR GOOD/BAD DATA. THESE ARE ALWAYS PRINTED UNLESS THE "IER", "IBR" OR "IXR" FLAGS ARE SET (SEE THE FLAGS SECTION OF THIS DOCUMENT).  
THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

### 3.2 ERROR MESSAGES

This program is intended to provide a go/no go indication of the functionality of DHV11-M boards. To execute the program in this mode the operator can run with the inhibit basic error reporting switch. In this mode the program prints error messages which contain the error message header described above, plus the name of the failing test. For a list of the test names in this program see the test summaries section of this document. An example of such an error message is the following:

```
CVDHB DVC FTL ERR 01603 ON UNIT 02 TST 015 SUB 000 PC: 015244  
DEVICE REGISTER WORD READ/WRITE TEST
```

This error indicates that a fatal error was encountered within the test which tests the read/write capability of the DHV11-M registers.

If the operator requires more extensive error reporting he can run with all error reporting enabled by not using the inhibit reporting switches. The above error message would then become the following:

```
CVDHB DVC FTL ERR 01603 ON UNIT 02 TST 015 SUB 000 PC: 015244  
DEVICE REGISTER WORD READ/WRITE TEST  
BAD BIT(S) IN DEVICE TBUFFAD1 REGISTER FOR LINE 7 (D).  
EXPECTED DATA: 000000 (0).  
ACTUAL DATA: 000023 (0).
```

### 4.0 PERFORMANCE AND PROGRESS REPORTS

AT THE END OF EACH PASS, THE PASS COUNT IS GIVEN ALONG WITH THE TOTAL NUMBER OF ERRORS REPORTED SINCE THE DIAGNOSTIC WAS STARTED. THE "EOP" SWITCH CAN BE USED TO CONTROL HOW OFTEN THE END OF PASS MESSAGE IS PRINTED. FOR FURTHER INFORMATION SEE THE SWITCHES SECTION OF THIS DOCUMENT.

## PROGRAM DOCUMENT

## 5.0 TEST SUMMARIES

The following tests are included within CVDHB:

1. Device register address test - Verifies that the UUT registers will respond with the proper Q-BUS handshaking when accessed. Verifies that the UUT is at the proper address.
2. DMA.START test - Verifies that each DMA.START bit will initiate a DMA TX on a line.
3. DMA.ABORT test - Verifies that the DMA.ABORT bit on each line will stop a DMA transmission and return a TX.ACTION and that the DMA can then be restarted.
4. O.AUTO inactive test - Verifies that the UUT will not respond to incoming XON and XOFF characters when O.AUTO is inactive.
5. O.AUTO active test - Verifies that the UUT respond correctly to incoming XON and XOFF characters when O.AUTO is active.
6. I.AUTO inactive test - Verifies that the UUT will not generate and TX XON or XOFF characters in response to the FIFO conditions if the I.AUTO bit is inactive.
7. I.AUTO active test - Verifies that the UUT will generate and TX XON and XOFF characters in response to the FIFO conditions if the I.AUTO bit is active.
8. FIFO data test - Verifies that the FIFO will hold 256 characters without corrupting data.
9. FIFO 3/4 level inactive test - Verifies that the FIFO 3/4 alarm does not become active until the FIFO becomes 3/4 full.
10. FIFO 3/4 level active test - Verifies that the FIFO 3/4 alarm becomes active, and remains active, when the FIFO is more than 3/4 full.
11. FIFO 3/4 level active/inactive test - Verifies that the FIFO 3/4 alarm, once activated, remains active until the FIFO level is reduced below 1/2.
12. FIFO 1/2 level test - Verifies that FIFO 1/2 level indicator becomes active and remains active as the FIFO level is reduced below the 1/2 full point.
13. BREAK test - Verifies that setting the BREAK bit on any line causes that line to go to a spacing state and that clearing the BREAK bit removes the line from the spacing state.
14. No OVERRUN.ERROR test - Verifies that the UUT will receive the maximum number of characters without causing an overrun error.

## PROGRAM DOCUMENT

15. OVERRUN.ERROR test - Verifies that if more than the maximum number of characters are sent to the UUT overrun errors occur.
16. DTR test - Verifies that changing the UUT LNCTRL DTR bit affects the state of the DTR control line.
17. RTS test - Verifies that changing the UUT LNCTRL RTS bit affects the state of the RTS control line.
18. DSR test - Verifies that DSR status signal correctly reports the state of the looped back DTR control line.
19. RI test - Verifies that RI status signal correctly reports the state of the looped back DTR control line.
20. CTS test - Verifies that CTS status signal correctly reports the state of the looped back RTS control line.
21. DCD test - Verifies that DCD status signal correctly reports the state of the looped back RTS control line.
22. DTR interactions test - Verifies that changing the state of the DTR control signal on any line does not affect the state of any status signals that it is not looped back to.
23. RTS interactions test - Verifies that changing the state of the RTS control signal on any line does not affect the state of any status signals that it is not looped back to.
24. TX lines test - Verifies that each active line has continuity to the correct receive line. This test is only run if H325 or H3277 loopback is specified in the hardware questions dialogue. This test is intended to be used to verify cable and distribution panel data line connections.
25. TX lines interactions test - Verifies that each active TX line does not influence any RX lines (except the one that it is looped back to) or any modem status signals. This test is only run if H325 or H3277 loopback is specified in the hardware questions dialogue.
26. Report BMP codes test - This pseudo test reports the first 32 BMP codes which were discovered in the FIFO during the execution of the other tests. This avoids the interruption of other tests by these codes, if they are not critical to the tests being performed.

## PROGRAM DOCUMENT

## 6.0 EXAMPLE ERROR FREE PASS

The following is an example of an error free pass dialogue:

```
.R CVDHBD0
CVDHBD0.BIC
```

```
DRS
CVDHB-D-0
DHV11-M FUNC TST PART 2
UNIT IS DHV11-M
RESTART ADDR: 147670
DR>STA
```

```
CHANGE HW (L) ? Y
```

```
# UNITS (D) ? 2
```

```
UNIT 0
CSR ADDRESS: (0) 160460 ? +Z
```

```
UNIT 1
CSR ADDRESS: (0) 160460 ? 160040
INTERRUPT VECTOR ADDRESS: (0) 300 ? 320
ACTIVE LINE BIT MAP: (0) 377 ? <CR>
TYPE OF LOOPBACK (1=INTERNAL, 2=H3277, 3=H325): (0) 2 ? 1
INTERRUPT BR LEVEL: (0) 4? <CR>
```

```
CHANGE SW (L) ? Y
```

```
REPORT UNIT NUMBER AS EACH UNIT IS TESTED: (L) Y ? <CR>
NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE: (D) 0 ? 4
```

```
TESTING UNIT : 0(D)
```

```
TESTING UNIT : 1(D)
```

```
CVDHB EOP 1
0 CUMULATIVE ERRORS
```

```
TESTING UNIT : 0(D)
```

```
+C
DR> EXIT
```

ε

1098  
1099

```
.LIST SEQ,LOC,BIN,MEB
.NLIST CND
```

PROGRAM DOCUMENT

```

1101 ;*****
1102 ;
1103 ;           V D H B . P H D
1104 ;
1105 ;*****
1106
1107
1108
1109 .SBTTL Program Header
1110
1111
1112 .MCALL SVC
1113 000000 SVC ; INITIALIZE SUPERVISOR MACROS
1114
1115 ;*****
1116 ; IF STRUCTURED MACROS ARE TO BE USED, ADD ".MCALL STRUCT" AND "STRUCT"
1117 ; TO INITIALIZE THE STRUCTURED MACROS.
1118
1119 000001 SVCINS= 1 ; LIST INSTRUCTIONS, SHIFTED RIGHT
1120 000001 SVCTST= 1 ; LIST TEST TAGS, SHIFTED RIGHT
1121 000001 SVCSUB= 1 ; LIST SUBTEST TAGS, SHIFTED RIGHT
1122 000001 SVCGBL= 1 ; LIST GLOBAL TAGS, SHIFTED RIGHT
1123 000001 SVCTAG= 1 ; LIST OTHER TAGS, SHIFTED RIGHT
1124
1125 ; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
1126 ; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
1127 ; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
1128 ; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.
1129 ;*****
1130
1131 000000 .ENABL ABS
1132 ;.ENABL AMA
1133 002000 = 2000
1134
1135 002000 BGNMOD
1136
1137 ;**
1138 ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
1139 ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
1140 ;--
1141
1142 002000 POINTER BGNRPT,BGNSW,BGNSFT,BGNDU,ERRTBL
1143
1160
1161 002000 HEADER CVDHB,D,0,22,0,PRI07
                                L$NAME::
                                .ASCII /C/
                                .ASCII /V/
                                .ASCII /D/
                                .ASCII /H/
                                .ASCII /B/
                                .BYTE 0
                                .BYTE 0
                                .BYTE 0
                                L$REV::
                                .ASCII /D/
                                L$DEPO::

```

Program Header

002011 060  
 002012  
 002012 000000  
 002014  
 002014 000022  
 002016  
 002016 035724  
 002020  
 002020 036206  
 002022  
 002022 002212  
 002024  
 002024 002224  
 002026  
 002026 036446  
 002030  
 002030 000000  
 002032  
 002032 000000  
 002034  
 002034 000000  
 002036  
 002036 000000  
 002040  
 002040 002124  
 002042  
 002042 000340  
 002044  
 002044 000000  
 002046  
 002046 000000  
 002050  
 002050 004  
 002051 000  
 002052  
 002052 000000  
 002054 000000  
 002056  
 002056 000000  
 002060  
 002060 004154  
 002062  
 002062 017064  
 002064  
 002064 000000  
 002066  
 002066 000000  
 002070  
 002070 000000  
 002072  
 002072 020012  
 002074  
 002074 000000  
 002076  
 002076 004164  
 002100  
 002100 104035

.ASCII /0/  
 L\$UNIT:: .WORD 0  
 L\$TIML:: .WORD 22  
 L\$HPCP:: .WORD L\$HARD  
 L\$SPCP:: .WORD L\$SOFT  
 L\$HPTP:: .WORD L\$HW  
 L\$SPTP:: .WORD L\$SW  
 L\$LADP:: .WORD L\$LAST  
 L\$STA:: .WORD 0  
 L\$CO:: .WORD 0  
 L\$DTYP:: .WORD 0  
 L\$APT:: .WORD 0  
 L\$DTP:: .WORD 0  
 L\$DISPATCH  
 L\$PRIO:: .WORD PRI07  
 L\$ENVI:: .WORD 0  
 L\$EXP1:: .WORD 0  
 L\$MREV:: .WORD 0  
 .BYTE C\$REVISION  
 .BYTE C\$EDIT  
 L\$EF:: .WORD 0  
 .WORD 0  
 L\$SPC:: .WORD 0  
 L\$DEVP:: .WORD L\$DVTYP  
 L\$REPP:: .WORD L\$RPT  
 L\$EXP4:: .WORD 0  
 L\$EXP5:: .WORD 0  
 L\$AUT:: .WORD 0  
 L\$DUT:: .WORD L\$DU  
 L\$LUN:: .WORD 0  
 L\$DESP:: .WORD L\$DESC  
 L\$LOAD:: EMT E\$LOAD



Program Header

002102  
002102 004104  
002104  
002104 017100  
002106  
002106 017754  
002110  
002110 017752  
002112  
002112 017072  
002114  
002114 000000  
002116  
002116 000000  
002120  
002120 000000

1162

L\$ETP:: .WORD L\$ERRTBL  
L\$ICP:: .WORD L\$INIT  
L\$CCP:: .WORD L\$CLEAN  
L\$ACP:: .WORD L\$AUTO  
L\$PRT:: .WORD L\$PROT  
L\$TEST:: .WORD 0  
L\$DLY:: .WORD 0  
L\$HIME:: .WORD 0

DISPATCH TABLE

```

1174
1175
1176
1177
1178
1179
1180
1181 002122
      002122 000032
      002124
      002124 020126
      002126 020416
      002130 021004
      002132 021424
      002134 022164
      002136 022724
      002140 023320
      002142 023742
      002144 024232
      002146 024534
      002150 025226
      002152 025716
      002154 026362
      002156 027006
      002160 027506
      002162 030434
      002164 031134
      002166 031634
      002170 032250
      002172 032664
      002174 033300
      002176 033714
      002200 034310
      002202 034704
      002204 035222
      002206 035642

```

.SBTTL DISPATCH TABLE

```

; **
; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
; --

```

DISPATCH 26

```

      .WORD 26
L$DISPATCH:
      .WORD T1
      .WORD T2
      .WORD T3
      .WORD T4
      .WORD T5
      .WORD T6
      .WORD T7
      .WORD T8
      .WORD T9
      .WORD T10
      .WORD T11
      .WORD T12
      .WORD T13
      .WORD T14
      .WORD T15
      .WORD T16
      .WORD T17
      .WORD T18
      .WORD T19
      .WORD T20
      .WORD T21
      .WORD T22
      .WORD T23
      .WORD T24
      .WORD T25
      .WORD T26

```

1182

DISPATCH TABLE

```

1190
1191 ;*****
1192 ;
1193 ;           V D H B . D H T
1194 ;
1195 ;*****
*****
1196
1197
1198
1199 .SBTTL  DEFAULT HARDWARE P-TABLE
1200
1201 ;**
1202 ; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
1203 ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
1204 ; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,
1205 ; AND IS USED AS A "TEMPLATE" FOR BUILDING THE P-TABLES.
1206 ;--
1207
1208 002210      BGNHW  DFPTBL
      002210      000004
      002212
      002212
      L$HW::      .WORD  L10000 L$HW/2
      DFPTBL::

1209
1210 002212      160460      .WORD  160460 ;Default CSR Address
1211 002214      000300      .WORD  300    ;Default Vector Address
1212 002216      000377      .WORD  MAPLNS ;Default Active lines bit map
1213 002220      002       .BYTE  2     ;Default Loopback mode
1214 002221      004       .BYTE  4     ;Default BR Level
1215
1216 002222      ENDHW
      002222
      L10000:

```

DEFAULT HARDWARE P-TABLE

```

1218
1219 ;*****
1220 ;
1221 ;           VDHB.SWT
1222 ;
1223 ;*****
1224
1225
1226
1227 .SBTTL  SOFTWARE P TABLE
1228
1229 ;**
1230 ; THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
1231 ; PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
1232 ; SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
1233 ; AT RUN TIME.
1234 ;--
1235
1236 002222          BGNSW  SFPTBL
      002222 000002
      002224
      002224
                                L$SW:: .WORD  L10001-L$SW/2
                                SFPTBL::
1237
1238 002224 000020          OPTION:: .WORD  20          ;bit map of program control flags
1239 002226 000000          NDERPT:: .WORD  0           ;Default number of individual data errors to rpt.
1240
1241 002230          ENDSW
      002230
                                L10001:

```

SOFTWARE P-TABLE

```

1243
1244 ;*****
1245 ;
1246 ;           VDHB.EQU
1247 ;
1248 ;*****
1249
1250
1251 .SBTTL GLOBAL EQUATES SECTION
1252
1262
1263
1264
1265 ;**
1266 ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
1267 ; ARE USED IN MORE THAN ONE TEST.
1268 ;-
1269
1270         000010          NUMLNS==10          ;NUMBER OF LINES ON DHV11 IS 8.
1271         000377          MAPLNS==377         ;BIT MAP OF LINES ON DHV11.
1272
1273 ;***** DEVICE REGISTER OFFSETS FROM THE CSR'S ADDRESS *****
1274         000000          CSRO==0             ;CSR REGISTER OFFSET FROM THE CSR ADDRESS
1275         000002          RBUFO==2            ;RECEIVE REGISTER OFFSET FROM THE CSR ADDRESS
1276         000002          TXCHRO==2           ;TRANSMIT REGISTER OFFSET FROM THE CSR ADDRESS
1277         000004          LPRO==4             ;LINE PARAMETER REGISTER OFFSET FROM THE CSR ADDRESS
1278         000006          STATO==6            ;STATUS REGISTER OFFSET FROM THE CSR ADDRESS
1279         000010          LNCTRO==10          ;LINE CONTROL REGISTER OFFSET FROM THE CSR ADDRESS
1280         000012          TXAD10==12          ;TRANSMIT ADDRESS 1 REGISTER OFFSET FROM THE CSR ADDRESS
1281         000014          TXAD20==14          ;TRANSMIT ADDRESS 2 REGISTER OFFSET FROM THE CSR ADDRESS
1282         000016          TXBFCO==16          ;TRANSMIT COUNT REGISTER OFFSET FROM THE CSR ADDRESS
1283
1284 ;***** EQUATES USED WITH RESPECT TO THE RX BUFFER *****
1285         000020          RXBETX==16.         ;LEVEL OF RX BUFFER AT WHICH TO RE-ENABLE TRANSMISSION.
1286         000030          RXBDTX==24.         ;LEVEL OF RX BUFFER AT WHICH TO DISABLE TRANSMISSION.
1287         000100          RXBFUL==64.         ;TOTAL CHARACTER CAPACITY OF THE RX BUFFER.
1288
1289
1304 002230          EQUALS
;
; BIT DIFINITIONS
;
100000          BIT15== 100000
040000          BIT14== 40000
020000          BIT13== 20000
010000          BIT12== 10000
004000          BIT11== 4000
002000          BIT10== 2000
001000          BIT09== 1000
000400          BIT08== 400
000200          BIT07== 200
000100          BIT06== 100
000040          BIT05== 40
000020          BIT04== 20
000010          BIT03== 10
000004          BIT02== 4
000002          BIT01== 2

```

GLOBAL EQUATES SECTION

```

000001      BIT00== 1
;
001000      BIT9==  BIT09
000400      BIT8==  BIT08
000200      BIT7==  BIT07
000100      BIT6==  BIT06
000040      BIT5==  BIT05
000020      BIT4==  BIT04
000010      BIT3==  BIT03
000004      BIT2==  BIT02
000002      BIT1==  BIT01
000001      BIT0==  BIT00
;
; EVENT FLAG DEFINITIONS
; EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
;
; BIT POSITION IN SECOND STATUS WORD
000040      EF.START==      32.      ; (100000) START COMMAND WAS ISSUED
000037      EF.RESTART==    31.      ; (040000) RESTART COMMAND WAS ISSUED
000036      EF.CONTINUE==   30.      ; (020000) CONTINUE COMMAND WAS ISSUED
000035      EF.NEW==        29.      ; (010000) A NEW PASS HAS BEEN STARTED
000034      EF.PWR==        28.      ; (004000) A POWER-FAIL/POWER-UP OCCURRED
;
; PRIORITY LEVEL DEFINITIONS
;
000340      PRI07== 340
000300      PRI06== 300
000240      PRI05== 240
000200      PRI04== 200
000140      PRI03== 140
000100      PRI02== 100
000040      PRI01== 40
000000      PRI00== 0
;
; OPERATOR FLAG BITS
;
000004      EVL==      4
000010      LOT==     10
000020      ADR==     20
000040      IDU==     40
000100      ISR==    100
000200      UAM==    200
000400      BOE==    400
001000      PNT==   1000
002000      PRI==   2000
004000      IXE==   4000
010000      IBE==  10000
020000      IER==  20000
040000      LOE==  40000
100000      HOE== 100000

```

GLOBAL EQUATES SECTION

1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327 002230 000300  
1328 002232 000304  
1329 002234 000377  
1330 002236 000  
1331 002237 004  
1332 002240 000000  
1333  
1334  
1335  
1336  
1337 002242  
1338 002242 160000  
1339 002244 160002  
1340 002246 160004  
1341 002250 160006  
1342 002252 160010  
1343 002254 160012  
1344 002256 160014  
1345 002260 160016  
1346  
1347  
1348  
1349  
1350 002262 000000  
1351 002264 000000  
1352 002266 000001  
1353 002270 000000  
1354 002272 031463  
1355 002274 146314  
1356 002276 000000  
1357 002300 000000  
1358 002302 000000  
1359 002304 000000  
1360 002306 000000  
1361 002310 000000  
1362 002312 000000  
1363 002314 000000

```
*****  
: V D H B . G D T  
:*****
```

.SBTTL GLOBAL DATA SECTION

```
;;  
; THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED  
; IN MORE THAN ONE TEST.  
;--
```

```
*****  
; Unit Variable Area  
;*****
```

```
RXVECA:: .WORD 300 ;RX VECTOR ADDRESS.  
TXVECA:: .WORD 304 ;TX VECTOR ADDRESS.  
ACTLNS:: .WORD 377 ;ACTIVE LINE BIT MAP.  
LOPBCK:: .BYTE 0 ;LOOPBACK MODE  
BRLEVL:: .BYTE 4 ;INTERRUPT BUS REQUEST LEVEL  
UNITN:: .WORD 0 ;UNIT NUMBER.
```

```
*****  
; Device Register Address Table  
;*****
```

```
DRADRT::  
CSRA:: .WORD 160000 ;DHV11-M CSR ADDRESS  
TXCHA:: RBUFA:: .WORD 160002 ;DHV11-M RECEIVE/TRANSMIT BUFFER ADDRESS  
LPRA:: .WORD 160004 ;DHV11-M LINE PARAMETER REGISTER ADDRESS  
STATA:: .WORD 160006 ;DHV11-M STATUS REGISTER ADDRESS  
LNCTRA:: .WORD 160010 ;DHV11-M LINE CONTROL REGISTER ADDRESS  
TXAD1A:: .WORD 160012 ;DHV11-M TRANSMIT BUFFER 1 REGISTER ADDRESS  
TXAD2A:: .WORD 160014 ;DHV11-M TRANSMIT BUFFER 2 REGISTER ADDRESS  
TXBFCA:: .WORD 160016 ;DHV11-M TRANSMIT BUFFER COUNT REGISTER ADDRESS
```

```
*****  
; Assorted global variables:  
;*****
```

```
BUFPTR:: .WORD 0 ;STORAGE FOR RECEIVE CHARACTER BUFFER POINTER.  
CTRLCF:: .WORD 0 ;STORAGE FOR THE CONTROL-C FLAG.  
TSTNUM:: .WORD 1 ;STORAGE FOR THE TEST NUMBER.  
IESTAT:: .WORD 0 ;STORAGE FOR STATES OF THE DUT INT ENABLE BITS.  
LGRP1M:: .WORD 31463 ;BIT MAP OF LINES IN LINE GROUP I.  
LGRP2M:: .WORD 146314 ;BIT MAP OF LINES IN LINE GROUP II.  
PASCNT:: .WORD 0 ;STO'G FOR PASS COUNT USED IN ROM VERSION# TST.  
RXINTC:: .WORD 0 ;STORAGE FOR RECEIVER INTERRUPT FLAGS.  
RXINTF:: .WORD 0 ;STORAGE FOR RECEIVER INTERRUPT FLAGS.  
TXINTC:: .WORD 0 ;STORAGE FOR TRANSMIT INTERRUPT COUNT.  
TXINTF:: .WORD 0 ;STORAGE FOR TRANSMIT INTERRUPT FLAGS.  
TP4VEC:: .WORD 0 ;STORAGE FOR THE NORMAL 004 TRAP VECTOR.  
TP4FLG:: .WORD 0 ;FLAGS SET WHEN AN EXPECTED 004 TRAP OCCURS.  
WORD1:: .WORD 0 ;LOCATION FOR PASSING INDIRECT PARAMETERS.
```

GLOBAL DATA SECTION

```

1364 ;*****
1365 ; Line Time Clock variables and storage.
1366 ;*****
1367 002316 177546 CLKCSR:: .WORD 177546 ;CSR ADDRESS OF THE LTC.
1368 002320 000300 CLKBRL:: .WORD PRI06 ;INTERRUPT PRIORITY LEVEL OF THE LTC.
1369 002322 000100 CLKVEC:: .WORD 100 ;INTERRUPT VECTOR ADDRESS OF THE LTC.
1370 002324 000074 CLKHRZ:: .WORD 60. ;INTERRUPT FREQUENCY OF THE LTC.
1371 002326 000000 TIMER1:: .WORD 0 ;HARDWARE CLOCK COUNTER #1.
1372 002330 000000 TIMER2:: .WORD 0 ;HARDWARE CLOCK COUNTER #2.
1373 002332 000170 TIMER3:: .WORD 120. ;HARDWARE BREAK COUNTER LOCATION.
1374 002334 000170 BCOUNT:: .WORD 120. ;BREAK COUNT VALUE IN CLOCK TICKS.
1375 002336 000021 MSTICK:: .WORD 17. ;NUMBER OF MILLI-SECONDS PER LTC TICK.
1376 002340 000062 MSLCNT:: .WORD 62 ;LOOP COUNT (USED BY MSLOOP) TO DELAY 1 MS.
1377
1378 ;*****
1379 ; Memory Management Variables and Flags.
1380 ;*****
1381 002342 177572 MMSRO:: .WORD 177572 ;ADDRESS OF MEM MGT STATUS REGISTER #0.
1382 002344 000000 MMPRES:: .WORD 0 ;MEM MGT PRESENT FLAG (0 IF MM NOT PRESENT).
1383 002346 000000 MMENAB:: .WORD 0 ;MEM MGT ENABLED FLAG (0 IF MM NOT ENABLED).
1384
1385 002350 PARATB:: ;BASE OF MEM MGT PAR ADDRESS TABLE.
1386 002350 172340 PAR0A:: .WORD 172340 ;ADDRESS OF MEM MGT PAR #0.
1387 002352 172342 PAR1A:: .WORD 172342 ;ADDRESS OF MEM MGT PAR #1.
1388 002354 172344 PAR2A:: .WORD 172344 ;ADDRESS OF MEM MGT PAR #2.
1389 002356 172346 PAR3A:: .WORD 172346 ;ADDRESS OF MEM MGT PAR #3.
1390 002360 172350 PAR4A:: .WORD 172350 ;ADDRESS OF MEM MGT PAR #4.
1391 002362 172352 PAR5A:: .WORD 172352 ;ADDRESS OF MEM MGT PAR #5.
1392 002364 172354 PAR6A:: .WORD 172354 ;ADDRESS OF MEM MGT PAR #6.
1393 002366 172356 PAR7A:: .WORD 172356 ;ADDRESS OF MEM MGT PAR #7.
1394 002370
1395 PARATE:: ;END OF PAR ADDRESS TABLE.
1396 ;*****
1397 ; Table of words with corresponding bit set for generation of bit maps.
1398 ;*****
1398 002370 000001 BITTBL:: .WORD 1 ;BIT 0 SET.
1399 002372 000002 .WORD 2 ;BIT 1 SET.
1400 002374 000004 .WORD 4 ;BIT 2 SET.
1401 002376 000010 .WORD 10 ;BIT 3 SET.
1402 002400 000020 .WORD 20 ;BIT 4 SET.
1403 002402 000040 .WORD 40 ;BIT 5 SET.
1404 002404 000100 .WORD 100 ;BIT 6 SET.
1405 002406 000200 .WORD 200 ;BIT 7 SET.
1406 002410 000400 .WORD 400 ;BIT 8 SET.
1407 002412 001000 .WORD 1000 ;BIT 9 SET.
1408 002414 002000 .WORD 2000 ;BIT 10 SET.
1409 002416 004000 .WORD 4000 ;BIT 11 SET.
1410 002420 010000 .WORD 10000 ;BIT 12 SET.
1411 002422 020000 .WORD 20000 ;BIT 13 SET.
1412 002424 040000 .WORD 40000 ;BIT 14 SET.
1413 002426 100000 .WORD 100000 ;BIT 15 SET.
1414
1415 ;*****
1416 ;* GPR Save Areas Zero and One.
1417 ;*****
1418 002430 GPRSOB:: ;BASE OF GPR SAVE AREA NUMBER ZERO.
1419 002430 000000 .WORD 0 ;WORD 1, STORAGE FOR R1.
1420 002432 000000 .WORD 0 ;WORD 2, STORAGE FOR R2.

```



GLOBAL DATA SECTION

```

1421 002434 000000      .WORD 0      ;WORD 3, STORAGE FOR R3.
1422 002436 000000      .WORD 0      ;WORD 4, STORAGE FOR R4.
1423 002440 000000      .WORD 0      ;WORD 5, STORAGE FOR R5.
1424
1425 ;*****
;      Storage area for the BMP code queue.
1426 ;*****
1427 002442 000000  BMPCQP:: .WORD 0      ;POINTER USED TO ACCESS THE NEXT CELL IN QUE.
1428 002444      BMPCQB:: .BLKW 64.    ;STORAGE FOR 32 CELLS, TEST# PLUS BMP CODE.
1429 002644      BMPCQE::      ;LAST ADDRESS PLUS 2 OF THE BMP CODE QUEUE.
1430 ;*****
1431 ;      Storage area for the contents of the DUT STAT register states.
1432 ;*****
1433 002644      STSTB::      ;BASE OF DUT STAT STORAGE TABLE.
1434 002644 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 0.
1435 002646 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 1.
1436 002650 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 2.
1437 002652 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 3.
1438 002654 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 4.
1439 002656 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 5.
1440 002660 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 6.
1441 002662 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 7.
1442 002664 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 8.
1443 002666 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 9.
1444 002670 000000      .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 10.
1445 002672 000000     .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 11.
1446 002674 000000     .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 12.
1447 002676 000000     .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 13.
1448 002700 000000     .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 14.
1449 002702 000000     .WORD 0      ;STORAGE FOR STAT REGISTER FOR LINE 15.
1450 002704      STSTE::      ;END OF DUT STAT STORAGE TABLE.
1451 ;*****
1452 ;      General table and buffer area--513 words.
1453 ;*****
1454 002704      BUFBAS::      ;BASE OF MEMORY BUFFER.
1455 002704      ERLTBL:: .BLKW 128.  ;FIRST HALF OF GENERAL TABLE OR BUFFER.
1456 003304      BUF3QT:: .BLKW 64.    ;SECOND HALF OF GENERAL TABLE OR BUFFER.
1457 003504      BUFEND::      ;LAST QUARTER OF THE BUFFER AREA.
1458 003704      ENDETBT:: .BLKW 16.   ;BUFFER OVERFLOW SPACE.
1459 003704
1460 ;*****
1461 ;      Reception table of counters
1462 ;*****
1463 003744      RXCNTB:: .BLKW 16.    ;RECEPTION CHARACTER COUNTERS TABLE.
1464 ;*****
1465 ;* Table for storage of RX/TX line number associations.
1466 ;* The associations are stored as line number times 2 for use as offsets
1467 ;* when accessing a table of words.
1468 ;* NOTE: Do not write a non-zero value into the upper byte of any entry.
1469 ;*****
1470 004004      TXRXLB::      ;BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
1471 004004 000000      .WORD 0      ;TX/RX LINE OFFSET FOR RX/TX LINE 0.
1472 004006 000002      .WORD 2.      ;TX/RX LINE OFFSET FOR RX/TX LINE 1.
1473 004010 000004      .WORD 4.      ;TX/RX LINE OFFSET FOR RX/TX LINE 2.
1474 004012 000006      .WORD 6.      ;TX/RX LINE OFFSET FOR RX/TX LINE 3.
1475 004014 000010      .WORD 8.      ;TX/RX LINE OFFSET FOR RX/TX LINE 4.
1476 004016 000012      .WORD 10.     ;TX/RX LINE OFFSET FOR RX/TX LINE 5.
1477 004020 000014      .WORD 12.     ;TX/RX LINE OFFSET FOR RX/TX LINE 6.

```

GLOBAL DATA SECTION

```

1478 004022 000016 .WORD 14. ;TX/RX LINE OFFSET FOR RX/TX LINE 7.
1479 004024 000020 .WORD 16. ;TX/RX LINE OFFSET FOR RX/TX LINE 8.
1480 004026 000022 .WORD 18. ;TX/RX LINE OFFSET FOR RX/TX LINE 9.
1481 004030 000024 .WORD 20. ;TX/RX LINE OFFSET FOR RX/TX LINE 10.
1482 004032 000026 .WORD 22. ;TX/RX LINE OFFSET FOR RX/TX LINE 11.
1483 004034 000030 .WORD 24. ;TX/RX LINE OFFSET FOR RX/TX LINE 12.
1484 004036 000032 .WORD 26. ;TX/RX LINE OFFSET FOR RX/TX LINE 13.
1485 004040 000034 .WORD 28. ;TX/RX LINE OFFSET FOR RX/TX LINE 14.
1486 004042 000036 .WORD 30. ;TX/RX LINE OFFSET FOR RX/TX LINE 15.
1487 004044 TXRXLE:: ;END OF TX/RX LINE NUMBER ASSOCIATION TABLE.
1488 .EVEN ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
1489 ;*****
1490 ;* Table for storage of RX/TX line number associations.
1491 ;* The associations are stored as line numbers which can be used as such or
1492 ;* as offsets when accessing a table of bytes.
1493 ;*****
1494 004044 TXRLNB:: ;BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
1495 004044 000 .BYTE 0 ;TX/RX LINE FOR RX/TX LINE 0.
1496 004045 001 .BYTE 1. ;TX/RX LINE FOR RX/TX LINE 1.
1497 004046 002 .BYTE 2. ;TX/RX LINE FOR RX/TX LINE 2.
1498 004047 003 .BYTE 3. ;TX/RX LINE FOR RX/TX LINE 3.
1499 004050 004 .BYTE 4. ;TX/RX LINE FOR RX/TX LINE 4.
1500 004051 005 .BYTE 5. ;TX/RX LINE FOR RX/TX LINE 5.
1501 004052 006 .BYTE 6. ;TX/RX LINE FOR RX/TX LINE 6.
1502 004053 007 .BYTE 7. ;TX/RX LINE FOR RX/TX LINE 7.
1503 004054 010 .BYTE 8. ;TX/RX LINE FOR RX/TX LINE 8.
1504 004055 011 .BYTE 9. ;TX/RX LINE FOR RX/TX LINE 9.
1505 004056 012 .BYTE 10. ;TX/RX LINE FOR RX/TX LINE 10.
1506 004057 013 .BYTE 11. ;TX/RX LINE FOR RX/TX LINE 11.
1507 004060 014 .BYTE 12. ;TX/RX LINE FOR RX/TX LINE 12.
1508 004061 015 .BYTE 13. ;TX/RX LINE FOR RX/TX LINE 13.
1509 004062 016 .BYTE 14. ;TX/RX LINE FOR RX/TX LINE 14.
1510 004063 017 .BYTE 15. ;TX/RX LINE FOR RX/TX LINE 15.
1511 004064 TXRLNE:: ;END OF TX/RX LINE NUMBER ASSOCIATION TABLE.
1512 .EVEN ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
1513 ;*****
1514 ;* Table of TX/RX line number associations in staggered loopback.
1515 ;* The associations are stored as line number times 2 for use as offsets
1516 ;* when accessing a table of words.
1517 ;* This is a table of data for reading only. Use to load the above table
1518 ;* NOTE: Must convert from BYTES to WORDS when loading above table.
1519 ;*****
1520 004064 STGTRB:: ;BASE OF STAGGERED TX/RX LINE NUMBER TABLE.
1521 004064 004 .BYTE 4. ;TX/RX LINE OFFSET FOR RX/TX LINE 0.
1522 004065 006 .BYTE 6. ;TX/RX LINE OFFSET FOR RX/TX LINE 1.
1523 004066 000 .BYTE 0. ;TX/RX LINE OFFSET FOR RX/TX LINE 2.
1524 004067 002 .BYTE 2. ;TX/RX LINE OFFSET FOR RX/TX LINE 3.
1525 004070 014 .BYTE 12. ;TX/RX LINE OFFSET FOR RX/TX LINE 4.
1526 004071 016 .BYTE 14. ;TX/RX LINE OFFSET FOR RX/TX LINE 5.
1527 004072 010 .BYTE 8. ;TX/RX LINE OFFSET FOR RX/TX LINE 6.
1528 004073 012 .BYTE 10. ;TX/RX LINE OFFSET FOR RX/TX LINE 7.
1529 004074 024 .BYTE 20. ;TX/RX LINE OFFSET FOR RX/TX LINE 8.
1530 004075 026 .BYTE 22. ;TX/RX LINE OFFSET FOR RX/TX LINE 9.
1531 004076 020 .BYTE 16. ;TX/RX LINE OFFSET FOR RX/TX LINE 10.
1532 004077 022 .BYTE 18. ;TX/RX LINE OFFSET FOR RX/TX LINE 11.
1533 004100 034 .BYTE 28. ;TX/RX LINE OFFSET FOR RX/TX LINE 12.
1534 004101 036 .BYTE 30. ;TX/RX LINE OFFSET FOR RX/TX LINE 13.

```

GLOBAL DATA SECTION

```

1535 004102      030      .BYTE 24.      ;TX/RX LINE OFFSET FOR RX/TX LINE 14.
1536 004103      032      .BYTE 26.      ;TX/RX LINE OFFSET FOR RX/TX LINE 15.
1537              .EVEN      ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
1550 004104      ERRTBL      L$ERRTBL::
      004104      000000      ERRRYP::      .WORD 0
      004106      000000      ERRNBR::      .WORD 0
      004110      000000      ERRMSG::      .WORD 0
      004112      000000      ERRBLK::      .WORD 0
1551
1552              .EVEN

```

GPR HANDLING ROUTINES FOR SUBROUTINE CALLS.

```

1554 .SBTTL GPR HANDLING ROUTINES FOR SUBROUTINE CALLS.
1555 ;*****
1556 ;* There are 4 routines and macro definitions used for the handling of
1557 ;* GPR values during subroutine calls within this program. The four
1558 ;* routines/macro calls have the following names:
1559 ;*
1560 ;* SAVE - Macro definition used at the beginning of a subroutine to
1561 ;* save the GPR contents for later restoration.
1562 ;* PASS - Macro definition used at the end of a subroutine to restore
1563 ;* the previously saved GPR contents and to leave the contents
1564 ;* of the specified GPR(s) intact (NOT restored).
1565 ;* PREG05 - Subroutine which is called from the SAVE and PASS macro
1566 ;* expansions which actually performs the actions on the GPRs.
1567 ;*
1568 ;* During a subroutine which uses these GPR save routines the values
1569 ;* of the GPRs are stored on the stack in the following stack frame:
1570 ;*
1571 ;* SP -> RET PC INTO PREG05 ROUTINE.
1572 ;* SP+2 -> GPR R0 CONTENTS.
1573 ;* SP+4 -> GPR R1 CONTENTS.
1574 ;* SP+6 -> GPR R2 CONTENTS.
1575 ;* SP+8 -> GPR R3 CONTENTS.
1576 ;* SP+10 -> GPR R4 CONTENTS.
1577 ;* SP+12 -> GPR R5 CONTENTS.
1578 ;* SP+14 -> RET PC INTO CALLER OF SUB'TNE WHICH CALLED PREG05.
1579 ;*
1580 ;* Each level of sub'tne calling uses 8 words of stack overhead.
1581 ;* The SAVE and PASS macros can also be used in "straight line code"
1582 ;* to save and restore the GPR values. In any case, after the
1583 ;* issuing of a PASS call the GPRs will be restored to the values
1584 ;* they had prior to the last SAVE call (except for the excepted,
1585 ;* or passed intact, GPRs specified as parameters to the PASS call)
1586 ;* and the SP will also be restored to its condition before the last
1587 ;* SAVE call. The programmer must be sure that the SP has the same
1588 ;* value when the PASS macro is called as it had immediately after
1589 ;* the SAVE macro was called.
1590 ;*****

```

GPR FRAME ACCESS EQUATES

.SBTTL GPR FRAME ACCESS EQUATES

```
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599          000036          LPCSLT==          36          ;Offset for last return PC.  
1600          000016          PCSLOT==          16          ;Offset for return PC.  
1601          000014          R5SLOT==          14          ;Offset for R5.  
1602          000012          R4SLOT==          12          ;Offset for R4.  
1603          000010          R3SLOT==          10          ;Offset for R3.  
1604          000006          R2SLOT==           6          ;Offset for R2.  
1605          000004          R1SLOT==           4          ;Offset for R1.  
1606          000002          ROSLOT==           2          ;Offset for R0.  
  
;***  
;Equates that allow access to the stack frame. These are the  
;offsets into the stack for registers saved during the PREG05  
;routine.  
;---
```

GLOBAL MACRO DEFINITION

- SAVE -

```

1608 .SBTTL GLOBAL MACRO DEFINITION - SAVE -
1609 ;*****
1610 ;* This macro is used at the beginning of a subroutine to save the
1611 ;* contents of the GPRs R0 thru R5.
1612 ;*
1613 ;* INPUTS: SP - Unchanged since subroutine was entered
1614 ;* R5SLOT - Offset to stack slot for R5 (Equated to 14 Octal)
1615 ;*
1616 ;* OUTPUTS: GPR save area on the stack is loaded with the contents of GPRs
1617 ;* TOP OF STACK - Loaded with the return address into PREG05
1618 ;*
1619 ;* CALLING SEQUENCE: SAVE
1620 ;*
1621 ;* COMMENTS: No arguments are allowed.
1622 ;* The PASS macro should be called to restore the GPR values.
1623 ;*
1624 ;* SUBORDINATE ROUTINES CALLED: PREG05.
1625 ;*****
1626
1627 .MACRO SAVE
1628 .LIST JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
1629
1630 .NLIST
1631 .ENDM SAVE

```

GLOBAL MACRO DEFINITION

- PASS -

```

1633 .SBTTL GLOBAL MACRO DEFINITION - PASS -
1634 ;*****
1635 ;* This macro is used in conjunction with the SAVE macro. It is
1636 ;* called at end of a subroutine to pass parameters in GPRs back to the
1637 ;* calling routine by altering the GPR save area on the stack and then
1638 ;* returning to PREG05 to restore the GPRs to their saved values.
1639 ;*
1640 ;* INPUTS: Only allowed ARGUMENTS are "R0" thru "R5".
1641 ;* ROSLOT thru R5SLOT must be equated to their respective GPR save
1642 ;* slot offsets before calling this macro.
1643 ;*
1644 ;* OUTPUTS: The GPR values are put in their respective slots on the stack.
1645 ;*
1646 ;* CALLING SEQUENCE: PASS R0,R1,...
1647 ;*
1648 ;* COMMENTS: Any combination of GPR arguments may be listed in any order.
1649 ;* For example, the following are legal:
1650 ;* PASS R1
1651 ;* PASS R4,R0,R2
1652 ;* The GPRs listed as arguments will be passed intact to the
1653 ;* calling routine, all other GPRs will be restored.
1654 ;* The SP must be at its original value when PASS is called.
1655 ;*
1656 ;* The macro call
1657 ;* PASS R0,R3
1658 ;* expands into the following assembly code:
1659 ;* MOV R0,ROSLOT(SP) ;PUT R0 IN STACK SLOT.
1660 ;* MOV R3,R3SLOT(SP) ;PUT R3 IN STACK SLOT.
1661 ;* JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
1662 ;* In this example GPRs R1, R2, R4, and R5 will be restored to
1663 ;* their values contained in the stack frame and R0 and R3
1664 ;* will be left at their values prior to this PASS call.
1665 ;*
1666 ;* SUBORDINATE ROUTINES CALLED: (PREGRT - Label within PREG05, value on stack.)
1667 ;*****
1668
1669 .MACRO PASS A,B,C,D,E,F
1670 .IRP X,<A,B,C,D,E,F>
1671 .IF NB,X
1672 .LIST
1673 MOV X,X'SLOT(SP) ;PUT X IN STACK SLOT.
1674 .NLIST
1675 .ENDC
1676 .ENDM
1677 .LIST
1678 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
1679 .NLIST
1680 .ENDM PASS

```

GLOBAL SUBROUTINE

- PREG05 -

```

1682 .SBTTL GLOBAL SUBROUTINE - PREG05 -
1683 ;*****
1684 ;* Preserve Registers R0 through R5 for subroutine calls.
1685 ;*
1686 ;* INPUTS: The return address back into the calling routine must be in
1687 ;* GPR R5. (i.e.- Macros use "JSR R5,PREG05".)
1688 ;*
1689 ;* OUTPUTS: Registers R0 through R5 are saved on the stack.
1690 ;*
1691 ;*CALLING SEQUENCE: SAVE ;Macro expansion calls PREG05.
1692 ;* [Subroutine code]...
1693 ;* PASS ;Macro expansion recalls PREG05.
1694 ;*
1695 ;*COMMENTS: This routine is re-entrant.
1696 ;*
1697 ;* Parameters may be passed out of a subroutine by modifying the
1698 ;* register save area on the stack. Use the PASS GPRn macro
1699 ;* to return GPR values intact.
1700 ;* Use the RnSLOT offsets from the SP to pass other parameters.
1701 ;* [Example: MOV VALUE,ROSLOT(SP) ]
1702 ;* Make sure the SP is at its original value when you do this.
1703 ;*
1704 ;*SUBORDINATE ROUTINES CALLED: None.
1705 ;*****
1706
1707 004114 PREG05: ;R5 HAS BEEN LOADED ON THE STACK BY THE SUBROUTINE CALL
1708 004114 010446 MOV R4,-(SP) ;SAVE R4
1709 004116 010346 MOV R3,-(SP) ;SAVE R3
1710 004120 010246 MOV R2,-(SP) ;SAVE R2
1711 004122 010146 MOV R1,-(SP) ;SAVE R1
1712 004124 010046 MOV R0,-(SP) ;SAVE R0
1713 004126 010546 MOV R5,-(SP) ;PUSH RETURN PC ON TOP OF STACK
1714 004130 016605 000014 MOV R5SLOT(SP),R5 ;RESTORE R5 TO VALUE IT HAD BEFORE CALLS
1715
1716 004134 004736 JSR PC,@(SP)+ ;Call the subroutine at the return address
1717 ;from the PREG05 call, putting the present
1718 ;PC on the stack as a return address into
1719 ;this (PREG05) routine.
1720
1721 ;***
1722 ;The following code is executed when the calling routine does a
1723 ;"return" [JSR PC,@(SP)+] using the PC deposited on the stack above.
1724 ;---
1725
1726 004136 012605 PREGRT:: MOV (SP)+,R5 ;Put return PC in R5.
1727 004140 012600 MOV (SP)+,R0 ;Restore R0.
1728 004142 012601 MOV (SP)+,R1 ;Restore R1.
1729 004144 012602 MOV (SP)+,R2 ;Restore R2.
1730 004146 012603 MOV (SP)+,R3 ;Restore R3.
1731 004150 012604 MOV (SP)+,R4 ;Restore R4.
1732
1733 004152 000205 RTS R5 ;Return to the subroutine which called PREG05.
1734 ;restoring R5 in the process.

```



GLOBAL TEXT SECTION

```

1736 .SBTTL GLOBAL TEXT SECTION
1738 :*****
1739 :
1740 :           SKL1.P11
1741 :
1742 :*****

```

```

1746 :**
1747 : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
1748 : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
1749 : MORE THAN ONE TEST.
1750 :--

```

```

1752 :
1753 : NAMES OF DEVICES SUPPORTED BY PROGRAM
1754 :
1755 :           DEVTYP <DHV11-M>

```

```

004154
004154           104       110       126           L$DVTYP::
004154           061       061       055           .ASCIZ  *DHV11-M*
004162           115       000

```

.EVEN

```

1756 :
1762 :
1763 : TEST DESCRIPTION
1764 :
1765 :           DESCRIPT      <DHV11-M FUNC TEST PART 2>

```

```

004164
004164           104       110       126           L$DESC::
004164           061       061       055           .ASCIZ  /DHV11-M FUNC TEST P

```

ART 2/

```

004167           061       061       055
004172           115       040       106
004175           125       116       103
004200           040       124       105
004203           123       124       040
004206           120       101       122
004211           124       040       062
004214           000

```

.EVEN

.EVEN

1766  
1767  
1774

GLOBAL TEXT SECTION

```
1776  
1777 ;*****  
* 1778 ;  
1779 ; VDHB.FMT  
1780 ;  
1781 ;*****  
1782  
1783  
1784  
1785 ;  
1786 ; FORMAT STATEMENTS USED IN PRINT CALLS  
1787 ;  
1788  
1799  
1800
```

GLOBAL TEXT SECTION

```

1809
1810 ;*****
1811 ;
1812 ; VDHB.MSG
1813 ;
1814 ;*****

```

```

**
1815
1816
1817 .NLIST BIN
1818 .SBTTL GLOBAL MESSAGE AREA
1819 ;***** FORMAT STATEMENTS *****
1820 004216 MFUNIT:: .ASCIZ /N%A TESTING UNIT :D4N/
1821 004247 EF0503:: .ASCIZ /T%N/
1822 004254 EF0505:: .ASCIZ /%A %D5%A ILLEGAL INTERRUPTS RECEIVED.%N/
1823 004327 EF1601:: .ASCIZ /%A %T%A ABORTED %N/
1824 004353 EF3001:: .ASCIZ /%A EXPECTED OR CORRECT VALUE: %03%N/
1825 004422 EF3002:: .ASCIZ /%A ACTUAL OR MEASURED VALUE: %03%N/
1826 004471 EF6401:: .ASCIZ /%A %D2%A(D).%N/
1827 004546 EF7801:: .ASCIZ /T%A ON LINE %D2%A DECIMAL.%N/
1828 004604 EF8401:: .ASCIZ /%A %T%A FOR LINE %D2%A(D) AFFECTS OTHER MODEM SIGNALS.%N/
1829 004676 EF9402:: .ASCII /%A CHANGING %T%A FOR LINE %D2%A(D) AFFECTED /
1830 004761 .ASCIZ /T%A FOR LINE %D2%A(D).%N/
1831 005013 EF9001:: .ASCIZ /%A UNEXPECTED %T%A FOUND IN RECEIVE CHAR FIFO:%N/
1832 005075 EF9002:: .ASCIZ /%A CODE IS ASSOCIATED WITH LINE: %D2%N/
1833 005147 EF9003:: .ASCIZ /%A CODE IS: %03%N/
1834 005176 EF9004:: .ASCIZ /%A %T%A VALUE: %03%N/
1835 005226 EF9005:: .ASCIZ /%A %T%A VALUE: NONE%N/
1836 005257 EF9006:: .ASCIZ /%A %T%A %D2%N/
1837 005276 EF9019:: .ASCIZ /%A %T%A %06%N/
1838 005315 EF9301:: .ASCIZ /%A %T%D2%A(D), BMP CODE REPORTED :%03%A(0)%N/
1839 005373 EF9302:: .ASCIZ /%A OVERFLOW OCCURRED (MORE THAN 31 BMP CODES FOUND IN QUEUE)%N/
1840 ;***** MESSAGE AREA *****
1841 005473 EM0103:: .ASCIZ /DEVICE REGISTER ACCESS ERRORS/
1842 005531 EM0525:: .ASCIZ / RX INTERRUPT(S) RECEIVED WITH RX INTERRUPTS DISABLED./
1843 005621 EM0526:: .ASCIZ / TX INTERRUPT(S) RECEIVED WITH TX INTERRUPTS DISABLED./
1844 005711 EM1601:: .ASCIZ /TIMEOUT OCCURRED WAITING FOR MASTER RESET TO CLEAR/
1845 005774 EM4001:: .ASCIZ /DMA_START BIT TEST/
1846 006017 EM4002:: .ASCIZ /DMA_START BIT BAD ON LINE: /
1847 006053 EM4101:: .ASCIZ /DMA_ABORT BIT TEST/
1848 006076 EM4102:: .ASCIZ /DMA_ABORT BIT BAD ON LINE: /
1849 006132 EM4103:: .ASCIZ /DMA_START BIT FOUND SET AFTER DMA ABORTED ON LINE: /
1850 006216 EM4901:: .ASCIZ /OAUTO (INACTIVE) BIT TEST/
1851 006250 EM4902:: .ASCIZ / OAUTO BIT BAD ON LINE: /
1852 006302 EM5001:: .ASCIZ /OAUTO (ACTIVE) BIT TEST/
1853 006332 EM5101:: .ASCIZ /IAUTO (INACTIVE) TEST/
1854 006360 EM5102:: .ASCIZ /IAUTO BIT FOUND SET ON LINE: /
1855 006416 EM5103:: .ASCIZ /IAUTO BIT BAD ON LINE: /
1856 006446 EM5201:: .ASCIZ /IAUTO (ACTIVE) TEST/
1857 006472 EM5202:: .ASCIZ /IAUTO BIT FOUND CLR ON LINE: /
1858 006530 EM5301:: .ASCIZ /FIFO VALID DATA TEST/
1859 006555 EM5302:: .ASCIZ /FIFO BAD DATA FIELD CORRUPTED, TEST USED LINE:/
1860 006634 EM5303:: .ASCIZ /BMP CODE FOUND IN FIFO, TEST INVAILDATED/
1861 006705 EM5401:: .ASCIZ \FIFO 3/4 ALARM (INACTIVE) TEST\
1862 006744 EM5402:: .ASCIZ /FIFO BAD, ALARM SIGNAL DEFECTIVE/
1863 007005 EM5501:: .ASCIZ \FIFO 3/4 ALARM (ACTIVE) TEST\
1864 007042 EM5601:: .ASCIZ \FIFO 3/4 ALARM (ACTIVE/INACTIVE) TEST\
1865 007110 EM5701:: .ASCIZ \FIFO 1/2 LEVEL (ACTIVE/INACTIVE) TEST\

```

## GLOBAL MESSAGE AREA

```

1866 007156 EM6401:: .ASCIZ /BREAK GENERATION TEST /
1867 007205 EM6402:: .ASCIZ / BREAK NOT RECEIVED ON LINE(S):/
1868 007246 EM6601:: .ASCIZ /NO OVERRUN ERROR TEST/
1869 007274 EM6602:: .ASCIZ / OVERRUN ERROR REPORTED WHEN NONE FORCED/
1870 007346 EM6701:: .ASCIZ /OVERRUN ERROR TEST/
1871 007371 EM6702:: .ASCIZ / NO OVERRURN ERROR REPORTED, OVERRUN FORCED/
1872 007446 EM7801:: .ASCIZ /MODEM CONTROL DTR BIT TEST/
1873 007501 EM7802:: .ASCIZ / DTR BIT FAULTY ON LINE:/
1874 007532 EM7901:: .ASCIZ /MODEM CONTROL RTS BIT TEST/
1875 007565 EM7902:: .ASCIZ / RTS BIT FAULTY ON LINE:/
1876 007616 EM8001:: .ASCIZ /DSR MODEM STATUS SIGNAL TEST /
1877 007654 EM8002:: .ASCIZ / DSR MODEM STATUS SIGNAL DEFECTIVE/
1878 007720 EM6101:: .ASCIZ /RI MODEM STATUS SIGNAL TEST /
1879 007755 EM8102:: .ASCIZ / RI MODEM STATUS SIGNAL DEFECTIVE/
1880 010020 EM8201:: .ASCIZ /CTS MODEM STATUS SIGNAL TEST /
1881 010056 EM8202:: .ASCIZ / CTS MODEM STATUS SIGNAL DEFECTIVE/
1882 010122 EM8301:: .ASCIZ /DCD MODEM STATUS SIGNAL TEST /
1883 010160 EM8302:: .ASCIZ / DCD MODEM STATUS SIGNAL DEFECTIVE/
1884 010224 EM8401:: .ASCIZ /DTR MODEM CONTROL SIGNAL INTERACTIONS TEST /
1885 010300 EM8402:: .ASCIZ /DTR/
1886 010304 EM8403:: .ASCIZ /DSR/
1887 010310 EM8404:: .ASCIZ /RI/
1888 010313 EM8405:: .ASCIZ /DCD/
1889 010317 EM8406:: .ASCIZ /CTS/
1890 010323 EM8501:: .ASCIZ /RTS MODEM CONTROL SIGNAL INTERACTIONS TEST /
1891 010377 EM8502:: .ASCIZ /RTS/
1892 010403 EM8601:: .ASCIZ /CABLE, PANEL, LOOPBACK CONNECTOR DATA LINE TEST /
1893 010464 EM8602:: .ASCIZ \TX/RX LINE ERROR ON THE FOLLOWING LOOPED BACK TX LINES:\
1894 010554 EM8701:: .ASCIZ /TX DATA LINE INTERACTIONS TEST /
1895 010614 EM8702:: .ASCIZ /DATA LINE INTERACTIONS/
1896 010643 EM8703:: .ASCIZ /TX/
1897 010646 EM9009:: .ASCIZ /EXPECTED OR CORRECT/
1898 010672 EM9010:: .ASCIZ /ACTUAL OR MEASURED /
1899 010716 EM9017:: .ASCII / FIFO WILL NOT PURGE (DATA.VALID STUCK SET)./
1900 010773 .ASCIZ / REMAINDER OF TEST SKIPPED./
1901 011027 EM9026:: .ASCIZ / LPR CONTENTS: /
1902 011053 EM9104:: .ASCIZ / UNEXPECTED DATA FOUND IN FIFO FROM LINE: /
1903 011127 EM9301:: .ASCIZ /BMP CODE REPORT/
1904 011147 EM9302:: .ASCIZ /BMP CODE FOUND IN TEST /
1905 011177 EM9303:: .ASCIZ /THE LAST BMP CODE WAS FOUND IN TEST /
1906 011244 EM9304:: .ASCIZ /UNEXPECTED BMP CODES FOUND DURING THIS PASS/
1907
1908 .EVEN
1909 .LIST BIN

```

GLOBAL MESSAGE AREA

```

1911
1912      ;*****
1913      :
1914      :                   SKL2.P11
1915      :
1916      ;*****
1917
1918
1919
1920      .SBTTL  GLOBAL ERROR REPORT SECTION
1921
1922      ;++
1923      ; THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
1924      ; USED BY MORE THAN ONE TEST TO OUTPUT ADDITIONAL ERROR INFORMATION.  PRINTB
1925      ; (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
1926      ;--

```

GLOBAL ERROR REPORTING ROUTINE

- ER0101 -

```

1928 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER0101 -
1929 ;*****
1930 ;* This is an error reporting subroutine which prints additional error
1931 ;* information if an error is detected in TEST 1 (Register Address
1932 ;* Access Test). This subroutine reports the type of access (Read or
1933 ;* Write or both) which caused a bus time-out trap (004 trap).
1934 ;* A message indicating that the DHV may be at the wrong Q-bus address
1935 ;* is also printed.
1936 ;*
1937 ;* INPUTS: R5 - Error flag word.
1938 ;* If bit 0 is set, a read error occurred.
1939 ;* If bit 1 is set, a write error occurred.
1940 ;*
1941 ;* OUTPUTS: Messages are printed at the operator console.
1942 ;*
1943 ;* CALLING SEQUENCE: Include the label "ER0101" as the message pointer
1944 ;* parameter in the DRS error report macro call.
1945 ;*
1946 ;* COMMENTS:
1947 ;*
1948 ;* SUBORDINATE ROUTINES USED: None.
1949 ;*****
1950
1951 011320 BGNMSG ER0101
1952 011320 ER0101::
011320 SAVE ;SAVE THE GPR CONTENTS.
011320 004567 172570 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
1953
1954 011324 032705 000001 BIT #BIT0,R5 ;TEST FOR READ ERROR.
1955 011330 001410 BEQ 2$ ;SKIP READ ERROR MSG IF NO READ ERROR.
1956 011332 PRINTB #MSG1 ;PRINT READ ERROR MESSAGE.
011332 012746 011424 MOV #MSG1,-(SP)
011336 012746 000001 MOV #1,-(SP)
011342 010600 MOV SP,R0
011344 104414 TRAP C$PNTB
011346 062706 000004 ADD #4,SP
1957 011352 032705 000002 2$: BIT #BIT1,R5 ;TEST FOR WRITE ERROR.
1958 011356 001410 BEQ 4$ ;SKIP WRITE ERROR MSG IF NO WRITE ERROR.
1959 011360 PRINTB #MSG2 ;PRINT WRITE ERROR MESSAGE.
011360 012746 011502 MOV #MSG2,-(SP)
011364 012746 000001 MOV #1,-(SP)
011370 010600 MOV SP,R0
011372 104414 TRAP C$PNTB
011374 062706 000004 ADD #4,SP
1960 011400 4$: PRINTX #MSG3 ;SUGGEST THAT DHV MAY BE AT WRONG ADDRESS.
011400 012746 011561 MOV #MSG3,-(SP)
011404 012746 000001 MOV #1,-(SP)
011410 010600 MOV SP,R0
011412 104415 TRAP C$PNTX
011414 062706 000004 ADD #4,SP
1961 011420 PASS ;RESTORE THE GPR CONTENTS.
011420 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
1962 011422 ENDMSG
011422 L10002:
011422 104423 TRAP C$MSG
1963
1964 011424 045 101 102 MSG1:: .ASCIZ /*ABUS TIME-OUT TRAP CAUSED BY READ ATTEMPT.*N/

```

GLOBAL ERROR REPORTING ROUTINE

- ER0101 -

	011427	125	123	040	
	011432	124	111	115	
	011435	105	055	117	
	011440	125	124	040	
	011443	124	122	101	
	011446	120	040	103	
	011451	101	125	123	
	011454	105	104	040	
	011457	102	131	040	
	011462	122	105	101	
	011465	104	040	101	
	011470	124	124	105	
	011473	115	120	124	
	011476	056	045	116	
	011501	000			
1965	011502	045	101	102	MSG2:: .ASCIZ /*ABUS TIME OUT TRAP CAUSED BY WRITE ATTEMPT.*N/
	011505	125	123	040	
	011510	124	111	115	
	011513	105	055	117	
	011516	125	124	040	
	011521	124	122	101	
	011524	120	040	103	
	011527	101	125	123	
	011532	105	104	040	
	011535	102	131	040	
	011540	127	122	111	
	011543	124	105	040	
	011546	101	124	124	
	011551	105	115	120	
	011554	124	056	045	
	011557	116	000		
1966	011561	045	101	104	MSG3:: .ASCIZ /*ADHV MAY BE AT THE WRONG Q-BUS ADDRESS.*N*N/
	011564	110	126	040	
	011567	115	101	131	
	011572	040	102	105	
	011575	040	101	124	
	011600	040	124	110	
	011603	105	040	127	
	011606	122	117	116	
	011611	107	040	121	
	011614	055	102	125	
	011617	123	040	101	
	011622	104	104	122	
	011625	105	123	123	
	011630	056	045	116	
	011633	045	116	000	

1967  
1968

.EVEN

GLOBAL ERROR REPORTING ROUTINE

- ER0503 -

```

1970 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER0503 -
1971 ;*****
1972 ;* This is an error reporting subroutine which prints an additional error
1973 ;* message whose address is passed as an input parameter.
1974 ;*
1975 ;* INPUTS: R1 - Address of the message to print.
1976 ;*
1977 ;* OUTPUTS: A messages is printed at the operator console.
1978 ;*
1979 ;* CALLING SEQUENCE: Load the address of the message in R1.
1980 ;* Include the label "ER0503" as the message pointer
1981 ;* parameter in the Diag Super error report macro call.
1982 ;*
1983 ;* COMMENTS: The message is printed as Basic error information.
1984 ;*
1985 ;* SUBORDINATE ROUTINES USED: None.
1986 ;*****
1987
1988 011636 BGNMSG ER0503
1989 011636 ER0503::
1990 011636 PRINTB #EF0503,R1 ;PRINT THE MESSAGE.
1991 011636 010146 MOV R1,(SP)
1992 011640 012746 004247 MOV #EF0503,-(SP)
011644 012746 000002 MOV #2,-(SP)
011650 010600 MOV SP,R0
011652 104414 TRAP C$PNTB
011654 062706 000006 ADD #6,SP
1991
1992 011660 ENDMSG
011660 L10003:
011660 104423 TRAP C$MSG

```



GLOBAL ERROR REPORTING ROUTINE

- ER1603 -

1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014

```

.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER1603 -
;*****
;* This error reporting routine is used to print out a basic error
;* message, along with a message informing the operator which test is
;* about to be aborted.
;*
;* INPUTS: R1 - Contains the address of the message to be printed.
;* ERRMSG - Contains the address of the message that indicates
;* the test that is being performed, eg DMA, BREAK etc.
;*
;* OUTPUTS: Messages are printed at the operators console.
;* "testname TEST ABORTED"
;*
;* CALLING SEQUENCE: Include the label "ER1603" as the message pointer
;* parameter in the DRS error report macro call.
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: None.
;*****

```

2015 011662  
011662  
2016 011662 004567 172226  
2017  
2018 011666  
011666 010146  
011670 012746 004247  
011674 012746 000002  
011700 010600  
011702 104414  
011704 062706 000006  
2019  
2020 011710 016702 172174  
2021 011714  
011714 010246  
011716 012746 004327  
011722 012746 000002  
011726 010600  
011730 104414  
011732 062706 000006  
2022  
2023 011736  
011736 004736  
2024 011740  
011740  
011740 104423

```

BGNMSG ER1603
ER1603::
SAVE JSR R5,PREG05 ;SAVE THE CONTENTS OF THE GPRS.
;CALL REGISTER SAVE SUBRT.
PRINTB #EF0503,R1 ;PRINT BASIC MESSAGE ON OPERATORS CONSOLE.
MOV R1,-(SP)
MOV #EF0503,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTB
ADD #6,SP
MOV ERRMSG,R2 ;GET THE "TEST MESSAGE".
PRINTB #FF1601,R2 ;PRINT "TEST ABORTED" MESSAGE.
MOV R2,-(SP)
MOV #EF1601,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTB
ADD #6,SP
PASS JSR PC,@(SP) ;RESTORE THE CONTENTS OF THE GPRS.
;RETURN TO PREG05 SUBRT.
ENDMSG
L10004: TRAP C$MSG

```

GLOBAL ERROR REPORTING ROUTINE

- ER6401 -

2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066

```

.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER6401 -
;*****
;* This is an error reporting subroutine which prints additional error
;* information after the error message header.
;* This subroutine is passed a GPR containing flags which indicate
;* the line(s) for which the error condition should be reported.
;*
;* INPUTS: R1 Address of the message to be printed by this routine.
;* R5 - Contains the error flags, (1 flag per line).
;*
;* OUTPUTS: Messages are printed at the operator console.
;*
;* CALLING SEQUENCE: Load the address of the message in R1.
;* Include the label "ER6401" as the message pointer
;* parameter in the Diag Super error report macro call.
;*
;* COMMENTS: The output format of this message is:
;* TEXT MESSAGE
;* #nn
;* #nn
;*
;* Where each "#nn" is the number of a line with the error.
;*
;* SUBORDINATE ROUTINES USED: None.
;*****

```

```

011742
011742
011742 004567 172146
011746 005002
011750 012703 000010
011754
011754 010146
011756 012746 004247
011762 012746 000002
011766 010600
011770 104414
011772 062706 000006
011776 000241
012000 006205
012002 103011
012004
012004 010246
012006 012746 004471
012012 012746 000002
012016 010600
012020 104414
012022 062706 000006
012026 005202
012030 020302
012032 001362
012034
012034 004736
012036
012036

```

```

BGNMSG ER6401
ER6401::
SAVE ;SAVE THE CONTENTS OF THE GPRS.
R5,PREG05 ;CALL REGISTER SAVE SUBRT.
JSR

CLR R2 ;CLEAR LINE NUMBER TO ZERO.
MOV #NUMLNS,R3 ;SET UP MAX LINE COUNT.
PRINTB #EF0503,R1 ;PRINT MESSAGE.

MOV R1,-(SP)
MOV #EF0503,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTB
ADD #6,SP

2$: CLC ;CLEAR CARRY.
ASR R5 ;SHIFT FLAG OUT INTO CARRY BIT.
BCC 4$ ;SKIP ERROR REPORT IF CLEAR.
PRINTB #EF6401,R2 ;PRINT MESSAGE.

MOV R2,-(SP)
MOV #EF6401,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTB
ADD #6,SP

4$: INC R2 ;INCREMENT LINE COUNT.
CMP R3,R2 ;CHECK IF MAX LINE COUNT EXCEEDED.
BNE 2$ ;LOOP IF NOT DONE.

60$: PASS ;RESTORE THE SAVED CONTENTS OF THE GPRS.
PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

ENDMSG
L10005:

```

GLOBAL ERROR REPORTING ROUTINE - ER6401 -

012036 104423

TRAP C\$MSG

GLOBAL ERROR REPORTING ROUTINE

- ER7801 -

```

2068 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER7801 -
2069 ;*****
2070 ;* This is an error reporting subroutine which prints an additional error
2071 ;* message whose address is passed as an input parameter. A line number
2072 ;* is included at the end of the message.
2073 ;*
2074 ;* INPUTS: R1 - Address of the message to print.
2075 ;* R3 - Number of line on which error occurred.
2076 ;*
2077 ;* OUTPUTS: A messages is printed at the operator console.
2078 ;*
2079 ;* CALLING SEQUENCE: Load the address of the message in R1.
2080 ;* Load the line number into R3.
2081 ;* Include the label "ER7801" as the message pointer
2082 ;* parameter in the Diag Super error report macro call.
2083 ;*
2084 ;* COMMENTS: The message is printed as Basic error information.
2085 ;*
2086 ;* SUBORDINATE ROUTINES USED: None.
2087 ;*****
2088
2089 012040 BGNMSG ER7801
2090 012040 ER7801::
2091 012040 PRINTB #EF7801,R1,R3 ;PRINT THE MESSAGE.
2092 012040 010346 MOV R3,-(SP)
2093 012042 010146 MOV R1,-(SP)
012044 012746 004546 MOV #EF7801,-(SP)
012050 012746 000003 MOV #3,-(SP)
012054 010600 MOV SP,R0
012056 104414 TRAP C$PNTB
012060 062706 000010 ADD #10,SP
2092 012064 ENDMSG
2093 012064 L10006:
012064 104423 TRAP C$MSG

```

GLOBAL ERROR REPORTING ROUTINE

- ER8401 -

```

2095 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER8401 -
2096 ;*****
2097 ;* This error reporting subroutine is intended to report interactions
2098 ;* which have been found between a modem signal and other modem signals.
2099 ;* It analyzes the modem status which is stored in the STAT storage area
2100 ;* and reports any discrepancies which are found between this stored data
2101 ;* and the present state of the STAT registers. Specified bits on the
2102 ;* line associated with the specified line are ignored.
2103 ;*
2104 ;* INPUTS: R1 - Address of signal name message.
2105 ;* R2 - Bit map of bits to ignore on specified line.
2106 ;* R3 - Number of specified line.
2107 ;* CSRA - Contains the address of the DUT CSR.
2108 ;* NUMLNS - Equated to the number of lines on the DUT.
2109 ;* STATA - Contains the address of the DUT STAT register.
2110 ;* STSTB - Label at base of STAT storage table.
2111 ;* TXRLNB - Label at base of TX/RX line number association table.
2112 ;*
2113 ;* OUTPUTS: A messages is printed at the operator console.
2114 ;*
2115 ;* CALLING SEQUENCE: Include the label "ER8401" as the message pointer
2116 ;* parameter in the Diag Super error report macro call.
2117 ;*
2118 ;* COMMENTS: The message is printed as Basic and Extended error information.
2119 ;*
2120 ;* SUBORDINATE ROUTINES USED: None.
2121 ;*****
2122
2123 012066 BGNMSG ER8401
2124 012066 ER8401::
012066 004567 172022 SAVE JSR ;PRESERVE THE CONTENTS OF THE GPRS.
012066 ;CALL REGISTER SAVE SUBRT.
2125
2126 012072 PRINTB #EF8401,R1,R3 ;PRINT THE BASIC MESSAGE.
012072 010346 MOV R3,-(SP)
012074 010146 MOV R1,-(SP)
012076 012746 004604 MOV #EF8401,-(SP)
012102 012746 000003 MOV #3,-(SP)
012106 010600 MOV SP,R0
012110 104414 TRAP C$PNTB
012112 062706 000010 ADD #10,SP
2127
2128 012116 010167 000204 MOV R1,44$ ;SAVE THE ADDRESS OF THE SIGNAL NAME MESSAGE.
2129 012122 005001 CLR R1 ;CLEAR THE LINE COUNTER.
2130 012124 012704 002644 MOV #STSTB,R4 ;SET UP STAT STORAGE POINTER TO BASE OF TABLE.
2131 012130 010177 170106 2$: MOV R1,@CSRA ;SET UP THE CSR IND.ADR.REG FIELD.
2132 012134 017700 170110 MOV @STATA,R0 ;GET THE CONTENTS OF THIS LINE'S STAT REGISTER.
2133 012140 011405 MOV (R4),R5 ;GET THE PREVIOUS CONTENTS FROM STORAGE.
2134 012142 040005 BIC R0,R5
2135 012144 042400 BIC (R4)+,R0
2136 012146 050005 BIS R0,R5 ;XOR PRESENT AND STORED STAT VALUES.
2137 012150 012700 043777 MOV #43777,R0 ;PREPARE TO MASK OUT UNUSED BITS.
2138 012154 120163 004044 CMPB R1,TXRLNB(R3) ;IS THIS LINE ASSOCIATED WITH SPECIFIED LINE?
2139 012160 001002 BNE 4$ ;DON'T MASK OUT SPECIFIED BITS IF IT IS NOT.
2140 012162 056600 000006 BIS R2,SLOT(SP),R0 ;MASK OUT SPECIFIED BITS.
2141 012166 040005 4$: BIC R0,R5 ;GET BIT MAP OF UNDESIRED CHANGES.
2142 012170 032705 100000 BIT #BIT15,R5 ;CHECK FOR DSR SIGNAL INTERACTION.

```

GLOBAL ERROR REPORTING ROUTINE

- ER8401 -

```

2143 012174 001404          BEQ      6$          ;SKIP PRINTING LINE IF NO DSR INTERACTION.
2144 012176 012702 010304    MOV      #EM8403,R2   ;SELECT DSR ERROR MESSAGE.
2145 012202 004767 000064    JSR      PC,40$      ;PRINT THE LINE OF THE ERROR MESSAGE.
2146 012206 032705 020000    6$:     BIT      #BIT13,R5 ;CHECK FOR RI SIGNAL INTERACTION.
2147 012212 001404          BEQ      8$          ;SKIP PRINTING LINE IF NO RI INTERACTION.
2148 012214 012702 010310    MOV      #EM8404,R2   ;SELECT RI ERROR MESSAGE.
2149 012220 004767 000046    JSR      PC,40$      ;PRINT THE LINE OF THE ERROR MESSAGE.
2150 012224 032705 010000    8$:     BIT      #BIT12,R5 ;CHECK FOR DCD SIGNAL INTERACTION.
2151 012230 001404          BEQ     10$         ;SKIP PRINTING LINE IF NO DCD INTERACTION.
2152 012232 012702 010313    MOV      #EM8405,R2   ;SELECT DCD ERROR MESSAGE.
2153 012236 004767 000030    JSR      PC,40$      ;PRINT THE LINE OF THE ERROR MESSAGE.
2154 012242 032705 004000    10$:    BIT      #BIT11,R5 ;CHECK FOR CTS SIGNAL INTERACTION.
2155 012246 001404          BEQ     12$         ;SKIP PRINTING LINE IF NO CTS INTERACTION.
2156 012250 012702 010317    MOV      #EM8406,R2   ;SELECT CTS ERROR MESSAGE.
2157 012254 004767 000012    JSR      PC,40$      ;PRINT THE LINE OF THE ERROR MESSAGE.
2158
2159 012260 005201          12$:    INC      R1          ;SELECT NEXT LINE.
2160 012262 020127 000010    CMP      R1,#NUMLNS  ;ALL LINES DONE?
2161 012266 002720          BLT      2$          ;LOOP IF NOT ALL LINES DONE.
2162 012270 000417          BR       60$        ;EXIT THIS ROUTINE.
2163
2164          ;+
2164          ; Local error message line printing routine.
2165          ;-
2166 012272          40$:    PRINTX  #EF8402,44$,R3,R2,R1
2166 012272 010146          MOV      R1,-(SP)
2166 012274 010246          MOV      R2,-(SP)
2166 012276 010346          MOV      R3,-(SP)
2166 012300 016746 000022    MOV      44$,-(SP)
2166 012304 012746 004676    MOV      #EF8402,-(SP)
2166 012310 012746 000005    MOV      #5,-(SP)
2166 012314 010600          MOV      SP,R0
2166 012316 104415          TRAP    C$PNTX
2166 012320 062706 000014    ADD      #14,SP
2167 012324 000207          RTS      PC
2168 012326 000000          44$:    .WORD  0
2169 012330          60$:    PASS
2170 012332          ENDMSG          JSR
2170 012332          L10007: TRAP    C$MSG
2170 012332 104423

```

GLOBAL ERROR REPORTING ROUTINE

- ER9001 -

```

2172 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9001 -
2173 ;*****
2174 ;* This is an error reporting subroutine which reports an unexpected
2175 ;* code which has been found in the DUT CSR. This code can be a BMP
2176 ;* code, a Self-test code, or a Modem Status code.
2177 ;*
2178 ;* INPUTS: R1 - Address of message to print first.
2179 ;* R2 - Single byte code which has been read from the DUT.
2180 ;* R4 - Line number associated with the code.
2181 ;*
2182 ;* OUTPUTS: A messages is printed at the operator console.
2183 ;*
2184 ;* CALLING SEQUENCE: Include the label "ER9001" as the message pointer
2185 ;* parameter in the Diag Super error report macro call.
2186 ;*
2187 ;* COMMENTS: The message is printed as Basic and Extended error information.
2188 ;*
2189 ;* SUBORDINATE ROUTINES USED: None.
2190 ;*****
2191
2192 012334 BGNMSG ER9001
2193 012334 ER9001::
2194 012334 PRINTB #EF9001,R1 ;REPORT TYPE OF CODE FOUND.
2195 012334 010146 MOV R1,-(SP)
2196 012336 012746 005013 MOV #EF9001,-(SP)
2197 012342 012746 000002 MOV #2,-(SP)
2198 012346 010600 MOV SP,R0
2199 012350 104414 TRAP C$PNTB
2200 012352 062706 000006 ADD #6,SP
2201 012356 PRINTX #EF9002,R4 ;REPORT THE LINE NUMBER OF THE CODE.
2202 012356 010446 MOV R4,-(SP)
2203 012360 012746 005075 MOV #EF9002,-(SP)
2204 012364 012746 000002 MOV #2,-(SP)
2205 012370 010600 MOV SP,R0
2206 012372 104415 TRAP C$PNTX
2207 012374 062706 000006 ADD #6,SP
2208 012400 PRINTX #EF9003,R2 ;REPORT THE CODE WHICH WAS FOUND.
2209 012400 010246 MOV R2,-(SP)
2210 012402 012746 005147 MOV #EF9003,-(SP)
2211 012406 012746 000002 MOV #2,-(SP)
2212 012412 010600 MOV SP,R0
2213 012414 104415 TRAP C$PNTX
2214 012416 062706 000006 ADD #6,SP
2215
2216 012422 ENDMSG
2217
2218 012422 L10010:
2219 012422 104423 TRAP C$MSG

```

GLOBAL ERROR REPORTING ROUTINE

- ER9002 -

```

2200 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9002 -
2201 ;*****
2202 ;* This is an error reporting subroutine which is intended for use in the
2203 ;* transmission and reception tests. It reports the type of error which
2204 ;* has occurred when incorrect data is received from the DUT. This
2205 ;* routine also reports the read and expected data values.
2206 ;*
2207 ;* INPUTS: R1 - Address of message to print first.
2208 ;* R2 Data byte read from the DUT.
2209 ;* R3 - Line number multiplied by 2.
2210 ;* R4 Expected data byte, bit 15 set if "NONE".
2211 ;*
2212 ;* OUTPUTS: A messages is printed at the operator console.
2213 ;*
2214 ;* CALLING SEQUENCE: Include the label "ER9002" as the message pointer
2215 ;* parameter in the Diag Super error report macro call.
2216 ;*
2217 ;* COMMENTS: The message is printed as Basic and Extended error information.
2218 ;*
2219 ;* SUBORDINATE ROUTINES USED: PRTLPR.
2220 ;*****
2221
2222 012424 BGNMSG ER9002
2223 012424 ER9002::
2224 012424 006203 ASR R3 ;CALCULATE THE LINE NUMBER.
2225 012426 042702 177400 BIC #177400,R2 ;MASK OUT ALL BUT DATA IN READ CHAR.
2226 012432 PRINTB #EF9006,R1,R3 ;PRINT THE FIRST LINE OF THE MESSAGE.
2227 012432 010346 MOV R3,-(SP)
2228 012434 010146 MOV R1,(SP)
2229 012436 012746 005257 MOV #EF9006,(SP)
2230 012442 012746 000003 MOV #3,-(SP)
2231 012446 010600 MOV SP,R0
2232 012450 104414 TRAP C$PNTB
2233 012452 062706 000010 ADD #10,SP
2234 012456 PRINTX #EF9004,#EM9010,R2 ;PRINT ACTUAL DATA.
2235 012456 010246 MOV R2,-(SP)
2236 012460 012746 010672 MOV #EM9010,-(SP)
2237 012464 012746 005176 MOV #EF9004,-(SP)
2238 012470 012746 0000C3 MOV #3,-(SP)
2239 012474 010600 MOV SP,R0
2240 012476 104415 TRAP C$PNTX
2241 012500 062706 000010 ADD #10,SP
2242 012504 005704 TST R4 ;CHECK FOR "NONE" CODE SET IN EXPECTED DATA.
2243 012506 100414 BMI 2$ ;BRANCH TO PRINT "NONE" MESSAGE IF FLAG SET.
2244 012510 PRINTX #EF9004,#EM9009,R4 ;PRINT EXPECTED DATA.
2245 012510 010446 MOV R4,-(SP)
2246 012512 012746 010646 MOV #EM9009,-(SP)
2247 012516 012746 005176 MOV #EF9004,-(SP)
2248 012522 012746 000003 MOV #3,-(SP)
2249 012526 010600 MOV SP,R0
2250 012530 104415 TRAP C$PNTX
2251 012532 062706 000010 ADD #10,SP
2252 012536 000412 BR 60$ ;EXIT THIS ROUTINE.
2253 012540 PRINTX #EF9005,#EM9009 ;PRINT MESSAGE INDICATING NO EXPECTED DATA.
2254 012540 012746 010646 MOV #EM9009,-(SP)
2255 012544 012746 005226 MOV #EF9005,-(SP)

```



GLOBAL ERROR REPORTING ROUTINE

- ER9002 -

```

012550 012746 000002
012554 010600
012556 104415
012560 062706 000006
2233 012564 004767 002162
2234 012570
012570
012570 104423

```

```

60$: JSR PC,PRTLPR ;PRINT CONTENTS OF THE LPR REGISTER.
      ENDMSG

```

```

MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #6,SP

```

```

L10011: TRAP C$MSG

```

GLOBAL ERROR REPORTING ROUTINE

- ER9101 -

```

2236 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9101 -
2237 ;*****
2238 ;* This is a general error reporting subroutine which reports a message
2239 ;* which takes a single, 2 digit decimal argument after the end of an
2240 ;* ASCII message.
2241 ;*
2242 ;* INPUTS: R1 - Value to be printed after msg as 2 decimal digits.
2243 ;* R2 - Address of message to print first.
2244 ;*
2245 ;* OUTPUTS: A messages is printed at the operator console.
2246 ;*
2247 ;* CALLING SEQUENCE: Include the label "ER9101" as the message pointer
2248 ;* parameter in the Diag Super error report macro call.
2249 ;*
2250 ;* COMMENTS: The message is printed as Basic error information.
2251 ;*
2252 ;* SUBORDINATE ROUTINES USED: None.
2253 ;*****
2254
2255 012572 BGNMSG ER9101
2256 012572 ER9101::
2257 012572 PRINTB #EF9006,R2,R1 ;REPORT THE STRING FOLLOWED BY THE NUMBER.
2258 012572 010146 MOV R1,-(SP)
2259 012574 010246 MOV R2,-(SP)
012576 012746 005257 MOV #EF9006,-(SP)
012602 012746 000003 MOV #3,-(SP)
012606 010600 MOV SP,R0
012610 104414 TRAP C$PNTB
012612 062706 000010 ADD #10,SP
2258
2259 012616 ENDMSG
012616 L10012:
012616 104423 TRAP C$MSG

```

GLOBAL ERROR REPORTING ROUTINE

- ER9301 -

```

2261 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9301 -
2262 ;*****
2263 ;* This is an error reporting subroutine which prints any BMP codes
2264 ;* that are found in the BMP code queue, together with the the number of
2265 ;* the test that was executing at the time the BMP code was logged.
2266 ;*
2267 ;* INPUTS: R1 - The address of the first message to be reported.
2268 ;* R2 - The address of the next empty cell in the queue.
2269 ;*
2270 ;* OUTPUTS: The test number followed by the BMP code are printed at the
2271 ;* operator console.
2272 ;*
2273 ;* CALLING SEQUENCE: Include the label "ER9301" as the message pointer
2274 ;* parameter in the Diag Super error report macro call.
2275 ;*
2276 ;* COMMENTS: The message is printed as Basic error information.
2277 ;*
2278 ;* SUBORDINATE ROUTINES USED: None.
2279 ;*****
2280
2281 012620 BGNMSG ER9301
2282 012620 ER9301::
012620 004567 171270 SAVE ;SAVE THE GPRS ON THE STACK.
2283 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2284 012624 PRINTB #EF0503,R1 ;REPORT UNEXPECTED BMP CODES FOUND.
012624 010146 MOV R1,-(SP)
012626 012746 004247 MOV #EF0503,-(SP)
012632 012746 000002 MOV #2,-(SP)
012636 010600 MOV SP,R0
012640 104414 TRAP C$PNTB
012642 062706 000006 ADD #6,SP
2285 012646 012703 002444 MOV #BMPCQB,R3 ;GET THE START ADDRESS OF THE BMP CODE QUEUE.
2286 012652 012705 011147 MOV #EM9302,R5 ;GET THE MESSAGE TO BE REPORTED.
2287 012656 012301 2$: MOV (R3)+,R1 ;GET THE NUMBER OF THE TEST THAT WAS EXECUTING.
2288 012660 012304 MOV (R3)+,R4 ;GET BMP CODE THAT WAS REPORTED OFF THE QUEUE.
2289 012662 004767 000056 JSR PC,50$ ;GO REPORT THE BMP CODE.
2290 012666 020302 CMP R3,R2 ;CHECK IF ALL CODES HAVE BEEN REPORTED.
2291 012670 103772 BLO 2$ ;IF IT IS NOT THE LAST BMP CODE THEN LOOP.
2292 ;+
2293 ; Check if overflow has occurred.
2294 ; The conditions for overflow are: the pointer contains the address of the
2295 ; last cell in the queue, and a bmp code has already been written into that
2296 ; cell.
2297 ;-
2298 012672 020227 002640 CMP R2,#BMPCQE-4 ;CHECK IF THE POINTER IS AT THE LAST LOCATION.
2299 012676 001036 BNE 60$ ;EXIT IF NOT AT THE LAST LOCATION.
2300 012700 005762 000002 TST 2(R2) ;CHECK FOR A BMP CODE IN THE LAST CELL
2301 012704 001433 BEQ 60$ ;EXIT IF NO OVERFLOW HAS OCCURED, CELL EMPTY.
2302 012706 012301 MOV (R3)+,R1 ;GET THE TEST NUMBER OFF THE QUEUE.
2303 012710 011304 MOV (R3),R4 ;GET THE BMP CODE OFF THE QUEUE.
2304 012712 012705 011177 MOV #EM9303,R5 ;SELECT THE MESSAGE TO BE REPORTED.
2305 012716 PRINTX #EF9302 ;REPORT OVERFLOW CONDITION.
012716 012746 005373 MOV #EF9302,-(SP)
012722 012746 000001 MOV #1,-(SP)
012726 010600 MOV SP,R0
012730 104415 TRAP C$PNTX

```

GLOBAL ERROR REPORTING ROUTINE

- ER9301 -

```

012732 062706 000004
2306 012736 004767 000002      JSR    PC,50$      ;REPORT THE LAST BMP CODE PLACED ON THE QUEUE.
2307 012742 000414              BR     60$        ;EXIT.
2308
2309 012744              50$: PRINTX #EF9301,R5,R1,R4 ;PRINT THE MESSAGE.
012744 010446              MOV    R4,-(SP)
012746 010146              MOV    R1,-(SP)
012750 010546              MOV    R5,-(SP)
012752 012746 005315              MOV    #EF9301,-(SP)
012756 012746 000004              MOV    #4,-(SP)
012762 010600              MOV    SP,R0
012764 104415              TRAP  C$PNTX
012766 062706 000012              ADD   #12,SP
2310 012772 000207
2311 012774              60$: RTS    PC      ;RETURN.
012774 004736              PASS   ;RESTORE THE GPR CONTENTS.
2312              JSR    PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2313 012776              ENDMSG
012776
012776 104423              L10013: TRAP  C$MSG

```

GLOBAL SUBROUTINES SECTION

2315  
2317  
2318  
2319  
2320  
2321  
2323  
2324  
2325  
2326  
2327  
2328

```
.SBTTL GLOBAL SUBROUTINES SECTION  
;*****  
;  
; SKL3.P11  
;*****  
;  
;+  
; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES  
; THAT ARE USED IN MORE THAN ONE TEST.  
;--
```

GLOBAL SUBROUTINE

- ALTFLD -

2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364  
2365  
2366  
2367  
2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377  
2378  
2379  
2380  
2381  
2382  
2383  
2384  
2385

013000  
013000 004567 171110  
  
013004 010400  
013006 005100  
013010 040002  
013012 016705 167252  
  
013016 000241  
013020 006003  
013022 103006  
013024 010577 167212  
013030 011100  
013032 040400  
013034 050200  
013036 010011  
013040 005205  
013042 005703  
013044 001365

```

.SBTTL GLOBAL SUBROUTINE - ALTFLD -
;+ *****
;+ - Alter Device Register Fields Routine -
;+ This subroutine alters the specified field of the specified device
;+ register for the specified lines. This routine can be used to set
;+ or clear bits within selected fields of selected registers.
;+ Use examples: Set RX.BAUD.RATE fields on lines 3 and 6.
;+ Clear TX.DMA bits on all lines.
;+
;+ INPUTS: R1 - Address of the registers to alter.
;+ R2 - Bit fields set to desired states.
;+ R3 - Bit map of lines for which to alter register.
;+ R4 - Mask of bits to alter (1 indicates change bit).
;+ CSRA - Contains the address of the device CSR.
;+ IESTAT - Saved states of the interrupt enable bits.
;+
;+ OUTPUTS: DEVICE REGISTERS - Specified register fields altered.
;+ CSR IND.ADR.REG field Destroyed.
;+
;+ CALLING SEQUENCE: JSR PC,ALTFLD
;+
;+ COMMENTS: This routine reads the specified registers for all lines
;+ with numbers lower than the highest specified line.
;+ This routine does not read the CSR.
;+
;+ SUBROUTINES CALLED: None.
; - *****
ALTFLD.: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSP R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;+
; Set up to loop for each line:
; Prepare the word to be ORed into the register contents.
; Set up the word to write into the IND.ADR.REG field of the CSR.
;
MOV R4,R0 ;CALCULATE THE NEW CONTENTS OF THE
COM R0 ; REGISTER FIELDS WHICH ARE TO BE
BIC R0,R2 ; ALTERED BY THIS ROUTINE.
MOV IESTAT,R5 ;SET UP TO WRITE IND.ADR.REG FIELD TO 0.
;+
; Loop once for each line, altering the specified field in the specified
; register if the line has been selected for altering.
; Exit the loop if no more lines to alter, or if we have altered the max
; allowable number of lines (as specified by NUMLNS).
;
CLC ;PREPARE FOR ROTATE, "TST R5" DOES THIS BELOW.
2$: ROR R3 ;GET THE LINE SELECT BIT FOR THIS LINE.
BCC 4$ ;SKIP SETUP IF LINE IS NOT SELECTED.
MOV R5,@CSRA ;SET OUT CSR IND.ADR.REG FIELD TO THIS LINE.
MOV (R1),R0 ;GET THE PRESENT CONTENTS OF THE REG TO ALTER.
BIC R4,R0 ;CLEAR THE BIT FIELDS WE ARE TO ALTER.
BIS R2,R0 ;OR IN THE NEW STATES OF THE FIELDS.
MOV R0,(R1) ;WRITE THE NEW REGISTER CONTENTS TO THE REG.
4$: INC R5 ;SET LINE NUMBER TO THE NEXT LINE.
TST R3 ;CHECK FOR UNHANDLED LINES, CLEAR CARRY FLAG.
BNE 2$ ;LOOP IF SELECTED LINE(S) IS NOT HANDLED.

```

GLOBAL SUBROUTINE

- ALTFLD -

```
2386  
2387 013046          60$: PASS          JSR          ;RESTORE GPRS.  
      013046 004736          ;PC,@(SP)+ ;RETURN TO PREG05 SUBRT.  
2388 013050 000207          RTS          PC          ;RETURN TO CALLING ROUTNE.
```

GLOBAL SUBROUTINE

- ASLN TL -

2390  
2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402  
2403  
2404  
2405  
2406  
2407  
2408  
2409  
2410  
2411  
2412  
2413  
2414  
2415  
2416  
2417  
2418  
2419  
2420  
2421  
2422  
2423  
2424  
2425  
2426  
2427  
2428  
2429  
2430  
2431  
2432  
2433  
2434  
2435  
2436  
2437  
2438  
2439  
2440  
2441  
2442  
2443  
2444  
2445

013052  
013052 004567 171036  
013056 126727 167154 000002  
013064 001411  
  
013066 005005  
013070 010565 004004  
013074 005205  
013076 005205  
013100 020527 000020  
013104 002771  
013106 000411  
  
013110 012701 004064  
013114 012702 004004  
013120 112122  
013122 105022  
013124 020227 004044  
013130 002773  
  
013132 012701 004004  
013136 012702 004044  
013142 012103  
013144 006203  
013146 110322  
013150 020127 004044

```
.SBTTL GLOBAL SUBROUTINE - ASLN TL -
;+ *****
;* - Setup Associated Line Number Tables Routine -
;* This routine sets up the two tables which are contain information
;* about the TX/RX line which is associated with a particular RX/TX
;* line. One table is a table of words which contains word offset
;* values and the other table is a table of bytes which contains
;* line number values.
;*
;* INPUTS: LOPBCK - Storage for the type of loopback on the DUT.
;* NUMLNS - Equated to the number of lines on the DUT.
;* STGTRB - Label at base of staggered line association tbl.
;* TXRLNB - Label at base of byte TX/RX line number table.
;* TXRXLB - Label at base of word TX/RX line number table.
;* TXRXLE - Label at end of word TX/RX line number table.
;*
;* OUTPUTS: TXRXL, TXRLN - Tables initialized for selected loopback.
;*
;* CALLING SEQUENCE: JSR PC,ASLN TL
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: None.
;- *****
ASLN TL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
CMPB LOPBCK,#2 ;TEST FOR STAGGERED LOOPBACK.
BEQ 4$ ;GO SET UP STAGGERED TABLE IF STAGGERED LPBCK.
;+
; Set up the word table for non-staggered loopback.
;-
2$: CLR R5 ;CLEAR THE LINE COUNTER
MOV R5, TXRXLB(R5) ;SET UP A WORD OF THE TABLE.
INC R5
INC R5 ;SET LINE COUNTER TO NEXT LINE OFFSET.
CMP R5,#2*NUMLNS ;TEST FOR ALL LINES DONE.
BLT 2$ ;LOOP UNTIL ALL LINES DONE.
BR 8$ ;GO SET UP THE BYTE TABLE.
;+
; Set up the word table for staggered loopback.
;-
4$: MOV #STGTRB,R1 ;SET UP THE SOURCE POINTER.
MOV #TXRXLB,R2 ;SET UP THE DESTINATION POINTER.
6$: MOVB (R1)+,(R2)+ ;MOVE A BYTE INTO THE TABLE.
CLRB (R2)+ ;CLEAR THE UPPER BYTE OF WORD TABLE ENTRY.
CMP R2,#TXRXLE ;COMPARE POINTER WITH END ADR OF TABLE.
BLT 6$ ;LOOP IF NOT AT END YET.
;+
; Set up the byte table based on the word association table.
;-
8$: MOV #TXRXLB,R1 ;SET UP THE SOURCE POINTER.
MOV #TXRLNB,R2 ;SET UP THE DESTINATION POINTER.
10$: MOV (R1)+,R3 ;GET THE WORD OFFSET VALUE FROM WORD TABLE.
ASR R3 ;DIVIDE BY 2 TO GET LINE NUMBER VALUE.
MOVB R3,(R2)+ ;LOAD THE BYTE LINE NUMBER INTO TABLE.
CMP R1,#TXRXLE ;COMPARE SOURCE POINTER WITH ADR OF TABLE END.
```



GLOBAL SUBROUTINE

- ASLNTL -

```
2446 013154 002772      BLT      10$      ;LOOP IF NOT AT END OF TABLE YET.  
2447  
2448 013156      60$:  PASS      JSR      ;RESTORE GPRS.  
      013156 004736      PC,(SP)+ ;RETURN TO PREG05 SUBRT.  
2449 013160 000207      RTS      PC
```

GLOBAL SUBROUTINE

- CALMSL -

```

2451 .SBTTL GLOBAL SUBROUTINE - CALMSL -
2452 ;* *****
2453 ;* - Calibrate Milli Second Loop count subroutine -
2454 ;* This subroutine calibrates the timing loop which is used in the MSLOOP
2455 ;* routine. This subroutine calculates a value for the MSLCNT variable
2456 ;* which is the number of software loops which takes 1 ms to execute in
2457 ;* the MSLOOP routine. This routine calibrates the count by using the
2458 ;* Line Time Clock (LTC), so if no LTC is available the default value for
2459 ;* the delay count must be used.
2460 ;*
2461 ;*
2462 ;* INPUTS: MSLCNT - Default 1 ms delay loop count value, or
2463 ;* value from previous calibration.
2464 ;* MSTICK - Number of MS per LTC clock tick.
2465 ;* TIMER1 - Timer counter changed by LTC interrupt service rtn.
2466 ;* CLKHRZ - Number of LTC clicks per second (50 or 60).
2467 ;*
2468 ;* OUTPUTS: CARRY - Set if LTC is available, and new calibration performed.
2469 ;* MSLCNT - New 1 ms delay loop count value if LTC available, or
2470 ;* unchanged if no LTC is available.
2471 ;*
2472 ;* CALLING SEQUENCE: JSR PC,CALMSL
2473 ;*
2474 ;* COMMENTS:
2475 ;*
2476 ;* SUBORDINATE ROUTINES CALLED: UNSDIV,OOPS.
2477 ;*-- *****
2478
2479 013162 CALMSL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
013162 004567 170726 JSP R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2480 013166 .05067 000210 CLR 62$ ;CLEAR THE 2ND TIME FLAG.
2481 ;*
2482 ; Synchronize with the LTC.
2483 ;-
2484 013172 012705 000001 2$: MOV #1,R5 ;SET OUTER LOOP COUNTER TO 1 LOOP.
2485 ;INCREASE THE VALUE LOADED INTO THIS COUNTER IF THE <*&
2486 ;FOLLOWING LOOP FAILS ON FUTURE, FASTER PROCESSORS. <*&
2487 013176 005000 CLR R0 ;CLEAR THE WAIT FOR CLOCK INT COUNTER.
2488 013200 012767 000001 167120 MOV #1,TIMER1 ;SET UP COUNT OF 1 TO SYNCH WITH LTC.
2489 013206 005767 167114 4$: TST TIMER1 ;CHECK FOR COUNTER HAVING GONE TO ZERO.
2490 013212 001410 BEQ 6$ ;JUMP OUT OF LOOP IF LTC HAS INTERRUPTED.
2491 013214 005200 INC R0 ;COUNT THIS ITERATION OF THE INNER LOOP.
2492 013216 001373 BNE 4$ ;LOOP IF COUNTER HAS NOT TURNED OVER.
2493 013220 005305 DEC R5 ;DECREMENT THE INNER LOOP COUNTER.
2494 013222 003371 BGT 4$ ;LOOP IF OUTER LOOP COUNT NOT UP.
2495 ;*
2496 ; If we got no LTC interrupt, indicate that there is no LTC available.
2497 ; LTC must be flakey, or not really an LTC at all.
2498 ;-
2499 013224 005067 167074 CLR CLKHRZ ;CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.
2500 013230 000241 CLC ;INDICATE FAILURE FOR RETURN.
2501 013232 000461 BR 60$ ;BYPASS THE FOLLOWING CALIBRATION PROCEDURES.
2502 ;*
2503 ; We are now synchronized with the LTC.
2504 ; Set up for the calibration loop.
2505 ;
2506 013234 012704 002326 6$: MOV #TIMER1,R4 ;WILL TEST TIMER1 IN THE LOOP BELOW.

```

GLOBAL SUBROUTINE

- CALMSL -

```

2507 013240 005001          CLR    R1          ;CLEAR THE OUTER LOOP COUNTER.
2508 013242 005002          CLR    R2          ;INDICATE TO CHECK ALL BITS OF TIMER1.
2509 013244 005003          CLR    R3          ;INDICATE TO CHECK FOR TIMER1 CLEAR.
2510 013246 012714 000001   MOV    #1,(R4)     ;LOAD TIMER1 WITH COUNT OF 1.
2511
2512 013252 016705 167062   8$:   MOV    MSLCNT,R5 ;LOAD MS LOOP COUNT.
2513 013256 011400 10$:   MOV    (R4),R0     ;GET THE TIMER1 VALUE.
2514 013260 010067 000120   MOV    R0,64$     ;SAVE WORD (LIKE IN THE REAL LOOP).
2515 013264 040200          BIC    R2,R0      ;LEAVE ALL THE BITS.
2516 013266 020003          CMP    R0,R3      ;COMPARE AGAINST ZERO.
2517 013270 000261          SEC          ;SET CARRY IN CASE OF SUCCESS.
2518 013272 001406          BEQ    12$       ;EXIT LOOP IF TIMER1 HAS CLEARED.
2519 013274 005305          DEC    R5        ;COUNT DOWN THE INSIDE MS LOOP COUNT.
2520 013276 001367          BNE    10$       ;LOOP IF MS NOT UP.
2521 013300 005301          DEC    R1        ;DECREMENT THE MS TIME COUNT.
2522 013302 001363          BNE    8$        ;KEEP LOOPING.
2523 013304 004767 001214   JSR    PC,OOPS    ;WE OVERFLOWED, SOMETHING IS WRONG, ABORT.
2524
2525          ;*
2526          ; We have now have loop count information for one clock tick.
2527          ; We have negative of number of outer loops in R1, each is MSLCNT inner loops.
2528          ; We have the portion of the last outer loop not executed, in R5.
2529          ; Now we calculate the total number of inner loops executed.
2530 013310 005401 12$:   NEG    R1          ;GET NUMBER OF OUTER LOOPS.
2531 013312 016702 167022   MOV    MSLCNT,R2  ;GET THE NUMBER OF INNER LOOPS PER OUTER LOOP.
2532 013316 010203          MOV    R2,R3      ;COPY NUMBER OF LOOPS FOR MULTIPLY.
2533 013320 160502          SUB    R5,R2      ;CALC # OF INNER LOOPS DONE IN LAST OUTER LOOP
2534 013322 010204          MOV    R2,R4      ; AND ADD TO ACCUMULATOR LSWORD.
2535 013324 005005          CLR    R5        ;CLEAR ACCUMULATOR MSWORD.
2536 013326 005301 14$:   DEC    R1          ;CHECK R1 FOR 0 CONDITION
2537 013330 100403          BMI    16$       ; SKIP MULTIPLICATION IF ZERO
2538 013332 060304          ADD    R3,R4     ;MULTIPLY NUMBER OF INNER
2539 013334 005505          ADC    R5        ; LOOPS PER OUTER LOOP BY
2540 013336 000773          BR    14$       ;NUMBER OF OUTER LOOPS PERFORMED.
2541
2542          ;*
2543          ; Divide the total number of inner loops by the number of MS per LTC tick.
2544 013340 016701 166772   16$:   MOV    MSTICK,R1  ; # OF MS PER LTC TICK IS DIVISOR.
2545 013344 010403          MOV    R4,R3     ;LSWORD OF LOOP COUNT IS LSWORD OF DIVIDEND.
2546 013346 010502          MOV    R5,R2     ;MSWORD OF LOOP COUNT IS MSWORD OF DIVIDEND.
2547 013350 004767 003012   JSR    PC,UNSDIV  ;DIVIDE NUMBER OF LOOPS BY MS PER LTC TICK.
2548 013354 103402          BCS    18$       ;BYPASS OOPS IF WE'RE OK.
2549 013356 004767 001142   JSR    PC,OOPS    ;CLOCK ROUTINES ARE NOT LONG ENOUGH, OR BUG.
2550 013362 010167 166752   18$:   MOV    R1,MSLCNT ;SET NEW VALUE FOR MS LOOP COUNT.
2551 013366 005167 000010   COM    62$       ;SET THE 2ND ITERATION FLAGS IF 1ST ITERATION.
2552 013372 001277          BNE    2$        ;BRANCH IF ONLY ONE ITERATION DONE.
2553 013374 000261          SEC          ;SET THE SUCCESS FLAG FOR EXIT.
2554
2555 013376          60$:   PASS          ;RESTORE GPRS.
2556 013400 004736          RTS    PC        ;RETURN TO PREGOS SUBRT.
2557          ; CARRY - SUCCESS FLAG. SET IF SUCCESS.
2558 013402 000000 62$:   .WORD 0          ;2ND CALIBRATION ITERATION FLAGS.
2559 013404 000000 64$:   .WORD 0          ;DUMMY WORD FOR STORAGE OF THE READ WORD.

```

GLOBAL SUBROUTINE

- CHKBMP -

```

2561 .SBTTL GLOBAL SUBROUTINE - CHKBMP -
2562 ;** *****
2563 ;* CHECK IF CHARACTER IS A BMP CODE -
2564 ;* This subroutine is used to check for BMP codes.
2565 ;* If a BMP code is detected, it will be saved on the queue to be reported
2566 ;* later. The carry is used as a flag to indicate a code has been found.
2567 ;*
2568 ;* INPUTS: R2 - Contains the data to be checked.
2569 ;*
2570 ;* OUTPUTS: R1: Contains the message to be reported.
2571 ;* RRBLK - Contains the error reporting routine.
2572 ;* Carry bit is used to indicate a BMP code found, Carry set.
2573 ;*
2574 ;* CALLING SEQUENCE: JSR PC,CHKBMP
2575 ;*
2576 ;* COMMENTS:
2577 ;*
2578 ;* SUBORDINATE ROUTINES CALLED: SAVBMP.
2579 ;-- *****
2580
2581 CHKBMP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
2582 013406 004567 170502 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2583 013412 012700 170301 MOV #170301,R0 ;SET UP THE FLAGS OF A BMP CODE.
2584 013416 040200 BIC R2,R0 ;TRY TO CLEAR THE BMP CODE FLAGS.
2585 013420 001011 BNE 2$ ;IF NOT A BMP CODE, EXIT WITH FAILURE.
2586 013422 004767 002116 JSR PC,SAVBMP ;SAVE THE BMP CODE ON THE QUEUE.
2587 013426 012701 006634 MOV #EM5303,R1 ;PASS THE MESSAGE TO BE REPORTED.
2588 013432 012767 011662 170452 MOV #ER1603,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
2589 013440 000261 SEC ;PASS FLAG TO INDICATE SUCCESS, BMP CODE FOUND.
2590 013442 000401 BR 60$ ;EXIT.
2591 013444 000241 2$: CLC ;PASS FLAG TO INDICATE FAILURE.
2592 013446 010166 000004 60$: PASS R1 ;RESTORE GPRS, EXCEPT
2593 013452 004736 MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
2594 013454 000207 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
;R1 - CONTAINS THE ADDRESS OF ERROR MESSAGE.
;CARRY BIT - SET INDICATES SUCCESS.
RTS PC

```

GLOBAL SUBROUTINE

- CKTRAP -

```

2596 .SBTTL GLOBAL SUBROUTINE - CKTRAP -
2597 ;*****
2598 ;* Check Trap Routine -
2599 ;* This subroutine is used to check for a bus time out trap (004 trap)
2600 ;* which is caused by an access to a non-existent memory or I/O location.
2601 ;* If the trap does not occur, this routine returns a success indication.
2602 ;*
2603 ;* INPUTS: R0 - Source address for move.
2604 ;* R1 - Destination address for move.
2605 ;* (R0) - Source for the move.
2606 ;*
2607 ;* OUTPUTS: (R1) - Written to the contents of (R0).
2608 ;* Carry flag - Set on return if no 004 trap detected.
2609 ;* TP4FLG - Nonzero if trap occurred, cleared otherwise.
2610 ;*
2611 ;* CALLING SEQUENCE: JSR PC,CKTRAP
2612 ;*
2613 ;* COMMENTS: If this subroutine causes a trap, either the address which
2614 ;* is labeled ADRPTR will be the trap PC address on the stack.
2615 ;*
2616 ;* SUBORDINATE ROUTINES CALLED: None.
2617 ;*****
2618
2619 013456 CKTRAP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
013456 004567 170432 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2620 013462 005067 166624 CLR TP4FLG ;CLEAR THE 004 TRAP FLAGS.
2621 013466 011011 MOV (R0),(R1) ;PERFORM THE MOVE IN QUESTION.
2622 013470 005767 166616 ADRPTR:: TST TP4FLG ;CHECK FOR OCCURENCE OF TRAP.
2623 013474 000261 SEC ;INDICATE SUCCESS.
2624 013476 001401 BEQ 60$ ;EXIT WITH SUCCESS IF TRAP DID NOT OCCUR.
2625 013500 000241 CLC ;INDICATE FAILURE.
2626 013502 004736 60$: PASS ;RESTORE GPRS.
013502 000207 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2627 013504 000207 RTS PC

```

GLOBAL SUBROUTINE

CLNRST -

```

2629 .SBTTL GLOBAL SUBROUTINE - CLNRST -
2630 ;*****
2631 ;* - Clean Reset of the Device Under Test -
2632 ;* This subroutine is used to reset the DUT to a known state.
2633 ;* The DUT's self-test is skipped, and the fifo is purged of any error
2634 ;* codes, etc.
2635 ;* If the reset does not successfully complete, then the carry bit is
2636 ;* passed back to the calling routine (clear).
2637 ;*
2638 ;* INPUTS: CSRA - Contains the address of the CSR
2639 ;* TXBFCA - Contains address of DUT DMA Buffer Count register.
2640 ;* ERRNBR - Error number for possible error report.
2641 ;* ERRTBL- ERRTYP,ERNBR,and ERRMSG set up correctly.
2642 ;*
2643 ;* OUTPUTS: The DUT performs its reset function into a known state.
2644 ;* CARRY - Clear indicates the test is to be aborted.
2645 ;* ERRBLK - value may be destroyed.
2646 ;* IESTAT - TX and RX interrupt flags are cleared.
2647 ;* TX and RX interrupt enable bits in the DUT's CSR are cleared.
2648 ;*
2649 ;* CALLING SEQUENCE: JSR PC,CLNRST
2650 ;*
2651 ;* COMMENTS: This subroutine can report errors with numbers ERRNBR.
2652 ;* This routine does not destroy the value of ERRNBR.
2653 ;*
2654 ;* SUBORDINATE ROUTINES CALLED: DELAY,MSLGET,PUFIFO,RESETT.
2655 ;*****
2656
2657 013506 013506 004567 170402 CLNRST:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSP R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2658 ;+
2659 ; Reset the DUT.
2660 ; This routine reports errors with numbers from ERRNBR thru ERRNBR+2.
2661 ;-
2662 013512 004767 001654 JSR PC,RESETT ;RESET THE DUT TO A KNOWN STATE.
2663 013516 103002 BCC 60$ ;EXIT ROUTINE WITH ABORT TEST INDICATOR.
2664 ;+
2665 ; Purge the FIFO of error codes, save any BMP codes found.
2666 ;-
2667 013520 004767 001310 JSR PC,PUFIFO ;PURGE THE FIFO.
2668
2669 013524 60$: ;EXIT THE TEST USING RESETT OR PUFIFO STATUS.
2670 013524 PASS ;RESTORE GPRS, PASS THE FOLLOWING INTACT:
013524 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
;CARRY BIT:IF CLEAR, THEN ABORT THE TEST.
2671
2672 013526 000207 RTS PC

```

GLOBAL SUBROUTINE

- CLR16W -

2674  
 2675  
 2676  
 2677  
 2678  
 2679  
 2680  
 2681  
 2682  
 2683  
 2684  
 2685  
 2686  
 2687  
 2688  
 2689  
 2690 013530  
 013530 004567 170360  
 2691 013534 012701 000020  
 2692 013540 005020  
 2693 013542 005301  
 2694 013544 001375  
 2695 013546  
 013546 004736  
 2696 013550 000207

```
.SBTTL GLOBAL SUBROUTINE - CLR16W -
; * *****
; * - Clear Sixteen Words Routine -
; * This subroutine clears 16 words starting with the specified word.
; *
; * INPUTS: R0 - Address of the first word to clear.
; *
; * OUTPUTS: (R0) to (R0+15) - 16 words of memory are cleared to 0.
; *
; * CALLING SEQUENCE: JSR PC,CLR16W
; *
; * COMMENTS:
; *
; * SUBORDINATE ROUTINES CALLED: None.
; *
; * *****
CLR16W:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;CALL REGISTER SAVE SUBRT.
MOV #16.,R1 ;SET THE LOOP COUNTER TO 16.
2$: CLR (R0)+ ;CLEAR A WORD OF MEMORY.
DEC R1 ;COUNT THIS LOOP.
BNE 2$ ;LOOP IF NOT 16 WORD CLEARED.
60$: PASS ;RESTORE GPRS.
;RETURN TO PREG05 SUBRT.
JSR PC,@(SP)+
RTS PC
```

GLOBAL SUBROUTINE

- CMPMST -

```

2698 .SBTTL GLOBAL SUBROUTINE - CMPMST -
2699 ;* *****
2700 ;* - Compare Modem Status Routine -
2701 ;* This routine is used to compare the present modem status against the
2702 ;* modem status which is stored in the modem status storage table. It
2703 ;* ignores the states of the specified signals on a specified line.
2704 ;*
2705 ;* INPUTS: R1 - Line number of specified line.
2706 ;* R2 - Bit map of bits to ignore on specified line.
2707 ;* CSRA - Contains the address of the DUT CSR.
2708 ;* NUMLNS Equated to the number of lines on the DUT.
2709 ;* STATA - Contains the address of the DUT STAT register.
2710 ;* STSTB - Label at base of STAT storage table.
2711 ;* TXRLNB - Label at base of TX/RX line number association table.
2712 ;*
2713 ;* OUTPUTS: CARRY - Success flag (Set if no discrepancies were found).
2714 ;*
2715 ;* CALLING SEQUENCE: JSR PC,CMPMST
2716 ;*
2717 ;* COMMENTS:
2718 ;*
2719 ;* SUBORDINATE ROUTINES CALLED: None.
2720 ;-- *****
2721
2722 013552 CMPMST:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
013552 004567 170336 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2723 013556 005003 CLR R3 ;CLEAR THE LINE COUNTER.
2724 013560 012704 002644 MOV #STSTB,R4 ;SET UP STAT STORAGE POINTER TO BASE OF TABLE.
2725 013564 010377 166452 2$: MOV R3,@CSRA ;SET UP THE CSR IND.ADR.REG FIELD.
2726 013570 017700 166454 MCV @STATA,R0 ;GET THE CONTENTS OF THIS LINE'S STAT REGISTER.
2727 013574 011405 MOV (R4),R5 ;GET THE PREVIOUS CONTENTS FROM STORAGE.
2728 013576 040005 BIC R0,R5
2729 013600 042400 BIC (R4)+,R0
2730 013602 050005 BIS R0,R5 ;XOR PRESENT AND STORED STAT VALUES.
2731 013604 012700 043777 MOV #43777,R0 ;PREPARE TO MASK OUT UNUSED BITS.
2732 013610 120301 CMPB R3,R1 ;TEST FOR THIS BEING SPECIFIED LINE.
2733 013612 001001 BNE 10$ ;DON'T MASK OUT SPECIFIED BITS IF IT IS NOT.
2734 013614 050200 BIS R2,R0 ;MASK OUT SPECIFIED BITS.
2735 013616 040005 10$: BIC R0,R5 ;GET BIT MAP OF UNDESIREED CHANGES.
2736 013620 001006 BNE 50$ ;EXIT WITH FAILUP IF CHANGES OCCURRED.
2737 013622 005203 INC R3 ;SELECT NEXT LINE.
2738 013624 020327 000010 CMP R3,#NUMLNS ;ALL LINES DONE?
2739 013630 002755 BLT 2$ ;LOOP IF NOT ALL LINES DONE.
2740 013632 000261 SEC ;INDICATE SUCCESS.
2741 013634 000401 BR 60$ ;EXIT THIS ROUTINE WITH SUCCESS.
2742
2743 013636 000241 50$: CLC ;INDICATE FAILURE.
2744
2745 013640 60$: PASS ;RESTORE GPRS.
013640 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2746 013642 000207 - RTS PC ; CARRY SUCCESS FLAG (SET IF SUCCESS).

```



GLOBAL SUBROUTINE

- CONMAP -

```

2748 .SBTTL GLOBAL SUBROUTINE - CONMAP -
2749 ;** *****
2750 ;* - Convert Line bit map.
2751 ;* This subrouitne is used to convert a bit map passed to it , into
2752 ;* another line bit map that is based upon the associated TX/RX line
2753 ;* number/offset table.
2754 ;*
2755 ;* INPUTS: R5 - Contains the line bit map to be transformed.
2756 ;* TXRXLB - Base address of associated TX/RX line number table.
2757 ;*
2758 ;* OUTPUTS: R5 - Contains an associated line bit map.
2759 ;*
2760 ;* CALLING SEQUENCE: JSR PC,CONMAP
2761 ;*
2762 ;* COMMENTS: The TX/RX association table must be initialised before this
2763 ;* routine is called.
2764 ;*
2765 ;* SUBORDINATE ROUTINES CALLED: NONE.
2766 ;-- *****
2767
2768 013644 CONMAP::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
013644 004567 170244 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2769 013650 012702 004004 MOV #TXRXLB,R2 ;GET THE BASE ADDRESS OF THE LINE ASSOC TABLE.
2770 013654 010503 MOV R5,R3 ;COPY THE BIT MAP TO BE TRANSFORMED.
2771 013656 012704 000010 MOV #NUMLNS,R4 ;SET MAX LINE COUNTER.
2772 013662 005005 CLR R5 ;CLEAR ASSOCIATED LINE BIT MAP.
2773 013664 006203 2$: ASR R3 ;SHIFT ACTLNS BIT MAP INT BOOLEAN REGISTER.
2774 013666 103005 BCC 4$ ;SKIP SETTING ASSOCIATED LINE NUMBER BIT MAP.
2775 013670 011201 MOV (R2),R1 ;GET ASSOCIATED LINE NUMBER OFFSET FROM TABLE.
2776 013672 006201 ASR R1 ;SHIFT RIGHT TO GET LINE NUMB FROM OFFSET.
2777 013674 004767 000414 JSR PC,LINBIT ;GENERATE AN SINGLE BIT MAP FOR THIS LINE.
2778 013700 050005 BIS R0,R5 ;SET BIT FOR THIS LINE IN ASSOCIATED BIT MAP.
2779 013702 005722 4$: TST (R2)+ ;INCREMENT ADDRESS FOR THE NEXT LINE NUMBER.
2780 013704 005304 DEC R4 ;DECREMENT LINE COUNT.
2781 013706 001366 BNE 2$ ;LOOP IF NOT DONE.
2782 013710 60$: PASS R5 ;RESTORE GPRS, EXCEPT
013710 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
013714 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2783 ;R5 CONTAINS THE ASSOCIATED LINE BIT MAP.
2784 013716 000207 RTS PC

```

GLOBAL SUBROUTINE

- DELAY -

```

2786 .SBTTL GLOBAL SUBROUTINE - DELAY -
2787 ;*****
2788 ;* - DELAY SUBROUTINE -
2789 ;* This subroutine is used to delay a variable number of milli-seconds.
2790 ;*
2791 ;* INPUTS: R4 - Contains the number of ms to delay.
2792 ;* MSLCNT.
2793 ;*
2794 ;* OUTPUTS: None.
2795 ;*
2796 ;* CALLING SEQUENCE: JSR PC,DELAY
2797 ;*
2798 ;* COMMENTS: If no hardware clock interrupts are occurring, control-Cs will
2799 ;* not be honored for the duration of the delay.
2800 ;*
2801 ;* SUBORDINATE ROUTINES CALLED: None.
2802 ;*****
2803
2804 013720 DELAY:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
013720 004567 170170 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2805 013724 010401 MOV R4,R1 ;PASS NUMBER OF MS DELAY AS TIME-OUT VALUE.
2806 013726 012702 MOV #-1,R2 ;TELL MSLOOP ROUTINE TO CHECK ALL BITS.
2807 013732 005003 CLR R3 ;TELL MSLOOP RTN TO CHECK FOR ALL BITS CLEAR.
2808 013734 012704 013756 MOV #62$,R4 ;TELL MSLOOP TO CHECK DUMMY NON-ZERO WORD.
2809 013740 004767 000544 JSR PC,MSLOOP ;DELAY THE REQUESTED # OF MS.
2810 013744 103002 BCC 60$ ;EXIT ROUTINE IF WE TIMED-OUT.]
2811 013746 004767 000552 JSR PC,00PS ;IF NO TIME-OUT, BAD PROGRAM OR HOST MACHINE.
2812 013752 60$: PASS ;RESTORE GPRS.
013752 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2813 013754 000207 RTS PC
2814
2815 013756 177777 62$: .WORD -1 ;DUMMY, NON ZERO WORD.

```

GLOBAL SUBROUTINE

- DODMA -

2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831  
2832  
2833  
2834  
2835  
2836  
2837  
2838  
2839  
2840  
2841  
2842  
2843  
2844  
2845  
2846  
2847  
2848  
2849  
2850  
2851  
2852  
2853  
2854  
2855  
2856  
2857  
2858  
2859  
2860  
2861  
2862  
2863  
2864  
2865  
2866  
2867  
2868  
2869  
2870  
2871  
2872

```

.SBTTL GLOBAL SUBROUTINE - DODMA -
;+ *****
;* Initiate DMA Transmission Routine -
;* This routine writes the DMA parameter to the specified device and
;* initiates the DMA transmission.
;*
;* INPUTS: R1 - Line number on which to initiate the DMA.
;* R2 - Start address of the DMA buffer (16 bit virtual).
;* R3 - Character count of the DMA buffer.
;* CSRA - Contains address of the DUT CSR.
;* IESTAT - Storage for states of the interrupt enable bits.
;* MMENAB - Memory management flag (0 if MEM MGT not enabled).
;* HOST MEM MGT PAR REGISTERS - If MEM MGT is in use.
;* TXAD1A - Contains address of DMA TX buffer address reg #1.
;* TXAD2A - Contains address of DMA TX buffer address reg #2.
;* TXBFCA - Contains address of DMA character count register.
;*
;* OUTPUTS: CARRY - Success flag (set if DMA_START found clear).
;* DUT TBUFFAD1 - LS 16 bits of DMA buffer address (initialized).
;* DUT TBUFFAD2 - MS 6 bits of DMA buffer address (initialized),
;* DMA START bit set.
;* DUT TBUFFCT - DMA buffer character count (initialized).
;*
;* CALLING SEQUENCE: JSR PC,DODMA
;*
;* COMMENTS: This routine determines if Memory Management is being used
;* and sets up the full 22 bit physical address if necessary.
;*
;* SUBORDINATE ROUTINES CALLED: None.
;-- *****
DODMA:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV #200,R4 ;PREPARE TO CLEAR UPPER 6 BITS OF DMA BUFF ADR.
TST MMENAB ;CHECK FOR MEMORY MANAGEMENT IN USE.
BEQ 6$ ;GOTO SET UP DEVICE IF MEM MGT NOT IN USE.
;+
; Memory management is in use.
; Construct 22 bit physical address from the 16 bit virtual address.
;-
MOV R2,R5 ;STRIP THE MOST SIGNIFICANT 3 BITS OF THE
MOV #5,R0 ; DMA BUFFER VIRTUAL ADDRESS AND MULTIPLY
2$: ROL R5 ; THEIR VALUE BY TWO TO GET AN OFFSET INTO
DEC R0 ; THE TABLE OF MEMORY MANAGEMENT PAGE
BNE 2$ ; ADDRESS REGISTERS (PAR).
BIC #177761,R5 ;
ADD PAR0A,R5 ;ADD IN THE BASE VALUE OF THE MM PAR REGISTERS.
MOV (R5),R5 ;GET THE 16 BIT PHYSICAL ADDRESS BLOCK COUNT.
MOV #6,R0 ;SHIFT UPPER 6 BITS OF THE PHYSICAL ADDRESS
4$: ASL R5 ; BLOCK COUNT (GOTTEN FROM THE PROPER PAR)
ROL R4 ; INTO THE LS 6 BITS OF THE WORD TO WRITE
DEC R0 ; INTO THE DUT TBUFFAD2 REGISTER.
BNE 4$ ;
BIC #160000,R2 ;ADD THE 13 BIT DISPLACEMENT FIELD FROM VIRTUAL
ADD R5,R2 ; ADR TO THE SHIFTED BLOCK NUMBER FROM THE
ADC R4 ; MEMORY MANAGEMENT PAR.
BIS #200,R4 ;SET THE DMA START BIT IN WORD FOR TBUFFAD2.

```

GLOBAL SUBROUTINE

- DODMA -

```

2873
2874      ;+
2875      ; Write the DMA parameters out to the DUT DMA registers.
2876      ; Disable interrupts.
2877      ; Set up DUT CSR IND.ADR.REG field.
2878      ; Write the DMA transmit character count.
2879      ; Write the least significant 16 bits of the DMA buffer start address.
2880      ; Write the most significant 6 bits of the address,
2881      ; setting the DMA_START bit, and initiating the DMA transmission.
2882      ;-
2882 6$: 014054 GETPRI R5 ;GET THE PRESENT PROCESSOR PRIORITY.
      014054 104440 TRAP C$GPRI
      014056 010005 MOV RO,R5
2883 014060 SETPRI #PRI07 ;DISABLE ALL HARDWARE INTERRUPTS.
      014060 012700 000340 MOV #PRI07,RO
      014064 104441 TRAP C$SPRI
2884 014066 056701 166176 BIS IESTAT,R1 ;PREPARE FOR SETUP OF LINE NUMBER IN DUT CSR.
2885 014072 010177 166144 MOV R1,@CSRA ;SET UP THE DUT CSR IND.ADR.REG FIELD.
2886 014076 105777 166154 TSTB @TXAD2A ;TEST THE DUT DMA_START BIT.
2887 014102 000241 CLC ;INDICATE FAILURE IN CASE DMA.HO BIT IS SET.
2888 014104 100411 BMI 60$ ;EXIT WITH FAILURE IF DMA.HO BIT IS SET.
2889 014106 010377 166146 MOV R3,@TXBFCA ;WRITE THE DMA CHARACTER COUNT.
2890 014112 010277 166136 MOV R2,@TXAD1A ;WRITE THE LS 16 BITS OF BUFFER ADDRESS.
2891 014116 110477 166134 MOVB R4,@TXAD2A ;WRITE MS 6 BITS OF ADR AND START DMA TX.
2892 014122 SETPRI R5 ;RESTORE THE PROCESSOR PRIORITY.
      014122 010500 MOV R5,RO
      014124 104441 TRAP C$SPRI
2893 014126 000261 SEC ;INDICATE SUCCESS.
2894
2895 60$: 014130 PASS ;RESTORE R5,
      014130 004736 PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2896 014132 000207 RTS PC JSP ; CARRY SUCCESS FLAG (SET IF SUCCESS).

```

GLOBAL SUBROUTINE

- FINACT -

```

2898 .SBTTL GLOBAL SUBROUTINE - FINACT -
2899 ;+ *****
2900 ;* - FIND FIRST ACTIVE LINE -
2901 ;* This subroutine calculates the number of the first active line that
2902 ;* is found in the active line bit map ACTLNS.
2903 ;*
2904 ;* INPUTS: ACTLNS - Contains the active line bit map.
2905 ;*
2906 ;* OUTPUTS: R1 - Contains the number of the first active line.
2907 ;* R5 - Contains the bit map representation of the active line.
2908 ;* Carry set indicates success.
2909 ;*
2910 ;* CALLING SEQUENCE: JSR PC,FINACT
2911 ;*
2912 ;* COMMENTS:
2913 ;*
2914 ;* SUBORDINATE ROUTINES CALLED: NONE.
2915 ;-- *****
2916
2917 014134 FINACT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
014134 004567 167754 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2918
2919 ;+
2920 ; Find an active line on which to perform the test.
2921 ;-
2921 014140 005001 CLR R1 ;CLEAR THE LINE NUMBER COUNTER.
2922 014142 012703 000010 MOV #NUMLNS,R3 ;GET MAX LINE NUMBER.
2923 014146 016700 166062 MOV ACTLNS,R0 ;GET THE ACTIVE LINE BIT MAP.
2924 014152 012705 000001 MOV #1,R5 ;SET UP A LINE BIT MASK.
2925 014156 030500 2$: BIT R5,R0 ;LOOK FOR AN ACTIVE LINE.
2926 014160 001006 BNE 4$ ;BRANCH TO BEGIN TEST IF A LINE HAS BEEN FOUND.
2927 014162 006305 ASL R5 ;SHIFT THE BIT MASK FOR THE NEXT LINE.
2928 014164 005201 INC R1 ;INCREMENT THE LINE NUMBER COUNTER.
2929 014166 020103 CMP R1,R3 ;CHECK IF ALL LINES HAVE BEEN TRIED.
2930 014170 002772 BLT 2$ ;LOOP TO TRY THE NEXT LINE.
2931 014172 000241 CLC ;CLEAR CARRY BIT, NO ACTIVE LINE FOUND.
2932 014174 000401 BR 60$ ;EXIT WITH FAILURE.
2933 014176 000261 4$: SEC ;SET CARRY, SUCCESS.
2934
2935 014200 60$: PASS R1,R5 ;RESTORE GPRS, EXCEPT
014200 010166 000004 MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
014204 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
014210 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2936 ;R1 - CONTAINS THE NUMBER OF FIRST ACTIVE LINE.
2937 ;R5 - CONTAINS THE BIT MAP OF THE ACTIVE LINE.
2938 ;CARRY - SET INDICATES SUCCESS.
2939 014212 000207 RTS PC

```

GLOBAL SUBROUTINE

- INDATP -

```

2941 .SBTTL GLOBAL SUBROUTINE - INDATP -
2942 ;* *****
2943 ;* - INITIALISE DATA PATTERN -
2944 ;* This subroutine is used to initialise an incremental byte data pattern
2945 ;* in the general buffer area.
2946 ;* The data pattern will be sequential from 0 to 255 (decimal).
2947 ;*
2948 ;* INPUTS: BUFBAS - Address of the start of the general buffer area.
2949 ;* BUFMID - Address of the 255 th location.
2950 ;*
2951 ;* OUTPUTS: The first 255 locations of the general buffer area contain data
2952 ;*
2953 ;* CALLING SEQUENCE: JSR PC,INIDATP
2954 ;*
2955 ;* COMMENTS:
2956 ;*
2957 ;* SUBORDINATE ROUTINES CALLED: NONE.
2958 ;*
2959 ;* *****
2960 014214 004567 167674 INDATP:: SAVE JSR ;SAVE CONTENTS OF GPRS R0 THRU R5.
014214 ; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2961
2962 014220 012702 002704 MOV #BUFBAS,R2 ;INITIALIZE THE DATA PATTERN IN THE GENERAL
2963 014224 005003 CLR R3 ; DATA BUFFER TO A 256 BYTE PATTERN.
2964 014226 110322 2$: MOVB R3,(R2)+ ;
2965 014230 005203 INC R3 ;SELECT THE NEXT CHARACTER.
2966 014232 020227 003304 CMP R2,#BUFMID ;CHECK IF WE HAVE 256 DATA PATTERNS.
2967 014236 103773 BLO 2$ ;
2968
2969 014240 004736 60$: PASS JSR ;RESTORE GPRS.
014240 ; PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2970 014242 000207 RTS PC

```

GLOBAL SUBROUTINE

- INDTPX -

```

2972 .SBTTL GLOBAL SUBROUTINE - INDTPX -
2973 ;* *****
2974 ;* - INITIALISE DATA PATTERN WITHOUT XON OR XOFF -
2975 ;* This subroutine is used to initialise an incremental byte data pattern
2976 ;* in the general buffer area.
2977 ;* The data pattern will be from 0 to 255, but will exclude the following
2978 ;* two characters; (ASCII DC1, DC3) XON AND XOFF. This will cause the
2979 ;* last two data characters to be the same as the first two.
2980 ;*
2981 ;* INPUTS: BUFBAS - Address of the start of the general buffer area.
2982 ;* BUFMID - Address of the 255 th location.
2983 ;*
2984 ;* OUTPUTS: The first 255 locations of the general buffer area contain data
2985 ;*
2986 ;* CALLING SEQUENCE: JSR PC,INDTPX
2987 ;*
2988 ;* COMMENTS:
2989 ;*
2990 ;* SUBORDINATE ROUTINES CALLED: NONE.
2991 ;*
2992 ;* *****
2993 014244 INDTPX:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
014244 004567 167644 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2994 ;*
2995 ; Initialize the 256 byte data pattern.
2996 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
2997 ; Note: the first two characters and the last two characters will be the same.
2998 ;*
2999 014250 012702 002704 MOV #BUFBAS,R2 ;INITIALIZE THE DATA PATTERN IN THE GENERAL
3000 014254 005003 CLR R3 ; DATA BUFFER TO A 256 BYTE PATTERN.
3001 014256 110322 2$: MOVB R3,(R2)+ ;
3002 014260 105203 INCB R3 ;SELECT THE NEXT CHARACTER.
3003 014262 122703 000021 CMPB #21,R3 ;CHECK FOR AN XON CHARACTER.
3004 014266 001001 BNE 4$ ;BRANCH IF CHAR NOT AN XON.
3005 014270 105203 INCB R3 ;FORCE THE NEXT CHARACTER.
3006 014272 122703 000023 4$: CMPB #23,R3 ;CHECK FOR AN XOFF CHARACTER.
3007 014276 001001 BNE 6$ ;BRANCH IF NOT AN XOFF CHARACTER.
3008 014300 105203 INCB R3 ;FORCE THE NEXT CHARACTER.
3009 014302 020227 003304 6$: CMP R2,#BUFMID ;CHECK IF WE HAVE 256 DATA PATTERNS.
3010 014306 103763 BLO 2$ ;
3011
3012 014310 60$: PASS ;RESTORE GPRS.
014310 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3013 014312 000207 RTS PC

```

GLOBAL SUBROUTINE

- LINBIT -

```

3015 .SBTTL GLOBAL SUBROUTINE - LINBIT -
3016 ;** *****
3017 ;* - Line Number to Bit Map conversion subroutine -
3018 ;* This subroutine is used to generate a bit map (one bit of 16 set)
3019 ;* based on a line number (range: 1 to 16). Only the LS 4 bits of the
3020 ;* line number word are used, the others are masked out (so unmasked
3021 ;* MSBytes of DUT CSRs can be passed to this routine without error).
3022 ;*
3023 ;* INPUTS: R1 - Line number (only LS 4 bits used, others disregarded).
3024 ;* BITTBL - Base label of a 16 word bit table.
3025 ;*
3026 ;* OUTPUTS: R0 - Bit map, bit corresponding to line number is set:
3027 ;* If line number is 3, then bit3 is set, etc.
3028 ;*
3029 ;* CALLING SEQUENCE: JSR PC,LINBIT
3030 ;*
3031 ;* COMMENTS: No checking is performed to verify that the line number is
3032 ;* a legal line number for the DUT (ie - less than NUMLNS).
3033 ;* NOTE: The line number is not destroyed or altered, so this
3034 ;* routine can be used easily in loops.
3035 ;*
3036 ;* SUBORDINATE ROUTINES CALLED: None.
3037 ;-- *****
3038
3039 014314 LINBIT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
3040 014314 004567 167574 ;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3041 014320 042701 177760 BIC #177760,R1 ;MASK OUT ALL BUT 4 LSBITS OF THE LINE #.
3042 014324 006301 ASL R1 ;MULTIPLY LINE # BY 2 TO GET WORD TABLE OFFSET.
3043 014326 016100 002370 MOV BITTBL(R1),R0 ;GET THE SINGLE BIT BIT MAP.
3044 014332 010066 000002 60$: PASS R0 ;RESTORE GPRS, EXCEPT THE FOLLOWING.
3045 014336 004736 MOV R0,ROSLOT(SP) ;PUT R0 IN STACK SLOT.
3046 014340 000207 RTS PC JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
;R0 - BIT MAP WITH LINE # BIT SET.

```



GLOBAL SUBROUTINE

- MAPCNT -

```

3046 .SBTTL GLOBAL SUBROUTINE - MAPCNT -
3047 ;** *****
3048 ;* - Count Bits in Bit Map Routine -
3049 ;* This subroutine counts the number of bits which are set in a bit map.
3050 ;*
3051 ;* INPUTS: R2 - The bit map for which to count the bits.
3052 ;*
3053 ;* OUTPUTS. R2 - Count of the number of bits that were set.
3054 ;*
3055 ;* CALLING SEQUENCE: JSR PC,MAPCNT
3056 ;*
3057 ;* COMMENTS:
3058 ;*
3059 ;* SUBORDINATE ROUTINES CALLED: None.
3060 ;-- *****
3061
3062 014342 MAPCNT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
014342 004567 167546 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3063 014346 010201 MOV R2,R1
3064 014350 001405 BEQ 60$ ;EXIT WITH ZERO IF NO BITS ARE SET IN MAP.
3065
3066 014352 005002 CLR R2 ;CLEAR THE BIT COUNT.
3067 014354 000261 SEC ;COUNT THE LAST BIT TO BE SHIFTED OUT.
3068
3069 014356 005502 2$: ADC R2 ;COUNT THE BIT IF IT WAS SET.
3070 014360 006301 ASL R1 ;SHIFT ANOTHER BIT OUT OF THE MAP.
3071 014362 001375 BNE 2$ ;LOOP IF ALL BITS NOT SHIFTED OUT OF MAP.
3072
3073 014364 010266 000006 60$: PASS R2 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
014364 010266 000006 MOV R2,R2SLOT(SP) ;PUT R2 IN STACK SLOT.
014370 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3074 014372 000207 RTS PC ; R2 - COUNT OF BITS SET IN BIT MAP.

```

GLOBAL SUBROUTINE

- MSLGET -

3076  
3077  
3078  
3079  
3080  
3081  
3082  
3083  
3084  
3085  
3086  
3087  
3088  
3089  
3090  
3091  
3092  
3093  
3094  
3095  
3096  
3097  
3098  
3099  
3100  
3101  
3102  
3103  
3104  
3105  
3106  
3107  
3108  
3109  
3110  
3111  
3112  
3113  
3114 014374  
014374 004567 167514  
3115  
3116  
3117  
3118  
3119 014400 005102  
3120 014402 040203  
3121  
3122  
3123  
3124 014404 005701  
3125 014406 001011  
3126 014410 011400  
3127 014412 010067 000070  
3128 014416 040200  
3129 014420 020003  
3130 014422 000261  
3131 014424 001420

```

.SBTTL GLOBAL SUBROUTINE - MSLGET -
;*****
;* - Milli Seconds Loop which returns read word and remaining time -
;* This subroutine is a general purpose test loop subroutine. It is used
;* to verify that a certain action occurs before a time-out period. The
;* calling routine passes in which bits should be set and cleared for the
;* desired condition and the time-out value in milli-seconds.
;* This routine checks for the desired condition upon entrance into the
;* routine and then once each milli-second thereafter.
;* Upon return, the last word which was read to check for the condition
;* is returned by this subroutine.
;*
;* INPUTS: R1 - Time-out value in milli-seconds (up to 64K ms).
;* R2 - Bit map of bits to test (1 indicates to test the bit).
;* R3 - Desired states of the indicated fields in R2.
;* R4 - Address of the word to test.
;* MSLCNT - Milli second software loop count.
;*
;* OUTPUTS: R0 - The last word which was read to check for the condition.
;* R1 - Remaining number of ms in time-out time.
;* CARRY - Success flag (set if condition is met before time-out).
;*
;* CALLING SEQUENCE: JSR PC,MSLGET
;*
;* COMMENTS: This routine works with or without a hardware clock, but the
;* calibration is only guaranteed when a line clock is available
;* on the system.
;* This routine can be used as a delay routine, by specifying the
;* desired delay as the time-out and specifying a condition to
;* look for which will not be met during the delay.
;* If a time-out value of 0 is specified, this routine checks for
;* the desired condition before returning. It indicates success
;* if the condition is met, failure otherwise.
;*
;* SUBORDINATE ROUTINES CALLED: None.
;*****
MSLGET:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;+
; Set up mask for removing unused bits in the test word, and clear unused
; bits in the desired state word to allow direct comparison.
;-
COM R2 ;GET MASK OF UNUSED BITS.
BIC R2,R3 ;MASK OUT UNUSED BITS IN DESIRED STATE WORD.
;+
; Handle the test and exit if we have a 0 time-out value.
;-
TST R1 ;TEST THE TIME-OUT VALUE FOR ZERO.
BNE 2$ ;IF NON-ZERO TIME-OUT, GO LOOP AND TEST.
MOV (R4),R0 ;GET THE WORD TO TEST BEFORE EXITING.
MOV R0,62$ ;SAVE VALUE SO WE CAN RETURN IT.
BIC R2,R0 ;MASK OUT UNTESTED BITS OF WORD.
CMP R0,R3 ;COMPARE AGAINST DESIRED STATE WORD.
SEC ;INDICATE SUCCESS IN CASE WORDS ARE EQUAL.
BEQ 6$ ;EXIT WITH SUCCESS IF WORDS ARE EQUAL.

```

GLOBAL SUBROUTINE

- MSLGET -

```

3132 014426 000241          CLC          ;INDICATE FAILURE (TIME-OUT).
3133 014430 000416          BR          6$          ;EXIT WITH FAILURE, WORDS AREN'T EQUAL.
3134                          ;+
3135                          ; Non-zero time-out value. Loop, waiting for condition or time-out.
3136                          ;-
3137 014432 016705 165702  2$:  MOV      MSLCNT,R5      ;LOAD MS LOOP COUNT.
3138 014436 011400          4$:  MOV      (R4),R0      ;GET THE WORD TO TEST.
3139 014440 010067 000042  MOV      R0,62$      ;SAVE WORD IN CASE THIS IS THE LAST.
3140 014444 040200          BIC      R2,R0      ;MASK OUT UNTESTED BITS OF WORD.
3141 014446 020003          CMP      R0,R3      ;COMPARE AGAINST DESIRED STATE WORD.
3142 014450 000261          SEC          ;SET CARRY IN CASE OF SUCCESS.
3143 014452 001405          BEQ      6$          ;EXIT WITH SUCCESS IF WORDS ARE EQUAL.
3144 014454 005305          DEC      R5          ;COUNT DOWN THE INSIDE MS LOOP COUNT.
3145 014456 001367          BNE      4$          ;LOOP IF MS NOT UP.
3146 014460 005301          DEC      R1          ;DECREMENT THE MS TIME COUNT.
3147 014462 001363          BNE      2$          ;IF TIME NOT UP, LOOP TO COUNT ANOTHER MS.
3148 014464 000241          CLC          ;CLEAR CARRY, WE TIMED-OUT.
3149                          ;+
3150                          ; Have either found condition, or timed-out (possibly from 0 time-out value).
3151                          ; Restore the last contents read from the test word. Exit routine.
3152                          ;-
3153 014466 016700 000014  6$:  MOV      62$,R0      ;PASS OUT THE LAST READ WORD.
3154 014472          60$:  PASS      R0,R1      ;RESTORE GPRS, EXCEPT THE FOLLOWING:
                                MOV      R0,ROSLOT(SP)      ;PUT R0 IN STACK SLOT.
                                MOV      R1,R1SLOT(SP)      ;PUT R1 IN STACK SLOT.
                                JSR      PC,@(SP)+          ;RETURN TO PREG05 SUBRT.
                                ;R0 - LAST READ WORD CHECKED FOR CONDITION.
                                ;R1 - REMAINING TIME (0 IF TIME-OUT OCCURED).
                                ;CARRY - SET IF SUCCESS, CLEAR IF TIME-OUT.
3155                          ;+
3156                          ; Local storage.
3157 014504 000207          RTS      PC
3158                          ;-
3159                          ;
3160                          ;
3161 014506 000000          62$:  .WORD  0          ;STORAGE FOR THE LAST READ WORD.

```

GLOBAL SUBROUTINE

- MSLOOP -

```

3163 .SBTTL GLOBAL SUBROUTINE - MSLOOP -
3164 ;*****
3165 ;* - Test Loop subroutine -
3166 ;* This subroutine is a general purpose test loop subroutine. It is used
3167 ;* to verify that a certain action occurs before a time-out period. The
3168 ;* calling routine passes in which bits should be set and cleared for the
3169 ;* desired condition and the time-out value in milli-seconds.
3170 ;* This routine checks for the desired condition upon entrance into the
3171 ;* routine and then once each milli-second thereafter.
3172 ;*
3173 ;* INPUTS: R1 - Time-out value in milli-seconds (up to 64K ms).
3174 ;* R2 - Bit map of bits to test (1 indicates to test the bit).
3175 ;* R3 - Desired states of the indicated fields in R2.
3176 ;* R4 - Address of the word to test.
3177 ;* MSLCNT - Milli second software loop count.
3178 ;*
3179 ;* OUTPUTS: CARRY - Success flag (set if condition is met before time-out).
3180 ;*
3181 ;* CALLING SEQUENCE: JSR PC,MSLOOP
3182 ;*
3183 ;* COMMENTS: This routine works with or without a hardware clock, but the
3184 ;* calibration is only guaranteed when a line clock is available
3185 ;* on the system.
3186 ;* This routine can be used as a delay routine, by specifying the
3187 ;* desired delay as the time-out and specifying a condition to
3188 ;* look for which will not be met during the delay.
3189 ;* If a time-out value of 0 is specified, this routine checks for
3190 ;* the desired condition before returning. It indicates success
3191 ;* if the condition is met, failure otherwise.
3192 ;*
3193 ;* SUBORDINATE ROUTINES CALLED: MSLGET.
3194 ;*****
3195
3196 014510 MSLOOP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
014510 004567 167400 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3197
3198 ;*
3199 ;* Calling the MSLGET routine from the MSLOOP routine isolates the caller of
3200 ;* MSLOOP from the returned test word and remaining time-out values.
3201 ;*
3202 014514 004767 177654 JSR PC,MSLGET ;CALL THE MULTI-PURPOSE MS LOOP AND SEARCH RTN.
3203
3204 014520 60$: PASS ;RESTORE GPRS,
014520 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3205 014522 000207 RTS PC ;CARRY SET IF SUCCESS, CLEAR IF TIME-OUT.

```

GLOBAL SUBROUTINE

- OOPS -

```

3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226 014524
      014524 004567 167364
3227
3228 014530
      014530 104454
      014532 000145
      014534 014570
      014536 000000
3229
3230 014540
      014540 012746 014654
      014544 012746 000001
      014550 010600
      014552 104417
      014554 062706 000004
3231 014560
      014560 104422
3232 014562 000776
3233 014564
      014564 004736
3234 014566 000207
3235
3236 014570 110 117 123
      014573 124 040 103
      014576 117 115 120
      014601 125 124 105
      014604 122 040 110
      014607 101 122 104
      014612 127 101 122
      014615 105 040 117
      014620 122 040 123
      014623 117 106 124
      014626 127 101 122
      014631 105 040 102
      014634 125 107 040
      014637 105 116 103
      014642 117 125 116
      014645 124 105 122

```

```

.SBTTL GLOBAL SUBROUTINE - OOPS -
; ** *****
;* - Program abort subroutine -
;* This subroutine is used to abort the program when a fatal error is
;* detected in the program or the host system hardware. An error message
;* is printed giving some information about the nature of the abort.
;*
;* INPUTS: R1 - Error code giving reason for abort.
;*
;* OUTPUTS: An error message is printed.
;* A list of return PC values for all subroutine calls is printed.
;*
;* CALLING SEQUENCE: JSR PC,OOPS
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: None.
; *****
OOPS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
      JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
; REPORT "HOST COMPUTER HARDWARE OR SOFTWARE BUG ENCOUNTERED." ERROR.
      ERRSF 101,EM0101
;
; REPORT "PROGRAM HUNG, WAITING FOR A CONTROL-C."
      PRINTF #EM0102
;
; LOOK FOR OPERATOR CONTROL-C INPUT.
2$: BREAK ;LOOK FOR OPERATOR CONTROL-C INPUT.
; INFINITE LOOP.
BR 2$ ;INFINITE LOOP.
; DON'T NEED THIS, BUT SOMEBODY MAY CHANGE THIS
60$: PASS ;DON'T NEED THIS, BUT SOMEBODY MAY CHANGE THIS
      JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
; ROUTINE IN THE FUTURE, SO BE CONSISTANT.
      RTS PC
EM0101:: .ASCIZ /HOST COMPUTER HARDWARE OR SOFTWARE BUG ENCOUNTERED./

```

```

TRAP C$ERSF
.WORD 101
.WORD EM0101
.WORD 0
MOV #EM0102,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #4,SP
TRAP C$BRK

```

GLOBAL SUBROUTINE

- OOPS -

	014650	105	104	056	
	014653	000			
3237	014654	045	116	045	EM0102:: .ASCIZ / <del>N</del> APROGRAM HUNG, WAITING FOR A CONTROL-C. <***** <del>N</del> <del>N</del> /
	014657	101	120	122	
	014662	117	107	122	
	014665	101	115	040	
	014670	110	125	116	
	014673	107	054	040	
	014676	127	101	111	
	014701	124	111	116	
	014704	107	040	106	
	014707	117	122	040	
	014712	101	040	103	
	014715	117	116	124	
	014720	122	117	114	
	014723	055	103	056	
	014726	040	074	052	
	014731	052	052	052	
	014734	052	052	052	
	014737	052	052	052	
	014742	052	052	052	
	014745	045	116	045	
3238	014750	116	000		.EVEN

GLOBAL SUBROUTINE

- PRTLPR -

```

3240 .SBTTL GLOBAL SUBROUTINE - PRTLPR -
3241 ;** *****
3242 ;* -Print the contents of the LPR.
3243 ;* This routine is used to print out extended information on the
3244 ;* contents of the Line Parameter Register (LPR).
3245 ;*
3246 ;* INPUTS: R3 - Contains the number of the line you wish to examine.
3247 ;* CSRA - Contains the address of the DUT's CSR.
3248 ;* IESTAT - Contains the current status of the TX and RX interrupt
3249 ;* enable bits in the DUT's CSR.
3250 ;* LPRA Contains the address of the DUT's LPR register.
3251 ;*
3252 ;* OUTPUTS: An extended information message is printed on the operators
3253 ;* console.
3254 ;*
3255 ;* CALLING SEQUENCE: JSR PC,PRTLPR
3256 ;*
3257 ;* COMMENTS: This routine changes the indirect address field of the device
3258 ;* under test's CSR.
3259 ;*
3260 ;* SUBORDINATE ROUTINES CALLED: NONE.
3261 ;-- *****
3262
3263 014752 PRTLPR::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
014752 004567 167136 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3264 014756 016701 165260 MOV CSRA,R1 ;GET THE CSR ADDRESS.
3265 014762 016702 165260 MOV LPRA,R2 ;GET THE LPR ADDRESS.
3266 014766 042703 177760 BIC #177760,R3 ;CLEAR ANY UNWANTED BITS.
3267 014772 056703 165272 BIS IESTAT,R3 ;SET STATE OF TX AND RX INTERRUPT ENABLE BITS.
3268 014776 010311 MOV R3,(R1) ;SELECT LINE.
3269 015000 011204 MOV (R2),R4 ;GET CONTENTS OF THE LPR.
3270 ;PRINT MESSAGE"CONTENTS OF THE LPR:nnnnn"
3271 015002 PRINTX #EF9019,#EM9026,R4;PRINT OUT MESSAGE ON OPERATORS CONSOLE.
015002 010446 MOV R4,-(SP)
015004 012746 011027 MOV #EM9026,(SP)
015010 012746 005276 MOV #EF9019,-(SP)
015014 012746 000003 MOV #3,-(SP)
015020 010600 MOV SP,R0
015022 104415 TRAP C$PNTX
015024 062706 000010 ADD #10,SP
3272 015030 60$: PASS ;RESTORE GPRS.
015030 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3273 015032 000207 RTS PC

```

GLOBAL SUBROUTINE

- PUFIFO -

```

3275 .SBTTL GLOBAL SUBROUTINE - PUFIFO -
3276 ;*****
3277 ;* - PURGE THE FIFO
3278 ;* This routine tries to remove all the characters from the FIFO.
3279 ;* Any BMP codes that are found are saved on the BMP code queue.
3280 ;*
3281 ;* INPUTS: RBUFA Contains the address of the Receiver.
3282 ;*
3283 ;*
3284 ;* OUTPUTS: Carry bit Indicates the state of the fifo, set:= purged.
3285 ;* BMPCQ - The contents of the BMP code queue may be updated.
3286 ;*
3287 ;* CALLING SEQUENCE: JSR PC,PUFIFO
3288 ;*
3289 ;* COMMENTS:
3290 ;*
3291 ;* SUBORDINATE ROUTINES CALLED: SAVBMP.
3292 ;*****
3293
3294 015034 PUFIFO::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015034 004567 167054 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3295 015040 012701 001000 MOV #512.,R1 ;SET MAXIMUM TRY COUNT OF 512.
3296 015044 016704 165174 MOV RBUFA,R4 ;GET ADDRESS OF THE RECEIVER BUFFER REGISTER.
3297
3298 015050 011402 2$: MOV (R4),R2 ;GET THE CONTENTS OF THE RECEIVER BUFFER REG.
3299 015052 100016 BPL 6$ ;EXIT IF THE FIFO IS EMPTY, DATA_VALID CLR.
3300
3301 ;+
3302 ; Check if the read character is actually a BMP code.
3303 ; If it is, then save it on the BMP code queue to be reported later.
3304 015054 012700 070000 MOV #70000,R0 ;GENERATE A BIT MAP OF CHAR ERROR BITS
3305 015060 040200 BIC R2,R0 ; WHICH ARE NOT SET FOR CHAR.
3306 015062 001006 BNE 4$ ;THROW CHAR AWAY IF NOT BMP OR SELFTEST CODE.
3307
3308 ;+
3309 ; Check if the read data is modem status , BMP or Selftest?.
3310 015064 012700 000300 MOV #300,R0 ; CHECK IF BMP OR SELFTEST?.
3311 015070 040200 BIC R2,R0 ;TRY TO CLEAR BMP FLAGS IN THE READ DATA.
3312 015072 001002 BNE 4$ ;IF IT IS MODEM OR SELFTEST CODE THROW IT AWAY.
3313 015074 004767 000444 JSR PC,SAVBMP ;SAVE BMP CODE ON THE QUEUE.
3314
3315 015100 005301 4$: DEC R1 ;DECREMENT THE TRY COUNT.
3316 015102 001362 BNE 2$ ;LOOP TO TRY AGAIN.
3317 015104 000241 CLC ;CLEAR CARRY,TO INDICATE FIFO NOT PURGED.
3318 015106 000401 BR 60$ ;EXIT WITH CARRY CLEAR.
3319 015110 000261 6$: SEC ;SET CARRY, TO INDICATE FIFO PURGED.
3320
3321 015112 004736 60$: PASS ;RESTORE GPRS,
015112 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3322 ;CARRY BIT, SET INDICATES FIFO PURGED.
3323 015114 000207 RTS PC

```



GLOBAL SUBROUTINE

- PUFIFR -

```

3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353 015116
      015116 004567 166772
3354 015122 016746 166760
3355 015126 012705 001000
3356
3357
3358
3359
3360 015132 017702 165106
3361 015136 100057
3362
3363
3364
3365 015140 012700 070000
3366 015144 040200
3367 015146 001012
3368
3369
3370
3371
3372 015150 012767 012334 166734
3373 015156 012700 000300
3374 015162 040200
3375 015164 001003
3376 015166 004767 000352
3377 015172 000424
3378
3379
3380

```

```

.SBTTL GLOBAL SUBROUTINE - PUFIFR -
;*****
;* - Purge FIFO report any errors found.
;* This routine removes all data from the FIFO. Any BMP codes that are
;* found are save on the queue to be reported later in the BMP report test.
;* Any unexpected data (ie any non-status inforamtion) that are found,
;* are reported as an error.
;* If the FIFO will not purge after 512 attempts, then the current test
;* that called this routine receives a failure flag that should be used
;* to abort the test.
;*
;* INPUTS: ERRTBL - ERRTYPE, ERRMSG, ERRNBR are set up correctly.
;* RBUFA- Contains the address of the Receiver.
;*
;* OUTPUTS: Carry bit - Abort test flag, Clr = ABORT TEST, Set = OK.
;* ERRBLK - Value will be destroyed.
;* BMPCQP - The BMP code queue pointer may be updated.
;* The contents of the BMP code queue may be udated.
;*
;* CALLING SEQUENCE: JSR PC,PUFIFR
;*
;* COMMENTS: This routine reports errors with numbers initial ERRNBR
;* thru to ERRNBR+2.
;* The ERRNBR is restored to its INITIAL value before returning.
;*
;* SUBORDINATE ROUTINES CALLED: ER1603,ER9001,ER9002,SAVBMP.
;*****
PUFIFR::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
      MOV ERRNBR, -(SP) ;R5, PREG05 ;CALL REGISTER SAVE SUBRT.
      MOV #512, R5 ;SAVE THE CONTENTS OF THE ERROR NUMBER.
      ;SET MAXIMUM READ COUNTER TO 2*FIFO SIZE.
;+
; Read data from the FIFO until DATA VALID is clear of read counter is zero.
; Report any BMP or Unexpected data as errors.
;-
2$: MOV @RBUFA, R2 ;GET THE CONTENTS OF THE RECEIVER BUFFER REG.
    BPL 8$ ;EXIT IF DATA VALID CLEAR, ie. FIFO PURGED.
;+
; Check if read data is status or unexpected character.
;-
      MOV #70000, R0 ;GENERATE A BIT MAP OF CHAR ERROR BITS
      BIC R2, R0 ; WHICH ARE NOT SET FOR CHAR.
      BNE 4$ ;SKIP BMP CHECK IF IT IS UNEXPECTED DATA.
;+
; Check if the read data is modem status , BMP or Selftest?.
; If it is a BMP code then save it on the queue.
;-
      MOV #ER9001, ERRBLK ;SET UP THE CORRECT ERROR REPORTING ROUTINE.
      MOV #300, R0 ; CHECK IF BMP OR SELFTEST?.
      BIC R2, R0 ;TRY TO CLEAR BMP FLAGS IN THE READ DATA.
      BNE 4$ ;SKIP BMP ERROR REPORT IF MODEM OR SELFTEST?.
      JSR PC, SAVBMP ;SAVE THE BMP CODE ON THE QUEUE.
      BR 6$ ;BRANCH TO CHECK READ COUNT.
;+
; Check if the read data is Modem, Selftest or Unexpected data.
;

```

GLOBAL SUBROUTINE

- PUFIFR -

```

3381 015174 032702 000001      4$:   BIT    #BIT0,R2      ;TEST THE MODEM STATUS INDICATION BIT.
3382 015200 001421              BEQ    6$              ;DO NOT REPORT ANY ERROR IF MODEM STATUS.
3383 015202 012701 011053      MOV    #EM9104,R1     ;PASS THE CORRECT ERROR MESSAGE TO REPORT.
3384 015206 010203              MOV    R2,R3          ;EXTRACT THE LINE NUMBER FROM
3385 015210 000303              SWAB   R3              ; THE READ DATA.
3386 015212 042703 177760      BIC    #177760,R3     ;
3387 015216 006303              ASL    R3              ;FORM LINE NUMBER TIMES 2 FOR ER9002 ROUTINE.
3388 015220 052704 100000      BIS    #BIT15,R4     ;SET THE "NONE" EXPECTED MAESSAGE FLAG.
3389 015224 005267 166656      INC    ERRNBR         ;SET ERROR NUMBER TO INTIAL ERRBR+1.
3390 015230 012767 012424 166654  MOV    #ER9002,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
3391                                ;REPORT ERROR "UNEXPECTED DATA FOUND IN FIFO".
3392 015236              ERROR ;
3393 015240 005367 166642      DEC    ERRNBR         ;RESTORE ERROR NUMBER TO INTIAL ERRNBR.
3394                                ;
3395 015244 005305      6$:   DEC    R5              ;DECREMENT READ COUNTER.
3396 015246 001331              BNE    2$              ;LOOP TO READ NEXT CHAR FROM FIFO IF COUNT > 0.
3397                                ;
3398                                ;+
3399                                ; The FIFO will not clear, report the error and indicate that the test is to
3400                                ; be ABORTED.
3401 015250 062767 000002 166630      ADD    #2,ERRNBR     ;SET ERROR NUMBER TO INTIAL ERRNBR+2.
3402 015256 012767 011662 166626      MOV    #ER1603,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
3403 015264 012701 010716      MOV    #EM9017,R1     ;PASS THE MESSAGE TO BE REPORTED.
3404                                ;REPORT THE ERROR "FIFO WILL NOT PURGE, (DATA VALID STUCK SET)"
3405                                ;"?????? TEST ABORTED!".
3406 015270              ERROR ;
3407 015272 000241              CLC                    ;INDICATE THE TEST IS TO BE ABORTED.
3408 015274 000401              BR     10$             ;EXIT THIS ROUTINE AND ABORT THE CURRENT TEST.
3409                                ;
3410 015276 000261      8$:   SEC                    ;SET THE CARRY, DO NOT ABORT THE TEST.
3411                                ;
3412 015300 012667 166602      10$:  MOV    (SP)+,ERRNBR ;RESTORE INITIAL ERROR NUMBER.
3413 015304 004736      60$:  PASS                    ;RESTORE GPRS,
3414                                JSR    PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3415                                ;CARRY BIT, SET INDICATES FIFO PURGED, DO NOT
3416 015306 000207      RTS    PC              ; ABORT THE TEST.

```

GLOBAL SUBROUTINE

- READBX -

```

3418 .SBT; GLOBAL SUBROUTINE - READBX -
3419 ;* *****
3420 ;* READ CHARACTERS FROM THE FIFO AND CHECKS FOR BMPS AND XONS-
3421 ;* It's subroutine is used in the FIHAVL.TST.
3422 ;* It reads the specified number of characters from the fifo and checks
3423 ;* for BMP codes and XON characters.
3424 ;*
3425 ;* INPUTS: R0 - Contains the number of chars to read from the fifo.
3426 ;*
3427 ;* OUTPUTS: R1 - Contains address of error message to be reported
3428 ;* Clear if no error found.
3429 ;* Carry used to indicate if fifo was found empty, carry clear.
3430 ;*
3431 ;* CALLING SEQUENCE: JSR PC,READ
3432 ;*
3433 ;* COMMENTS:
3434 ;*
3435 ;* SUBORDINATE ROUTINES CALLED: CHKBMP.
3436 ;* - *****
3437
3438 015310 READBX:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015310 004567 166600 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3439 015314 005001 CLR R1 ;CLEAR GPR THAT HOLDS THE ADDRESS OF ERRMSG.
3440 015316 016703 164722 MOV RBUFA,R3 ;GET THE ADDRESS OF THE RECEIVER BUFFER REG.
3441 015322 011302 2$: MOV (R3),R2 ;READ A CHARACTER FROM THE FIFO.
3442 015324 100015 BPL 8$ ;BRANCH IF FIFO IS EMPTY.
3443 ;*
3444 ; Check if the read character is a BMP code.
3445 ; If it is a BMP code save it on the queue to be reported later, and
3446 ; abort the test.
3447 ;*
3448 015326 004767 176054 JSR PC,CHKBMP ;CHECK IF CHARACTER IS A BMP CODE.
3449 015332 103410 BCS 6$ ;BRANCH IF A BMP CODE WAS FOUND.
3450 015334 120227 000021 CMPB R2,#21 ;CHECK IF IT IS AN XON.
3451 015340 001003 BNE 4$ ;BRANCH IF NOT AN XON.
3452 015342 012701 006744 MOV #EM5402,R1 ;PASS THE MESSAGE TO BE REPORTED.
3453 015346 000402 BR 6$ ;GO EXIT TEST.
3454 015350 005300 4$: DEC R0 ;DECREMENT THE READ COUNT.
3455 015352 001363 BNE 2$
3456 015354 000261 6$: SEC ;SET CARRY TO INDICATE SUCCESS.
3457 015356 000401 BR 60$ ;EXIT
3458 015360 000241 8$: CLC ;CLEAR CARRY BIT TO INDICATE FAILURE.
3459
3460 015362 60$: PASS R1 ;RESTORE GPRS,
015362 010166 000004 MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
015366 004736 JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
3461 015370 000207 RTS PC

```

GLOBAL SUBROUTINE

- RESETT -

```

3463 .SBTTL GLOBAL SUBROUTINE - RESETT -
3464 ;*****
3465 ;* - Reset Device Under Test -
3466 ;* This subroutine is used to reset the DUT to a known state.
3467 ;* If reset does not successfully complete, ie. time-out occurs, then
3468 ;* an abort test error message is reported.
3469 ;*
3470 ;* INPUTS: CSRA - Contains the address of the CSR
3471 ;* TXBFCA - Contains address of DUT DMA Buffer Count register.
3472 ;* ERRTBL- ERRTYP,ERNBR,and ERRMSG set up correctly.
3473 ;*
3474 ;* OUTPUTS: The DUT performs its reset function into a known state.
3475 ;* CARRY - Clear indicates the test is to be aborted.
3476 ;* ERRBLK - value may be destroyed.
3477 ;* IESTAT - TX and RX interrupt flags are cleared.
3478 ;* TX and RX interrupt enable bits in the DUT's CSR are cleared.
3479 ;*
3480 ;* CALLING SEQUENCE: JSR PC,RESETT
3481 ;*
3482 ;* COMMENTS: This subroutine can report errors with numbers initial ERRNBR
3483 ;* This routine does not destroy the value of ERRNBR.
3484 ;*
3485 ;* SUBORDINATE ROUTINES CALLED: DELAY,MSLGET.
3486 ;*****
3487
3488 015372 RESETT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015372 004567 166516 ;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3489 015376 012702 000040 MOV #BIT05,R2 ;SET BIT MASK OF MASTER RESET BIT.
3490
3491 ;*
3492 ; Test the state of the master reset bit in the CSR.
3493 ; If MR is set then wait for self-test to complete.
3494 ; If time-out occurs, report the error and pass-out abort test indicator.
3495 015402 016704 164634 ;-
3496 015406 030214 MOV CSRA,R4 ;GET THE ADDRESS OF THE DUT'S CSR.
3497 015410 001406 BIT R2,(R4) ;CHECK STATE OF MASTER RESET BIT.
3498 015412 005003 BEQ 2$ ;DON'T DELAY IF MR IS ALREADY CLEAR.
3499 015414 012701 004704 CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
3500 015420 004767 176750 MOV #2500.,R1 ;PASS TIME-OUT VALUE OF 2.5 SECONDS.
3501 015424 103012 JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
3502 BCC 4$ ;GO REPORT ERROR IF TIMEOUT OCCURRED.
3503
3504 ;*
3505 ; Set Master Reset bit in CSR. Clear TX and RX enable bits, etc.
3506 ; Skip the selftest.
3507 ; Time-out of 2.5 secs, just in case the self-test executes.
3508 015426 010277 164610 2$: MOV R2,@CSRA ;SET MASTER RESET BIT, DISABLE TX AND RX INTS.
3509 015432 004767 000266 JSR PC,SKPSTS ;TRY TO SKIP THE SELFTEST.
3510
3511 ;*
3512 ; Set Self-test time-out of 2.5 seconds, and wait for M.R to clear.
3513 ; If Time-out occurs, then report the fatal error and pass-out the abort
3514 ; test indicator.
3515 015436 005003 ;-
3516 015440 012701 004704 CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
3517 015444 004767 176724 MOV #2500.,R1 ;PASS TIME-OUT VALUE OF 2.5 SECONDS.
3518 015450 103410 JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
BCS 6$ ;SKIP ERROR REPORT IF MR CLEARED IN TIME.

```

GLOBAL SUBROUTINE

- RESETT -

```

3519
3520 ;+
3521 ; Set up error message to report "fatal error found during reset, test aborted".
3522 ; Indicate test is to be aborted by clearing the carry bit.
3523 015452 012701 005711 ;-
3524 015456 012767 011662 166426 4$: MOV #EM1601,R1 ;PASS ERROR MESSAGE TO REPORT.
3525 ;MOV #ER1603,ERRBLK ;PASS ADDRESS OF ERROR HANDLING ROUTINE.
3526 ;REPORT ERROR "TIME OUT OCCURRED WAITING FOR MASTER RESET TO CLEAR"
3527 015464 ; "TEST ABORTED"
015464 104460 ; ERROR >>>> ERROR <<<<< TRAP C$ERROR
3528 015466 000241 CLC ;INDICATE TEST IS TO BE ABORTED.
3529 015470 000403 BR 60$ ;EXIT THIS SUBROUTINE, ABORT TEST INDICATOR.
3530 ;+
3531 ; Clear TX and RX Interrupt enable status flags in IESTAT.
3532 ; Exit with continue test indicator set (ie, carry set).
3533 ;-
3534 015472 005067 164572 6$: CLR IESTAT ;CLEAR TX AND RX INTERRUPT STATUS FLAGS.
3535 015476 000261 SEC ;INDICATE SUCCESS, CONTINUE TEST.
3536
3537 015500 60$: PASS ;RESTORE GPRS, PASS THE FOLLOWING INTACT:
015500 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3538 ;CARRY BIT:IF CLEAR,INDICATES ABORT TEST.
3539 015502 000207 RTS PC
3540

```

GLOBAL SUBROUTINE

- RXIEO -

```

3542 .SBTTL GLOBAL SUBROUTINE - RXIEO -
3543 ;* *****
3544 ;* - RECEIVER INTERRUPT DISABLE -
3545 ;* This routine is used to disable receiver interrupts in the DHV11.
3546 ;*
3547 ;* INPUTS: NONE.
3548 ;*
3549 ;* OUTPUTS: The RX.INT.ENBL bit is cleared in the DUT CSR.
3550 ;* IESTST -contains the updated status of the TX and RX interrupt
3551 ;* enable bits.
3552 ;*
3553 ;* CALLING SEQUENCE: JSR PC,RXIEO
3554 ;*
3555 ;* COMMENTS: The contents of the indirect address register field in
3556 ;* the DUT CSR are destroyed.
3557 ;*
3558 ;* SUBORDINATE ROUTINES CALLED: NONE.
3559 ;*-- *****
3560 015504 010046 RXIEO:: MOV RO,-(SP) ;SAVE CONTENTS OF RO ON THE STACK.
3561 015506 GETPRI -(SP) ;SAVE PROCESSOR PRIORITY ON STACK.
015506 104440 TRAP C$GPRI
015510 010046 MOV RO,-(SP)
3562 015512 SETPRI #PRI07 ;IGNORE ANY INTERRUPT THAT MAY BE GENERATED.
015512 012700 000340 MOV #PRI07,RO
015516 104441 TRAP C$SPRI
3563 015520 042767 137777 164542 BIC #137777,IESTAT ;CLEAR RX.INT.ENBL BIT IN IESTAT.
3564 015526 016777 164536 164506 MOV IESTAT,@CSRA ;DISABLE RX INTERRUPTS.
3565 015534 SETPRI (SP)+ ;ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
015534 012600 MOV (SP)+,RO
015536 104441 TRAP C$SPRI
3566 015540 012600 MOV (SP)+,RO
3567 015542 000207 RTS PC ;RESTORE RO.

```

GLOBAL SUBROUTINE

- SAVBMP -

```

3569 .SBTTL GLOBAL SUBROUTINE - SAVBMP -
3570 ;+ *****
3571 ;* - Save BMP codes Routine -
3572 ;* This routine saves the parameter passed in, onto the BMP code queue
3573 ;* together with the number of the currently executing test.
3574 ;*
3575 ;* INPUTS: R2 - Contains the BMP code that is to be placed on the queue.
3576 ;* BMPCQP - Contains address of next location in the bmp queue.
3577 ;* BMPCQB - Label at base of the BMP code queue.
3578 ;* BMPCQE - Label of next location after the end of the BMP queue.
3579 ;* TSTNUM Contains the number of the current test.
3580 ;*
3581 ;* OUTPUTS: BMPCQP - Incremented by 4.
3582 ;* The contents of the BMP code queue are updated.
3583 ;*
3584 ;* CALLING SEQUENCE: JSR PC,SAVBMP
3585 ;*
3586 ;* COMMENTS: If the overflow occurs then the last location will be
3587 ;* overwritten by any subsequent attempts to update the queue.
3588 ;*
3589 ;* SUBORDINATE ROUTINES CALLED: None.
3590 ;-- *****
3591
3592 015544 SAVBMP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015544 004567 166344 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3593 015550 016704 164666 MOV BMPCQP,R4 ;GET THE POINTER TO THE NEXT LOCATION IN QUEUE.
3594 015554 116724 164506 MOV TSTNUM,(R4)+ ;SAVE THE CURRENT TEST NUMBER ON THE QUEUE.
3595 015560 005204 INC R4 ;INCREMENT THE POINTER TO GIVE AN EVEN ADDRESS.
3596 015562 042702 177400 BIC #177400,R2 ;CLEAR THE UNWANTED BITS FROM THE BMP CODE.
3597 015566 010224 MOV R2,(R4)+ ;SAVE THE BMP CODE ON THE QUEUE.
3598 015570 020427 002644 CMP R4,#BMPCQE ;CHECK IF OVERFLOW WILL OCCUR THE NEXT TIME.
3599 015574 103402 BLO 2$ ;GO SAVE THE POINTER IF WE WILL NOT OVERFLOW.
3600 015576 162704 000004 SUB #4,R4 ;RESET THE POINTER TO THE LAST LOCATION IN QUE.
3601 015602 010467 164634 2$: MOV R4,BMPCQP ;SAVE THE POINTER.
3602
3603 015606 60$: PASS ;RESTORE GPRS.
015606 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3604 015610 000207 RTS PC

```

GLOBAL SUBROUTINE

- SAVMST -

```

3606 .SBTTL GLOBAL SUBROUTINE - SAVMST -
3607 ;* *****
3608 ;* - Save Modem Status Routine -
3609 ;* This routine saves the present contents of the DUT STAT registers in
3610 ;* the STAT storage table.
3611 ;*
3612 ;* INPUTS: CSRA - Contains the address of the DUT CSR.
3613 ;* IESTAT - State of the DUT CSR interrupt enable bits.
3614 ;* NUMLNS - Equated to the number of lines on the DUT.
3615 ;* STATA - Contains the address of the DUT STAT register.
3616 ;* STSTB - Label at base of the STAT storage table.
3617 ;*
3618 ;* OUTPUTS: STST Table - Overwritten with present STAT contents.
3619 ;* CSR IND.ADR.REG field - Destroyed.
3620 ;*
3621 ;* CALLING SEQUENCE: JSR PC,SAVMST
3622 ;*
3623 ;* COMMENTS: If the contents of IESTAT changes during this test the CSR
3624 ;* interrupt enable bits will not track the change.
3625 ;*
3626 ;* SUBORDINATE ROUTINES CALLED: None.
3627 ;*
3628 ;*
3629 SAVMST:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
3630 015612 004567 166276 ;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3631 015616 016701 164446 MOV IESTAT,R1 ;GET IE STATES FOR UPDATING IND.ADR.REG FIELD.
3632 015622 012702 002644 MOV #STSTB,R2 ;SET UP STAT STORAGE POINTER TO BASE OF TABLE.
3633 015626 012703 000010 MOV #NUMLNS,R3
3634 015632 050103 BIS R1,R3 ;FORM COMPLETION COMPARISON WORD.
3635 015634 010177 164402 2$: MOV R1,@CSRA ;SET UP THE CSR IND.ADR.REG FIELD.
3636 015640 017722 164404 MOV @STATA,(R2)+ ;SAVE CONTENTS OF THIS LINE'S STAT REGISTER.
3637 015644 005201 INC R1 ;SET LINE COUNTER TO NEXT LINE.
3638 015646 020103 CMP R1,R3 ;CHECK FOR ALL LINES DONE.
3639 015650 002771 BLT 2$ ;LOOP IF NOT ALL LINES DONE.
3640 015652 004736 60$: PASS ;RESTORE GPRS.
3641 015654 000207 RTS PC JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

```



GLOBAL SUBROUTINE

- SETPAR -

```

3643 .SBTTL GLOBAL SUBROUTINE - SETPAR -
3644 ;** *****
3645 ;* - SET TX AND CONTROL PARAMETERS -
3646 ;* This suroutine is used in the FIHAVL.TST.
3647 ;* It initialises the selected line to the following state:
3648 ;* Internal loopback, IAUTO enabled, LPR:38.4k, 8 bits/char, 2 stop,
3649 ;* odd parity.
3650 ;*
3651 ;* INPUTS: R1 - Contains number of the line to be initialised.
3652 ;*
3653 ;* OUTPUTS: LNCTRL and LPR registers for the selected line are destroyed.
3654 ;*
3655 ;* CALLING SEQUENCE: JSR PC,SETPAR
3656 ;*
3657 ;* COMMENTS:
3658 ;*
3659 ;* SUBORDINATE ROUTINES CALLED: DELAY,WTWLNC,WTWLPR.
3660 ;-- *****
3661
3662 015656 SETPAR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015656 004567 166232 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3663 015662 004767 176426 JSR PC,LINBIT ;GET A BIT MAP FOR THIS LINE.
015666 010005 MOV R0,R5 ;COPY THE LINE BIT MAP.
3665 015670 012700 000206 MOV #206,R0 ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.
3666 015674 004767 001012 JSR PC,WTWLNC ;INITILAISE THE LINE CONTROL REGISTER.
3667 015700 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
3668 015704 004767 001032 JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
3669 015710 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
3670 015714 004767 176000 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
3671
3672 015720 60$: PASS ;RESTORE GPRS.
015720 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3673 015722 000207 RTS PC

```

GLOBAL SUBROUTINE

- SKPSTS -

```

3675 .SBTTL GLOBAL SUBROUTINE - SKPSTS -
3676 ;* *****
3677 ;* - Skip Selftest Routine -
3678 ;* This subroutine is used to skip the selftest after a DUT reset has been
3679 ;* initiated. It must be entered immediately after setting the DUT Master
3680 ;* Reset routine or after the execution of a bus reset (because of timing
3681 ;* considerations).
3682 ;*
3683 ;* INPUTS: CSRA - Contains address of the DUT CSR.
3684 ;* TXBFCA - Contains address of DUT DMA Buffer Count register.
3685 ;*
3686 ;* OUTPUTS: Skip selftest codes are written to the DUT registers.
3687 ;*
3688 ;* CALLING SEQUENCE: JSR PC,SKPSTS
3689 ;*
3690 ;* COMMENTS:
3691 ;*
3692 ;* SUBORDINATE ROUTINES CALLED: DELAY.
3693 ; - *****
3694
3695 015724 SKPSTS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
015724 004567 166164 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3696 015730 012704 000012 MOV #10,R4 ;PASS DELAY VALUE OF 10 MILLI-SECONDS.
3697 015734 004767 175750 JSR PC,DELAY ;DELAY FOR 10 MILLI-SECONDS.
3698 ;*
3699 ;* Write skip self-test code (52525) to all the indexed DUT Registers.
3700 ; -
3701 015740 012701 000050 MOV #NUMLNS!BIT05,R1 ;FORM IND.ADR.REG FIELD (PLUS M.R. BIT) WORD.
3702 ;THE ABOVE INCLUSION OF THE M.R. BIT IS NECESSARY BECAUSE OF THE
3703 ; LACK OF A M.R. BIT WRITE LOCK-OUT ON THE DHV11-M.
3704 015744 012703 052525 MOV #52525,R3 ;INITIALISE THE SKIP SELF-TEST CODE.
3705 015750 005301 4$: DEC R1 ;SELECT THE NEXT SET OF DEVICE REGISTERS.
3706 015752 016704 164264 MOV CSRA,R4 ;GET THE ADDRESS OF THE CSR OF THE DUT.
3707 015756 010124 MOV R1,(R4)+ ;SELECT A BANK OF DUT REGISTERS.
3708 015760 010324 6$: MOV R3,(R4)+ ;WRITE THE CODE TO A DUT REGISTER.
3709 015762 020467 164272 CMP R4,TXBFCA ;COMPARE POINTER WITH LAST REGISTER ADDRESS.
3710 015766 103774 BLO 6$ ;LOOP IF NOT ALL REGS DONE IN THIS BANK.
3711 015770 032701 000017 BIT #17,R1 ;TEST FOR IND.ADR.REG FIELD DECREMENTED TO 0.
3712 015774 001365 BNE 4$ ;LOOP UNTIL ALL REGISTERS CONTAIN THE CODE.
3713
3714 015776 60$: PASS ;RESTORE GPRS.
015776 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3715 016000 000207 RTS PC

```

GLOBAL SUBROUTINE

- TSABRT -

```

3717 .SBTTL GLOBAL SUBROUTINE - TSABRT -
3718 ;** *****
3719 ;* - TEST ABORT ROUTINE -
3720 ;* This subroutine is used when a non-test related error has been found
3721 ;* during the execution of the current test.
3722 ;* It is used to inform the operator that the current test has been
3723 ;* aborted.
3724 ;*
3725 ;* INPUTS: ERRMSG - Contains the name of the current test.
3726 ;* ERRNBR - Contains the correct error number.
3727 ;* The remainder of the ERRBLK is correctly initialised.
3728 ;*
3729 ;* OUTPUTS: Messages are reported to the operator.
3730 ;*
3731 ;* CALLING SEQUENCE: JSR PC,TSABRT
3732 ;*
3733 ;* COMMENTS:
3734 ;*
3735 ;* SUBORDINATE ROUTINES CALLED: ER1603.
3736 ;-- *****
3737
3738 TSABRT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016002 004567 166106 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3739 016006 012701 016024 MOV #2$,R1 ;PASS ADDRESS OF FIRST MESSAGE TO BE REPORTED.
3740 016012 012767 011662 166072 MOV #ER1603,ERRBLK ;SET-UP THE ERROR REPORTING ROUTINE.
3741 016020 104460 ERROR ; >>>> ERROR <<<<.
016020 000432 ; TRAP C$ERROR
3742 016022 040 116 117 2$: BR 60$ ;
3743 016024 116 055 122 .ASCIZ / NON-RELATED TEST ERROR FOUND DURING TEST EXECUTION/
016027 105 114 101
016032 124 105 104
016035 040 124 105
016040 123 124 040
016043 105 122 122
016046 117 122 040
016051 106 117 125
016054 116 104 040
016057 104 125 122
016062 111 116 107
016065 040 124 105
016070 123 124 040
016073 105 130 105
016076 103 125 124
016101 111 117 116
016104 000
016107
3744 .EVEN
3745 016110 60$: PASS ;RESTORE GPRS.
016110 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3746 016112 000207 RTS PC

```

GLOBAL SUBROUTINE

- TXDATP -

```

3748 .SBTTL GLOBAL SUBROUTINE TXDATP -
3749 ;* *****
3750 ;* - TRANSMIT DATA PATTERN -
3751 ;* This subroutine is used in the FIHAVL.TST.
3752 ;* It transmits a specified number of data bytes on the specified line.
3753 ;*
3754 ;* INPUTS: R0 - Contains the number of data bytes to TX.
3755 ;* R1 - Contains line numb on which transmission is to take place.
3756 ;* BUFBAS to BUFMID contains a 256 byte data pattern.
3757 ;*
3758 ;* OUTPUTS: Data is sent out on the specified line.
3759 ;* Carry set = TX successful.
3760 ;*
3761 ;* CALLING SEQUENCE: TXDATP
3762 ;*
3763 ;* COMMENTS:
3764 ;*
3765 ;* SUBORDINATE ROUTINES CALLED: DODMA.
3766 ;*-- *****
3767
3768 016114 TXDATP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016114 004567 165774 ;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3769 016120 010003 MOV R0,R3 ;PASS THE NUMBER OF CHARS TO TX.
3770 016122 012702 002704 MOV #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
3771 016126 004767 175626 JSR PC,DODMA ;TRANSMIT THE DATA PATTERN.
3772 016132 60$: PASS ;RESTORE GPRS.
016132 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3773 016134 000207 RTS PC

```

GLOBAL SUBROUTINE

- TXDSBL -

```

3775 .SBTTL GLOBAL SUBROUTINE - TXDSBL -
3776 ;* *****
3777 ;* - Transmitter Disable -
3778 ;* This subroutine is used to disable transmission on selected lines by,
3779 ;* clearing the associated TX.ENABLE bit on the DUT.
3780 ;*
3781 ;* INPUTS: R5 - Bit's set correspond to lines on which to clear TX.ENABLE.
3782 ;* CSRA - Contains the address of the DUT CSR.
3783 ;* IESTAT - Contains the state of TXIE and RXIE bits in the CSR.
3784 ;* NUMLNS - Equated to be the maximum number of lines available.
3785 ;* TXAD2A - Contains the address of the TBUFFAD2 register.
3786 ;*
3787 ;* OUTPUTS: R5 - Bit's set indicate the initial states of all TX.ENABLE bits.
3788 ;* TBUFFAD2 - The state of the TX.ENABLE bit may be altered.
3789 ;* The contents of the IND.ADD.REG field in the CSR are destroyed.
3790 ;*
3791 ;* CALLING SEQUENCE: JSR PC,TXDSBL
3792 ;*
3793 ;* COMMENTS:
3794 ;*
3795 ;* SUBORDINATE ROUTINES CALLED: NONE.
3796 ;* - *****
3797
3798 016136 TXDSBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016136 004567 165752 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3799 016142 010500 MOV R5,R0 ;COPY BIT MAP OF LINES TO DISABLE TRANSMISSION.
3800 016144 012701 000001 MOV #BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
3801 016150 016702 164102 MOV TXAD2A,R2 ;GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
3802 016154 005202 INC R2 ;GET THE ADDRESS OF THE MSBYTE OF TBUFFAD2 REG.
3803 016156 012703 000010 MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER PLUS ONE.
3804 016162 016704 164102 MOV IESTAT,P4 ;GET THE STATES OF THE INT ENABLE BITS.
3805 016166 005005 CLR R5 ;LOG POSSIBLE TX DISABLED ON ALL LINES.
3806 ;*
3807 ; Select every line in turn, and log the state of each TX.ENABLE bit.
3808 ;*
3809 016170 010477 164046 2$: MOV R4,@CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
3810 016174 105712 TSTB (R2) ;CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
3811 016176 100001 BPL 4$ ;SKIP NEXT INSTRUCTION IF TX.ENABLE CLEAR.
3812 016200 050105 BIS R1,R5 ;LOG TX ENABLE BIT SET FOR SELECTED LINE.
3813 ;*
3814 ; Clear TX.ENABLE on lines that have a corresponding bit set in the tx disable
3815 ; line bit map.
3816 ;*
3817 016202 030100 4$: BIT R1,R0 ;CHECK STATE OF DISABLE LINE BIT MAP.
3818 016204 001402 BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
3819 016206 142712 000200 BICB #BIT7,(R2) ;CLEAR TX.ENABLE BIT ON SELECTED LINE.
3820 016212 005204 6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
3821 016214 006301 ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
3822 016216 005303 DEC R3 ;DECREMENT LINE NUMBER.
3823 016220 001363 BNE 2$ ;LOOP TO CHECK NEXT LINE.
3824 ;*
3825 016222 60$: PASS R5 ;RESTORE GPRS,EXCEPT
016222 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
016226 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3826 ;R5 - PREVIOUS STATES OF ALL TX.ENABLE BITS.
3827 016230 000207 RTS PC

```

GLOBAL SUBROUTINE

- TXENBL -

```

3829 .SBTTL GLOBAL SUBROUTINE - TXENBL -
3830 ;** *****
3831 ;* - Transmitter Enable -
3832 ;* This subroutine is used to enable transmission on selected lines by
3833 ;* setting the associated TX.ENABLE bit on the DUT.
3834 ;*
3835 ;* INPUTS: R5 - Bit's set correspond to lines on which to set TX.ENABLE.
3836 ;* CSRA - Contains the address of the DUT CSR.
3837 ;* IESTAT - Contains the state of TXIE and RXIE bits in the CSR.
3838 ;* NUMLNS - Equated to be the maximum number of lines available.
3839 ;* TXAD2A - Contains the address of the TBUFFAD2 register.
3840 ;*
3841 ;* OUTPUTS: R5 - Bit's set indicate previously disabled lines.
3842 ;* TBUFFAD2 - The state of the TX.ENABLE bit may be altered.
3843 ;* The contents of the IND.ADD.REG field in the CSR are destroyed.
3844 ;*
3845 ;* CALLING SEQUENCE: JSR PC,TXENBL
3846 ;*
3847 ;* COMMENTS:
3848 ;*
3849 ;* SUBORDINATE ROUTINES CALLED: NONE.
3850 ;-- *****
3851
3852 016232 TXENBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
3853 016232 004567 165656 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3854 016236 010500 MOV R5,R0 ;COPY BIT MAP OF LINES TO ENABLE.
3855 016240 012701 000001 MOV #BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
3856 016244 016702 164006 MOV TXAD2A,R2 ;GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
3857 016250 005202 INC R2 ;GET THE ADDRESS OF THE MSBYTE OF TBUFFAD2 REG.
3858 016252 012703 000010 MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER.
3859 016256 016704 164006 MOV IESTAT,R4 ;GET THE STATES OF THE INT ENABLE BITS.
3860 016262 005005 CLR R5 ;CLEAR TX.ENABLE BIT LOG OF DISABLED LINES.
3861 ;*
3862 ; Select every line in turn, and log any TX.ENABLE bit that is clear.
3863 016264 010477 163752 2$: MOV R4,@CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
3864 016270 105712 TSTB (R2) ;CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
3865 016272 100401 BMI 4$ ;SKIP NEXT INSTRUCTION IF TX.ENABLE SET.
3866 016274 050105 BIS R1,R5 ;LOG TX ENABLE BIT CLEAR FOR SELECTED LINE.
3867 ;*
3868 ; Set TX.ENABLE on lines that have a corresponding bit set in the tx enable
3869 ; line bit map.
3870 ;
3871 016276 030100 4$: BIT R1,R0 ;CHECK STATE OF TX.ENABLE LINE BIT MAP.
3872 016300 001402 BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
3873 016302 152712 000200 BISB #BIT7,(R2) ;ENABLE TRANSMISSION ON SELECTED LINE.
3874 016306 005204 6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
3875 016310 006301 ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
3876 016312 005303 DEC R3 ;DECREMENT LINE NUMBER.
3877 016314 001363 BNE 2$ ;LOOP TO CHECK NEXT LINE.
3878 ;*
3879 016316 000014 60$: PASS R5 ;RESTORE GPRS, EXCEPT
3880 016316 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
3881 016322 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3882 016324 000207 RTS PC ;R5 - LINE BIT MAP CORRESPONDING TO THE
; PREVIOUS LINES THAT WERE DISABLED.

```

GLOBAL SUBROUTINE

- TXIEQ -

```

3884 .SBTTL GLOBAL SUBROUTINE - TXIEQ -
3885 ;** *****
3886 ;* - TRANSMITTER INTERRUPT DISABLE -
3887 ;* This routine is used to disable transmitter interrupts in the DHV11.
3888 ;*
3889 ;* INPUTS: NONE.
3890 ;*
3891 ;* OUTPUTS: The TX.INT.ENBL bit is cleared in the DUT CSR.
3892 ;* IESTST -contains the updated status of the TX and RX interrupt
3893 ;* enable bits.
3894 ;*
3895 ;* CALLING SEQUENCE: JSR PC,TXIEQ
3896 ;*
3897 ;* COMMENTS: The contents of the indirect address register field in
3898 ;* the DUT CSR are destroyed.
3899 ;*
3900 ;* SUBORDINATE ROUTINES CALLED: NONE.
3901 ;-- *****
3902 016326 010046 TXIEQ:: MOV RO,-(SP) ;SAVE CONTENTS OF RO ON THE STACK.
3903 016330 GETPRI -(SP) ;SAVE CURRENT PROCESSOR PRIORITY ON THE STACK.
3904 016332 010046 TRAP C$GPRI
016334 012700 000340 SETPRI #PRI07 ;IGNORE ANY INTERRUPTS THAT MAY BE GENERATED.
016340 104441 MOV #PRI07,RO
3905 016342 042767 177677 163720 BIC #177677,IESTAT ;CLEAR TX.INT.ENBL BIT IN IESTAT.
3906 016350 016777 163714 163664 MOV IESTAT,@CSRA ;DISABLE TX INTERRUPTS.
3907 016356 SETPRI (SP)+ ;ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
016356 012600 MOV (SP)+,RO
016360 104441 TRAP C$SPRI
3908 016362 012600
3909 016364 000207 MOV (SP)+,RO ;RESTORE RO.
RTS PC

```

GLOBAL SUBROUTINE

- UNSDIV -

```

3911 .SBTTL GLOBAL SUBROUTINE - UNSDIV -
3912 ;+ *****
3913 ;* - Unsigned Divide Routine -
3914 ;* This subroutine is used to divide a 32 bit unsigned dividend by a
3915 ;* 16 bit unsigned divisor giving a 16 bit quotient. All numbers are
3916 ;* considered to be unsigned. A success flag is not set on return if
3917 ;* the quotient was too big to be contained in 16 bits.
3918 ;*
3919 ;* INPUTS: R1 - The divisor, unsigned, 16 bits.
3920 ;* R2 - Most significant word of the dividend, unsigned, 16 bits.
3921 ;* R3 - Least significant word of the dividend, unsigned, 16 bits.
3922 ;*
3923 ;* OUTPUTS: R1 - Quotient, unsigned, 16 bits (17777 if overflow).
3924 ;* CARRY - Success flag, set if complete quotient fits in 16 bits.
3925 ;*
3926 ;* CALLING SEQUENCE: JSR PC,UNSDIV
3927 ;*
3928 ;* COMMENTS: If the divisor is 0 the quotient is returned as all ones
3929 ;* (17777) and the carry is clear regardless of the dividend.
3930 ;*
3931 ;* SUBORDINATE ROUTINES CALLED: None.
3932 ;-- *****
3933
3934 016366 004567 165522 UNSDIV:: SAVE JSR ;SAVE CONTENTS OF GPRS R0 THRU R5.
016366 ; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3935 ;+
3936 ; Check for quotient greater than 16 bits condition.
3937 ;-
3938 016372 010204 MOV R2,R4 ;GET MSW OF DIVIDEND FOR SUBTRACT.
3939 016374 160104 SUB R1,R4 ;SUBTRACT DIVISOR FROM MSW OF DIVIDEND.
3940 016376 103403 BCS 2$ ;IF IT DIDN'T GO, WE HAVE QUOTIENT < 16 BITS.
3941 016400 012701 177777 MOV #-1,R1 ;SET QUOTIENT TO ALL ONES (177777),
3942 016404 000442 BR 60$ ;EXIT WITH CARRY CLEAR.
3943 ;+
3944 ; Set up counters and various working GPRs.
3945 ;-
3946 016406 005004 2$: CLR R4 ;CLEAR THE LSW OF THE DIVISOR.
3947 016410 000241 CLC ;CLEAR CARRY FOR THE SHIFT OF THE DIVISOR.
3948 016412 006001 ROR R1 ; DIVISOR BY
3949 016414 006004 ROR R4 ; 2(UNSIGNED)
3950 016416 012700 000020 MOV #16.,R0 ;SET UP INITIAL SHIFT COUNT TO 16.
3951 ;+
3952 ; The subtract and shift loop.
3953 ;-
3954 016422 010246 4$: MOV R2,-(SP) ;SAVE MSWORD OF DIVIDEND.
3955 016424 010346 MOV R3,-(SP) ;SAVE LSWORD OF DIVIDEND.
3956 016426 160403 SUB R4,R3 ;LSWORD DIVIDEND - LSWORD OF DIVISOR.
3957 016430 005602 SBC R2 ;MSWORD DIVIDEND - BORROW.
3958 016432 103402 BCS 6$ ;IF BORROW FROM BORROW SUBTRACT, IT DIDN'T GO.
3959 016434 160102 SUB R1,R2 ;MSWORD DIVIDEND - MSWORD OF DIVISOR.
3960 016436 103003 BCC 8$ ;IF NO BORROW, IT WENT, CARRY IS CLEAR.
3961 ;+
3962 ; It didn't go, so we shift a 1 into the quotient (complemented later).
3963 ; Carry is set.
3964 ;-
3965 016440 012603 6$: MOV (SP)+,R3 ;RESTORE LSWORD OF DIVIDEND.
3966 016442 012602 MOV (SP)+,R2 ;RESTORE MSWORD OF DIVIDEND.

```



GLOBAL SUBROUTINE

- UNSDIV -

```

3967 016444 000401          BR      10$           ;GOTO SHIFT 1 INTO THE QUOTIENT.
3968
3969                        ;+
3970                        ; It went, so we restore the stack and shift a 0 into quotient (will be
3971                        ; complemented later).  Carry is clear.
3972 016446 012626      8$:  MOV      (SP)+,(SP)+       ;POP THE SAVED DIVIDEND OFF OF THE STACK.
3973
3974                        ;+
3975                        ; Shift the result of the subtract attempt into the quotient shift reg.
3976 016450 006105      10$:  ROL      R5           ;SHIFT NEXT BIT INTO THE INVERTED QUOTIENT.
3977 016452 000241          CLC                   ;DIVIDE THE
3978 016454 006001          ROR      R1           ; DEVISOR BY
3979 016456 006004          ROR      R4           ; 2 (UNSIGNED).
3980 016460 005300          DEC      R0           ;COUNT THIS SHIFT AND SUBTRACT.
3981 016462 001357          BNE      4$           ;LOOP FOR ANOTHER SHIFT & SUB IF NOT DONE.
3982 016464 005105          COM      R5           ;GET QUOTIENT FROM INVERTED QUOTIENT.
3983
3984                        ;+
3985                        ; Now we either round up or leave quotient alone.
3986 016466 000241          CLC                   ;CLEAR THE CARRY FOR THE SHIFT OF THE DIVIDEND.
3987 016470 006103          ROL      R3           ;MULTIPLY LSWORD OF DIVIDEND BY 2, MSWORD IS 0.
3988 016472 103402          BCS      12$          ;IF CARRY FROM SHIFT, ROUND UP.
3989 016474 160403          SUB      R4,R3       ;SUBTRACT DIVISOR FROM DIVIDEND.
3990 016476 103403          BCS      14$          ;IF BORROW, DON'T ROUND UP.
3991
3992                        ;+
3993                        ; Round up, extra subtract went.
3994 016500 005205      12$:  INC      R5           ;INCREMENT THE QUOTIENT BY ONE.
3995 016502 001001          BNE      14$          ;IF NO OVERFLOW, WE LEAVE THE ROUND UP.
3996 016504 005305          DEC      R5           ;DON'T LET ROUNDING CAUSE OVERFLOW.
3997
3998                        ;+
3999                        ; All done, pass quotient and exit.
4000 016506 010501      14$:  MOV      R5,R1       ;PASS QUOTIENT BACK IN R1.
4001 016510 000261          SEC                   ;INDICATE NO OVERFLOW.
4002
4003 016512 010166 000004  60$:  PASS      R1           ;RESTORE GPRS, LEAVE THE FOLLOWING INTACT:
4004 016516 004736          MOV      R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
4005 016520 000207          JSR      PC,@(SP)+   ;RETURN TO PREGOS SUBRT.
                                ;R1 - 16 BIT, UNSIGNED QUOTIENT,
                                ;CARRY - SET INDICATES NO OVERFLOW (SUCCESS).

```

GLOBAL SUBROUTINE

- WAIBIC -

```

4007 .SBTTL GLOBAL SUBROUTINE - WAIBIC -
4008 ;* *****
4009 ;* - Wait For Bit Clear Routine -
4010 ;* This subroutine waits for the specified bit to become clear. If the
4011 ;* specified bit goes to a clear state within the specified time-out
4012 ;* period a success indication is returned by this routine.
4013 ;* The last value which is read looking for the condition is returned to
4014 ;* allow the use of this routine to look for destructive read conditions.
4015 ;*
4016 ;* INPUTS: R1 - Time-out value and bit number indication:
4017 ;* Bits 15 thru 12 - Number of bit to test (range 0 thru 15).
4018 ;* Bits 11 thru 0 - Time-out value in milli-seconds (4095 max).
4019 ;* R2 - Address of word containing the bit to test.
4020 ;* MSLCNT.
4021 ;*
4022 ;* OUTPUTS: R2 - The last word which was read to check for the condition.
4023 ;* CARRY - Success flag (CARRY set if bit clr before time out).
4024 ;*
4025 ;* CALLING SEQUENCE: MOV #130040,R1 ;PASS BIT 11 (13 OCTAL) AND
4026 ;* ; 32 (40 OCTAL) MS DELAY.
4027 ;* MOV #LABEL,R2 ;TEST BIT IN WORD AT "LABEL".
4028 ;* JSR PC,WAIBIC ;WAIT 32 MS FOR BIT 11 TO CLR.
4029 ;*
4030 ;* COMMENTS:
4031 ;*
4032 ;* SUBORDINATE ROUTINES CALLED: MSLGET.
4033 ;*
4034 ;* *****
4035 016522 WAIBIC:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016522 004567 165366 JSP R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4036 016526 010204 MOV R2,R4 ;SET UP THE ADDRESS PARAMETER FOR MSLGET.
4037 016530 010102 MOV R1,R2
4038 016532 042701 170000 BIC #170000,R1 ;SEPERATE DELAY COUNT OUT OF PASSED PARAMETER.
4039 016536 042702 007777 BIC #7777,R2 ;SEPERATE LINE NUMBER FIELD OF PASSED PARAM.
4040 016542 000302 SWAB R2 ;PUT LINE NUMBER FIELD IN LSBYTE.
4041 016544 006202 ASR R2 ;SHIFT THE LINE NUMBER FIELD INTO THE PROPER
4042 016546 006202 ASR R2 ; POSITION TO USE IT AS A WORD TABLE OFFSET
4043 016550 006202 ASR R2 ; FOR THE TABLE LOOKUP OF THE LINE BIT MAP.
4044 016552 016202 002370 MOV BITTBL(R2),R2 ;GET BIT MAP OF LINE TO TEST FROM TABLE.
4045 016556 005003 CLR R3 ;INDICATE THAT THE BIT SHOULD BE CLR.
4046 016560 004767 175610 JSR PC,MSLGET ;WAIT FOR THE BIT TO BE CLR WITHIN TIME-OUT.
4047 ; CARRY IS CORRECT UPON MSLGET RETURN.
4048 016564 010002 MOV R0,R2 ;PASS LAST VALUE READ AS OUTPUT PARAMETER.
4049 016566 60$: PASS R2 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
016566 010266 000006 MOV R2,R2SLOT(SP) ;PUT R2 IN STACK SLOT.
016572 004736 JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
4050 ; R2 - LAST VALUE READ LOOKING FOR CONDITION.
4051 016574 000207 RTS PC ; CARRY SUCCESS FLAG (SET IF BIT FOUND CLR).

```

GLOBAL SUBROUTINE

- WAIBIS -

```

4053 .SBTTL GLOBAL SUBROUTINE - WAIBIS -
4054 ;* *****
4055 ;* - Wait For Bit Set Routine -
4056 ;* This subroutine waits for the specified bit to become set. If the
4057 ;* specified bit goes to a set state within the specified time-out
4058 ;* period a success indication is returned by this routine.
4059 ;* The last value which is read looking for the condition is returned to
4060 ;* allow the use of this routine to look for destructive read conditions.
4061 ;*
4062 ;* INPUTS: R1 - Time-out value and bit number indication:
4063 ;* Bits 15 thru 12 - Number of bit to test (range 0 thru 15).
4064 ;* Bits 11 thru 0 - Time-out value in milli-seconds (4095 max).
4065 ;* R2 Address of word containing the bit to test.
4066 ;* MSLCNT.
4067 ;*
4068 ;* OUTPUTS: R2 - The last word which was read to check for the condition.
4069 ;* CARRY - Success flag (CARRY set if bit set before time-out).
4070 ;*
4071 ;* CALLING SEQUENCE: MOV #130040,R1 ;PASS BIT 11 (13 OCTAL) AND
4072 ;* ; 32 (40 OCTAL) MS DELAY.
4073 ;* MOV #LABEL,R2 ;TEST BIT IN WORD AT "LABEL".
4074 ;* JSR PC,WAIBIS ;WAIT 32 MS FOR BIT 11 TO SET.
4075 ;*
4076 ;* COMMENTS:
4077 ;*
4078 ;* SUBORDINATE ROUTINES CALLED: MSLGET.
4079 ;* *****
4080
4081 016576 WAIBIS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016576 004567 165312 JSP R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4082 016602 010204 MOV R2,R4 ;SET UP THE ADDRESS PARAMETER FOR MSLGET.
4083 016604 010102 MOV R1,R2
4084 016606 042701 170000 BIC #170000,R1 ;SEPERATE DELAY COUNT OUT OF PASSED PARAMETER.
4085 016612 042702 007777 BIC #7777,R2 ;SEPERATE LINE NUMBER FIELD OF PASSED PARAM.
4086 016616 000302 SWAB R2 ;PUT LINE NUMBER FIELD IN LSBYTE.
4087 016620 006202 ASR R2 ;SHIFT THE LINE NUMBER FIELD INTO THE PROPER
4088 016622 006202 ASR R2 ; POSITION TO USE IT AS A WORD TABLE OFFSET
4089 016624 006202 ASR R2 ; FOR THE TABLE LOOKUP OF THE LINE BIT MAP.
4090 016626 016202 002370 MOV BITTBL(R2),R2 ;GET BIT MAP OF LINE TO TEST FROM TABLE.
4091 016632 010203 MOV R2,R3 ;INDICATE THAT THE BIT SHOULD BE SET.
4092 016634 004767 175534 JSR PC,MSLGET ;WAIT FOR THE BIT TO BE SET WITHIN TIME-OUT.
4093 ; CARRY IS CORRECT UPON MSLGET RETURN.
4094 016640 010002 MOV R0,R2 ;PASS LAST VALUE READ AS OUTPUT PARAMETER.
4095 016642 604: PASS R2 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
016642 010266 000006 MOV R2,R2SLOT(SP) ;PUT R2 IN STACK SLOT.
016646 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4096 ; R2 - LAST VALUE READ LOOKING FOR CONDITION.
4097 016650 000207 RTS PC ; CARRY SUCCESS FLAG (SET IF BIT FOUND SET).

```

GLOBAL SUBROUTINE

- WAITTX -

```

4099 .SBTTL GLOBAL SUBROUTINE - WAITTX -
4100 ;* *****
4101 ;* - WAIT FOR TX TO FINISH -
4102 ;* This subroutine is used in the FIHAVL.TST.
4103 ;* It waits for transmission to complete ie TX_ACTION. Then delays
4104 ;* for 5 milliseconds to allow time for the last character to get into
4105 ;* the fifo.
4106 ;*
4107 ;* INPUTS: CSRA - Contains the address of the CSR.
4108 ;*
4109 ;* OUTPUTS: Carry - Set indicates success.
4110 ;*
4111 ;* CALLING SEQUENCE: JSR PC,WAITTX
4112 ;*
4113 ;* COMMENTS:
4114 ;*
4115 ;* SUBORDINATE ROUTINES CALLED: DELAY,WAIBIS.
4116 ;* - *****
4117
4118 016652 WAITTX:: SAVE ;SAVE CONTENTS OF C'RS R0 THRU R5.
4119 016652 004567 165236 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4120 016656 012701 170536 MOV #170536,R1 ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
4121 016662 016702 163354 MOV CSRA,R2 ;PASS THE ADDRESS OF THE CSR.
4122 016666 004767 177704 JSR PC,WAIBIS ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
4123 016672 103005 BCC 60$ ;BRANCH IF FIFO EMPTY, ABORT THE TEST.
4124 016674 012704 000005 MOV #5,R4 ;PASS DELAY OF 5 MILLI SECS.
4125 016700 004767 175014 JSR PC,DELAY ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
4126 016704 000261 SEC ;SET CARRY TO I8NDICATE SUCCESS.
4127 016706 60$: PASS ;RESTORE GPRS.
4128 016706 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4129 016710 000207 RTS PC ;PASS THE CARRY BIT, SET INDICATES SUCCESS.

```

GLOBAL SUBROUTINE

- WTWLNC

```

4131 .SBTTL GLOBAL SUBROUTINE - WTWLNC -
4132 ;* *****
4133 ;* - Line Control Register Setup Routine -
4134 ;* This subroutine is used to set the Device Under Test (DUT) Line
4135 ;* Control Registers (LNCTRL) to the specified state. Only the LNCTRLS
4136 ;* for the specified lines are altered.
4137 ;*
4138 ;* INPUTS: R0 - New line parameters.
4139 ;* R5 - Bit map of lines to be altered.
4140 ;* CSRA - Contains address of the DUT CSR.
4141 ;* IESTAT - Contains the current state of the TX and RX interrupt
4142 ;* enable bits in the CSR.
4143 ;* LNCTRA Contains address of the DUT LNCTRL registers.
4144 ;*
4145 ;* OUTPUTS: LNCTRL - Specified DUT Line Control Registers are altered.
4146 ;*
4147 ;* CALLING SEQUENCE: JSR PC,WTWLNC
4148 ;*
4149 ;* COMMENTS:
4150 ;*
4151 ;* SUBORDINATE ROUTINES CALLED: ALTFLD.
4152 ;* *****
4153
4154 016712 WTWLNC:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016712 004567 165176 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4155 ;*
4156 ; Set up the parameters for the call to ALTFLD.
4157 ;-
4158 016716 016701 163330 MOV LNCTRA,R1 ;SET UP THE REGISTER ADDRESS PARAMETER.
4159 016722 010002 MOV R0,R2 ;SET UP THE DESIRED REGISTER CONTENTS.
4160 016724 010503 MOV R5,R3 ;SET UP THE BIT MAP OF LINES TO ALTER.
4161 016726 012704 177777 MOV #-1,R4 ;SELECT ALL REGISTER BITS TO BE ALTERED.
4162 ;*
4163 ; Call the subroutine which alters the register contents.
4164 ;-
4165 016732 004767 174042 JSR PC,ALTFLD ;ALTER THE REGISTER CONTENTS.
4166
4167 016736 60$: PASS ;RESTORE GPRS.
016736 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4168 016740 000207 RTS PC

```

GLOBAL SUBROUTINE

- WTWLPR -

4170  
4171  
4172  
4173  
4174  
4175  
4176  
4177  
4178  
4179  
4180  
4181  
4182  
4183  
4184  
4185  
4186  
4187  
4188  
4189  
4190  
4191  
4192  
4193 016742  
016742 004567 165146  
4194  
4195  
4196  
4197 016746 016701 163274  
4198 016752 010002  
4199 016754 010503  
4200 016756 012704 177777  
4201  
4202  
4203  
4204 016762 004767 174012  
4205  
4206 016766  
016766 004736  
4207 016770 000207

```

.SBTTL GLOBAL SUBROUTINE - WTWLPR -
;* *****
;* - Line Parameter Register Setup Routine -
;* This subroutine is used to set the Device Under Test (DUT) Line
;* Parameter Registers (LPR) to the specified state. Only the LPRs for
;* the specified lines are altered.
;*
;* INPUTS: R0 - New line parameters.
;* R5 - Bit map of lines to be altered.
;* CSRA - Contains address of the DUT CSR.
;* IESTAT - Contains the current state of the TX and RX interrupt
;* enable bits in the CSR.
;* LPRA - Contains address of the DUT LPR.
;*
;* OUTPUTS: LPR - Specified DUT Line Parameter Registers are altered.
;*
;* CALLING SEQUENCE: JSR PC,WTWLPR
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: ALTFLD.
;* *****
WTWLPR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;*
;* Set up the parameters for the call to ALTFLD.
;*
;*
MOV LPRA,R1 ;SET UP THE REGISTER ADDRESS PARAMETER.
MOV R0,R2 ;SET UP THE DESIRED REGISTER CONTENTS.
MOV R5,R3 ;SET UP THE BIT MAP OF LINES TO ALTER.
MOV # 1,R4 ;SELECT ALL REGISTER BITS TO BE ALTERED.
;*
;* Call the subroutine which alters the register contents.
;*
JSR PC,ALTFLD ;ALTER THE REGISTER CONTENTS.
60$: PASS ;RESTORE GPRS.
RTS PC JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

```

INTERRUPT SERVICE ROUTINE

- CLKINT -

```

4209      .SBTTL  INTERRUPT SERVICE ROUTINE      - CLKINT -
4210      ;++ *****
4211      ;*      This routine is executed CLKHRZ times per second.  It decrements the
4212      ;*      two timer counters down to zero.
4213      ;*
4214      ;* INPUTS:      TIMER1 - Timer counter #1.
4215      ;*              TIMER2 - Timer counter #2.
4216      ;*              TIMER3 - Timer counter for call of BREAK macro.
4217      ;*
4218      ;* OUTPUTS:     The 2 timer counters are decremented if they are not zero.
4219      ;*
4220      ;* CALLING SEQUENCE:  Put #CLKINT in the clock interrupt vector slot.
4221      ;*                  Put the desired time period (seconds times CLKHRZ) in
4222      ;*                  either TIMER1 or TIMER2 and poll the respective timer
4223      ;*                  counter to detect its going to 0 on time-out.
4224      ;*
4225      ;* COMMENTS:     The 2 counters will not wraparound but will stop at 0.  This
4226      ;*                  allows the detection of a time-out any time after the time-out
4227      ;*                  has occurred until the timer counter is set to another value.
4228      ;*
4229      ;* SUBORDINATE ROUTINES CALLED:  None.
4230      ;-- *****
4231
4232 016772 005767 163330  CLKINT:: TST    TIMER1      ;CHECK FOR TIMER1 AT ZERO.
4233 016776 001402          BEQ    2$          ;BRANCH TO LEAVE IT AT ZERO IF IT IS ZERO.
4234 017000 005367 163322          DEC    TIMER1      ;DECREMENT TIME COUNT.
4235 017004 005767 163320  2$:   TST    TIMER2      ;CHECK FOR TIMER2 AT ZERO.
4236 017010 001402          BEQ    4$          ;BRANCH TO LEAVE IT ALONE IF IT'S ALREADY ZERO.
4237 017012 005367 163312          DEC    TIMER2      ;DECREMENT TIME COUNT.
4238 017016 005367 163310  4$:   DEC    TIMER3      ;DECREMENT THE BREAK COUNT.
4239 017022 001006          BNE    60$          ;EXIT IF NOT TIME TO CALL BREAK.
4240 017024 016767 163304 163300  MOV    BCOUNT,TIMER3  ;SET UP TIME TILL NEXT BREAK.
4241 017032 010046          MOV    R0,-(SP)    ;SAVE CONTENTS OF R0 FROM BREAK MACRO.
4242 017034          BREAK          ;CHECK FOR OPERATOR CONTROL/C.
4243 017036 012600          MOV    (SP)+,R0    ;RESTORE CONTENTS OF R0.      TRAP    C$BRK
4244 017040 000002          RTI

```

GLOBAL TRAP SERVICE ROUTINE - TP4RTN -

```

4246 .SBTTL GLOBAL TRAP SERVICE ROUTINE - TP4RTN -
4247 ;*****
4248 ;* Bus Time-out Trap (004 trap) Service Routine -
4249 ;* This routine is used during the Device Register Address Access Test.
4250 ;* It determines if the 004 trap was caused by an "expected" error or
4251 ;* not by examining the return PC value on the stack. If the trap is
4252 ;* unexpected, this routine jumps to the normal Diagnostic Supervisor
4253 ;* 004 trap handling routine.
4254 ;*
4255 ;* INPUTS: SP - Points to the PC where the trap occurred.
4256 ;* ADRPTR Label at the address where "expected" traps occur.
4257 ;* TP4FLG - 004 trap flags.
4258 ;*
4259 ;* OUTPUTS: TP4FLG - Bit 15 is set if "expected" trap occurred.
4260 ;*
4261 ;* CALLING SEQUENCE: Put address pointed to by TP4RTN in 004 vector.
4262 ;* Occurrence of 004 trap vectors to this routine.
4263 ;*
4264 ;* COMMENTS: Any 004 trap which occurs at an address other than that labeled
4265 ;* ADRPTR will be handled by the normal 004 trap service routine.
4266 ;*
4267 ;* SUBORDINATE ROUTINES CALLED: None.
4268 ;*****
4269
4270 017042 021627 013470 TP4RTN:: CMP (SP),#ADRPTR ;COMPARE EXPECTED ADR AGAINST TRAP RET PC.
4271 017046 001402 BEQ 2$ ;IF THEY MATCH, CONTINUE THIS ROUTINE.
4272 017050 000177 163234 JMP @TP4VEC ;IF NOT, JUMP TO NORMAL 004 TRAP SERVICE RTN.
4273 017054 052767 100000 163230 2$: BIS #BIT15,TP4FLG ;SET THE 004 TRAP OCCURED FLAG.
4274 017062 000002 RTI ;ALL DONE, GO BACK TO THE TEST.

```



GLOBAL TRAP SERVICE ROUTINE - TP4RTN -

```

4276
4277 ;*****
** 4278 ;
4279 ;           VDHB.RPT
4280 ;
4281 ;*****
4282
4283
4284
4285 .SBTTL REPORT CODING SECTION
4286
4287 ;**
4288 ; THE REPORT CODING SECTION CONTAINS THE
4289 ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
4290 ;--
4291
4292 017064 BGNRPT
4293 017064 L$RPT::
4294 017064 EXIT RPT
4295 017064 000167 .WORD J$JMP
4296 017066 000000 .WORD L10014-2-.
4297
4298 017070 ENDRPT
4299 017070 L10014:
4300 017070 104425 TRAP C$RPT

```

PROTECTION TABLE

```

4300      .SBTTL  PROTECTION TABLE
4301
4302      ;*****
*
4303      ;
4304      ;           SKL4.P11
4305      ;
4306      ;*****
4307
4308
4309
4310      ;++
4311      ; THIS TABLE IS USED BY THE RUNTIME SERVICES
4312      ; TO PROTECT THE LOAD MEDIA.
4313      ;--
4314
4315 017072      BGNPROT
         017072
4316
4317 017072 177777      -1      ;OFFSET INTO P-TABLE FOR CSR ADDRESS
4318 017074 177777      -1      ;OFFSET INTO P TABLE FOR MASSBUS ADDRESS
4319 017076 177777      -1      ;OFFSET INTO P-TABLE FOR DRIVE NUMBER
4320
4321 017100      ENDPROT
4322

```

L\$PROT::

PROTECTION TABLE

```

4337
4338 ;*****
4339 ;
4340 ;           VDHB.INI
4341 ;
4342 ;*****
4343
4344
4345
4346 .SBTTL INITIALIZE SECTION
4347 ;++
4348 ;*****
4349 ;*   This section contains the code which is performed at the beginning of
4350 ;*   each pass or after a continue command.
4351 ;*   This code performs the following actions:
4352 ;*
4353 ;*   Moves the information held in the hardware P-table into the global
4354 ;*   data area.
4355 ;*
4356 ;*****
4357 ;--
4358 017100      BGNINIT
4359 017100
4360 017100      ;SEE IF PROGRAM JUST STARTED, BR IF YES
4360 017100 012700 000040      READEF #EF.START
4361 017104 104447      MOV #EF.START,RO
4361 017106      BCOMPLETE NEWSTA      TRAP C$REFG
4362 017106 103416      BCS NEWSTA
4363 017110      ;SEE IF PROGRAM JUST RESTARTED, BR IF YES
4363 017110 012700 000037      READEF #EF.RESTART
4364 017114 104447      MOV #EF.RESTART,RO
4364 017116      BCOMPLETE NEWRES      TRAP C$REFG
4365 017116 103556      BCS NEWRES
4366 017120      ;SEE IF THIS IS A NEW PASS, BR IF YES
4366 017120 012700 000035      READEF #EF.NEW
4367 017124 104447      MOV #EF.NEW,RO
4367 017126      BCOMPLETE NEWPAS      TRAP C$REFG
4368 017126 103555      BCS NEWPAS
4369 017130      ;SEE IF PROGRAM WAS JUST CONTINUED
4369 017130 012700 000036      READEF #EF.CONTINUE
4370 017134 104447      MOV #EF.CONTINUE,RO
4370 017136 103161      BNCOMPLETE GETPRM      TRAP C$REFG
4371 017140 000167 000556      JMP ENDIT      BCC GETPRM
4372 017144      NEWSTA:
4373 017144 104433      BRESET ;RESET THE BUS TO PREVENT ILLEGAL INTERRUPTS.
4374 ;*
4375 ; Set up for Line Time Clock interrupts.
4376 ;--
4377 017146      CLOCK L,R1 ;GET THE CLOCK PARAMETERS.
4377 017146 012700 000114      MOV #'L,RO
4377 017152 104462      TRAP C$CLCK

```

INITIALIZE SECTION

```

017154 010001
4378 017156 012167 163134      MOV      (R1)+,CLKCSR      ;STORE CLOCK CSR ADDRESS.
4379 017162 012167 163132      MOV      (R1)+,CLKBRL     ;STORE CLOCK BUS REQ INT LEVEL.
4380 017166 012167 163130      MOV      (R1)+,CLKVEC     ;STORE CLOCK INTERRUPT VECTOR.
4381 017172 012167 163126      MOV      (R1)+,CLKHRZ     ;STORE CLOCK FREQUENCY.
4382 017176 026727 163122 000062  CMP      CLKHRZ,#50.      ;TEST FOR 50HZ LINE FREQUENCY.
4383 017204 001004              BNE      2$              ;BRANCH IF CLOCK IS NOT 50HZ.
4384 017206 012767 000024 163122  MOV      #20.,MSTICK      ;INDICATE 20MS PER CLOCK TICK.
4385 017214 000403              BR       4$
4386 017216 012767 000021 163112 2$:  MOV      #17.,MSTICK      ;INDICATE 17 MS PER CLOCK TICK.
4387 017224 4$:  SETVEC  CLKVEC,#CLKINT,#PRIO6 ;INITIALIZE CLOCK INTERRUPT VECTOR.
                                MOV      #PRIO6,-(SP)
                                MOV      #CLKINT,-(SP)
                                MOV      CLKVEC,-(SP)
                                MOV      #3,-(SP)
                                TRAP     C$SVEC
                                ADD      #10,SP
4388 017252 016700 163046      MOV      CLKHRZ,R0       ;INITIALIZE THE BREAK COUNT
4389 017256 006200              ASR      R0              ; TO CAUSE A BREAK
4390 017260 010067 163050      MOV      R0,BCOUNT      ; EVERY 1/2 SECOND.
4391 017264              SETPRI  #PRIO5          ;ALLOW CLOCK INTERRUPTS DISABLE OTHERS.
                                MOV      #PRIO5,R0
                                TRAP     C$SPRI
4392
4393 ;+
4394 ; Enable the Line Time Clock (LTC) checking to make sure that the CSR
4395 ; is accessable.
4396 ; First set up to catch any 004 traps which occur:
4397 017272 016767 160506 163010  MOV      4,TP4VEC        ;SAVE THE EXISTING 004 TRAP VECTOR.
4398 017300 012767 017042 160476  MOV      #TP4RTN,4      ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
4399
4400 ;+
4401 ; Enable LTC checking for 004 trap in case CSR is not there.
4402 017306 005067 163000      CLR      TP4FLG         ;CLEAR THE 004 TRAP FLAG.
4403 017312 012767 000100 162774  MOV      #BIT6,WORD1     ;SET UP TO SET BIT6 OF THE LTC CSR.
4404 017320 012700 002314      MOV      #WORD1,R0      ;SET UP WORD1 AS THE CKTRAP MOVE SOURCE.
4405 017324 016701 162766      MOV      CLKCSR,R1      ;SET UP LTC CSR AS DESTINATION FOR CKTRAP MOVE.
4406 017330 004767 174122      JSR     PC,CKTRAP       ;MOVE AND CHECK FOR TRAP.
4407 017334 016767 162750 160442  MOV      TP4VEC,4       ;RESTORE THE NORMAL 004 TRAP VECTOR.
4408 017342 103403              BCS     6$              ;IF NO TRAP, LTC IS THERE SO CONTINUE.
4409 017344 005067 162754      CLR      CLKHRZ         ;CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.
4410 017350 000402              BR      8$              ;BYPASS THE FOLLOWING CALIBRATION PROCEDURES.
4411
4412 ;+
4413 ; Calibrate the DELAY routine milli-second delay count value.
4414 017352 004767 173604      6$:  JSR     PC,CALMSL
4415
4416 ;+
4417 ; Check for Memory Management present on this machine.
4418 ; If MEM MGT is present, disable it.
4419 017356 016767 160422 162724 8$:  MOV      4,TP4VEC        ;SAVE THE EXISTING 004 TRAP VECTOR.
4420 017364 012767 017042 160412  MOV      #TP4RTN,4      ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
4421 017372 005067 162714      CLR      TP4FLG         ;CLEAR THE 004 TRAP FLAG.
4422 017376 005067 162712      CLR      WORD1          ;PREPARE TO CLEAR THE MEM MGT SRO REGISTER.
4423 017402 012700 002314      MOV      #WORD1,R0      ;SELECT CLEARED WORD AS CKTRAP RTN SOURCE.
4424 017406 016701 162730      MOV      MMSRO,R1      ;SELECT MEM MGT SRO REGISTER AS DESTINATION.
4425 017412 005067 162726      CLR      MMPRES         ;INDICATE NO MEM MGT PRESENT IN CASE IT ISN'T.

```

INITIALIZE SECTION

```

4426 017416 005067 162724          CLR    MMENAB          ;INDICATE MEM MGT IS NOT ENABLED.
4427 017422 004767 174030          JSR    PC,CKTRAP      ;CLEAR THE MEM MGT SRO REG AND CHECK FOR TRAP.
4428 017426 016767 162656 160350    MOV    TP4VEC,4       ;RESTORE THE NORMAL 004 TRAP VECTOR.
4429 017434 103003                   BCC    10$            ;SKIP INDICATING MEM MGT PRESENT IF IT ISN'T.
4430 017436 012767 000001 162700    MOV    #1,MMPRES     ;INDICATE THAT MEM MGT IS PRESENT.
4431 017444 005067 162626 10$:   CLR    PASCNT         ;CLR COUNTER USED IN REPORTING ROM VERSION #.
4432 017450 000167 000006          JMP    NEWPAS        ;SKIP AROUND THE BUS RESET, IT'S BEEN DONE.
4433
4434 017454          NEWRES: BRESET      ;RESET THE BUS TO PREVENT ILLEGAL INTERRUPTS.
      017454 104433          TRAP    C$RESET
4435 017456 005067 162614          CLR    PASCNT        ;CLR COUNTER USED IN REPORTING ROM VERSION #.
4436 017462
4437 017462 012767 177777 162550    MOV    #-1,UNITN     ;RESET LOGICAL DEVICE TO -1
4438
4439          ;+
4440          ; Increment the pass counter, correct for any overflow.
4441          ; This counter is used in the Rom version test.
4442          ;-
4442 017470 005267 162602          INC    PASCNT        ;INCREMENT THE PASS COUNTER.
4443 017474 001002          BNE    GETPRM       ;BRANCH IF WE HAVE NOT YET! OVERFLOWED.
4444 017476 005367 162574          DEC    PASCNT        ;SET PASS COUNT TO 177777 OCTAL.
4445
4446          ; GET THE HARDWARE PARAMETERS FOR THIS UNIT.
4447 017502          GETPRM:
4448 017502 005267 162532          INC    UNITN         ;INCREMENT LOGICAL DEVICE NUMBER
4449 017506 026767 162526 162276    CMP    UNITN,L$UNIT  ;SEE IF MAXIMUM UNIT NO. EXCEEDED
4450 017514 002362          BGE    NEWPAS        ;BR IF YES
4451
4452 017516          GPHARD UNITN,R1    ;GET P-TABLE POINTER INTO R1
      017516 016700 162516          MOV    UNITN,RO
      017522 104442          TRAP   C$GPHRD
      017524 010001          MOV    RO,R1
4453 017526          BCOMPLETE 30$      ;BR IF DEVICE AVAILABLE
      017526 103401          BCS   30$
4454 017530 000764          BR    GETPRM        ;SKIP THIS DEVICE
4455
4456
4457          ;***** HARDWARE PARAMETER MOVING CODE *****
4458 017532 012167 162504 30$:   MOV    (R1)+,CSRA    ;STORE DHV11-M CSR ADDRESS IN DEV.REG.ADDRESS TABLE
4459 017536 012102          MOV    (R1)+,R2     ;GET THE RX INTERRUPT VECTOR ADDRESS.
4460 017540 010267 162464          MOV    R2,RXVECA    ;STORE RX INT VECTOR ADDRESS.
4461 017544 062702 000004          ADD    #4,R2        ;CALCULATE TX INTERRUPT VECTOR ADDRESS.
4462 017550 010267 162456          MOV    R2,TXVECA    ;STORE TX INT VECTOR ADDRESS.
4463 017554 012167 162454          MOV    (R1)+,ACTLNS ;STORE DHV11-M ACTIVE LINE BIT MAP
4464 017560 012702 000377          MOV    #MAPLNS,R2   ;GET THE BIT MAP FOR ALL LINES.
4465 017564 005102          COM    R2           ;GET A BIT MAP OF NON-EXISTANT LINES.
4466 017566 040267 162442          BIC    R2,ACTLNS    ;CLEAR NON-EXISTANT LINES FROM ACTLNS.
4467 017572 112167 162440          MOV    (R1)+,LOPBCK ;STORE DHV11-M LOOPBACK MODE
4468 017576 112167 162435          MOV    (R1)+,BRLEVL ;STORE DHV11-M INTERUPT BUS REQUEST LEVEL
4469
4470          ;+
4471          ;
4472          ;-
4473 017602 016701 162434          MOV    CSRA,R1      ;COPY CSR ADDRESS
4474 017606 005201          INC    R1           ;INCREMENT CSR ADDRESS
4475 017610 005201          INC    R1           ; COPY BY 2.
4476 017612 012703 000007          MOV    #7,R3        ;SET UP REGISTER COUNT
4477 017616 012702 002244          MOV    #RBUFA,R2    ;GET LOCATION WHERE RBUF ADDRESS GOES IN TABLE

```

INITIALIZE SECTION

```

4478 017622 010122      12$:   MOV    R1,(R2)+    ;STORE REGISTER ADDRESS IN TABLE
4479 017624 005201      INC    R1           ;INCREMENT REGISTER ADDRESS
4480 017626 005201      INC    R1           ; BY 2, FOR THE NEXT DEVICE REGISTER.
4481 017630 005303      DEC    R3           ;DECREMENT REGISTER COUNT
4482 017632 001373      BNE   12$          ;LOOP IF NOT DONE
4483
4484
4485      ;+
4485      ; Initialise the BMP code queue.
4486      ;-
4487 017634 012700 002444      MOV    #BMPCQB,R0    ;GET THE START ADDRESS OF THE QUEUE.
4488 017640 012701 002644      MOV    #BMPCQE,R1    ;GET THE END ADDRESS OF THE QUEUE.
4489 017644 010067 162572      MOV    R0,BMPCQP     ;SET THE POINTER TO THE START OF THE QUEUE.
4490 017650 005020      14$:   CLR    (R0)+         ;CLEAR OUT THE CONTENTS OF THE QUEUE.
4491 017652 020001      CMP    R0,R1         ;CHECK IF END OF QUEUE HAS BEEN REACHED.
4492 017654 103775      BLO   14$           ;LOOP IF NOT ALL DONE.
4493
4494      ;+
4494      ; Report the Unit number if the software P-table question was answered YES,
4495      ; and the maximum unit number is greater than 1.
4496      ;-
4497 017656 032767 000020 162340      BIT    #BIT4,OPTION  ;CHECK IF THE QUESTION WAS ANSWERED YES.
4498 017664 001416      BEQ   16$           ;SKIP REPORTING UNIT NUMBER IF IT IS DISABLED.
4499 017666 026727 162120 000001      CMP    L$UNIT,#1     ;CHECK MAXIMUM NUMBER OF UNITS SELECTED.
4500 017674 003412      BLE   16$           ;DO NOT REPORT UNIT NUMBER IF MAX NUMBER < 1.
4501 017676      PRINTF #MFUNIT,UNITN ;REPORT UNIT NUMBER.
      MOV    UNITN,-(SP)
      MOV    #MFUNIT,-(SP)
      MOV    #2,-(SP)
      MOV    SP,R0
      TRAP   C$PNTF
      ADD    #6,SP
4502 017722      16$:
4503
4504 017722 005067 162336      ENDIT: CLR   CTRLCF    ;CLR THE CTRL-C TEST ABORT FLAG.
4505
4506      ;+
4506      ; Set the processor priority to allow LTC interrupts but not others.
4507      ;-
4508 017726      SETPRI #PRI05      ;SET PROCESSOR PRIORITY TO 5.
      MOV    #PRI05,R0
      TRAP   C$SPRI
4509
4510      ;+
4510      ; Enable Line Time Clock if one is available.
4511      ;-
4512 017734 005767 162364      TST   CLKHRZ        ;CHECK FOR A LTC BEING PRESENT.
4513 017740 001403      BEQ   18$           ;LTC PRESENT? NO, SKIP LTC ENABLE.
4514 017742 012777 000100 162346      MOV    #BIT6,@CLKCSR ;YES, ENABLE THE LTC.
4515 017750      18$:
4516
4517 017750      ENDINIT
      L10016:
      TRAP   C$INIT
4518
4519      TNUM == 0      ;INITIALIZE THE ASSEMBLER TEST NUMBER VARIABLE.

```

INITIALIZE SECTION

```

4522 ;*****
4523 ;
4524 ;           VDHB.ATD
4525 ;
4526 ;*****
4528
4529
4530 .SBTTL  AUTODROP SECTION
4531
4532
4533 ;**
4534 ; THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
4535 ; THE "ADR" FLAG WAS SET.  THE UNIT(S) UNDER TEST ARE CHECKED TO
4536 ; SEE IF THEY WILL RESPOND.  THOSE THAT DON'T ARE IMMEDIATELY
4537 ; DROPPED FROM TESTING.
4538 ;--
4539
4540 017752          BGNAUTO
4541 017752
4542
4543
4544
4545
4546
4547
4548
4549 017752          ENDAUTO
4550 017752
4551 017752 104461
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596
4597
4598
4599

```

```

L$AUTO::
L10017: TRAP C$AUTO

```

AUTODROP SECTION

```

4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567 017754
      017754
4568
4569 017754 005767 162304
4570 017760 001401
4571 017762
      017762 104433
4572 017764
      017764 012700 000240
      017770 104441
4573 017772 005767 162326
4574 017776 001402
4575 020000 005077 162312
4576 020004
4577
4586
4587 020004
      020004 104432
      020006 000002
4588
4600
4601
4602
4603 020010
      020010
      020010 104412

```

```

*****
VDHB.CUC
*****

.SBTTL CLEANUP CODING SECTION

;+
; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
;

      BGNCLN
                                L$CLEAN::

      TST   CTRLCF                ;DID WE GET HERE BY CTRL-C FROM TEST?
      BEQ   2$                     ;CTRL-C FROM TEST? NO, SKIP BUS RESET.
      BRESET                                ;YES, CLR ANY DMAS OR OUTSTANDING INTERRUPTS.
                                TRAP   C$RESET
      2$:  SETPRI #PRI05            ;ALLOW LTC INTERRUPTS, DISALLOW OTHERS.
                                MOV    #PRI05,R0
                                TRAP   C$SPRI

      TST   CLKHRZ                ;SEE IF THE CLOCK IS ON
      BEQ   3$                     ;IF NOT GO ON
      CLR   @CLKCSR               ;STOP THE CLOCK
      3$:

      EXIT   CLN

                                TRAP   C$EXIT
                                .WORD  L10020-.

      .EVEN

      ENDCLN

                                L10020:
                                TRAP   C$CLEAN

```



CLEANUP CODING SECTION

```

4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621 020012
      020012
4622
4631 020012
      020012 010046
      020014 012746 020040
      020020 012746 000002
      020024 010600
      020026 104417
      020030 062706 000006
4632
4633 020034
      020034 000167
      020036 000056
4634
4635 020040 045 101 040 DROP: .ASCIZ/*A UNIT*06*A DROPPED FROM FURTHER TESTING.*N/
      020043 125 116 111
      020046 124 045 104
      020051 066 045 101
      020054 040 104 122
      020057 117 120 120
      020062 105 104 040
      020065 106 122 117
      020070 115 040 106
      020073 125 122 124
      020076 110 105 122
      020101 040 124 105
      020104 123 124 111
      020107 116 107 056
      020112 045 116 000
4636
4637
4638 020116
      020116
      020116 104453
      .EVEN
      ENDDU
      L10021: TRAP C$DU

```

DROP UNIT SECTION

```

4640
4641      ;*****
4642      ;
4643      ;           VDHB.ADD
4644      ;
4645      ;*****
4646
4647
4648
4649      .SBTTL  ADD UNIT SECTION
4650
4651      ;**
4652      ; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
4653      ; TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
4654      ; TO THE TEST CYCLE.
4655      ;--
4656
4657 020120      BGNAU
4658      020120
4659
4660
4661
4662
4663      020120      EXIT  AU
4664      020120      000167
4665      020122      000000
4666
4667
4668
4669
4670
4671      .EVEN
4672
4673 020124      ENDAU
4674      020124
4675      020124      104452
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699

```

```

L$AU:
.WORD  J$JMP
.WORD  L10022-2-.
L10022:
TRAP  C$AU

```

HARDWARE TEST - ADRA -

```

4676 .SBTTL HARDWARE TEST - ADRA -
4677 ;++
4678 ;*****
4679 ;* - REGISTER ADDRESS TEST -
4680 ;*
4681 ;* This test verifies that the Q-bus can read and write to the DHV11
4682 ;* device registers. If the DHV11 does not respond to the access
4683 ;* attempts (If the DHV11 is at the wrong address, for example) the
4684 ;* 004 bus time-out trap 's detected by this routine and an error
4685 ;* is reported.
4686 ;*
4687 ;*****
4688 ;--
4689
4690 020126 BGNTST
4691 020126
4691 000001 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
4692 020126 012767 000001 162132 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (1)
4693 020134 012767 177777 162122 MOV # 1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
4694
4695 ;+
4696 ; Set up to catch any 004 traps which occur:
4697 020142 016767 157636 162140 MOV 4,TP4VEC ;SAVE THE EXISTING 004 TRAP VECTOR.
4698 020150 012767 017042 157626 MOV #TP4RTN,4 ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
4699 020156 005005 CLR R5 ;CLEAR THE ERROR FLAGS.
4700
4701 ;+
4702 ; Set up for the initial iteration of the test loop:
4703 ;-
4704 020160 005004 CLR R4 ;CLEAR THE LINE COUNTER.
4705
4706 ;+
4707 ; Here begins the loop to test the registers for a line.
4708 ; First test the CSR and set the IND.ADR.REG (I.A.R) field.
4709 ;-
4710 020162 005067 162124 2$: CLR TP4FLG ;CLEAR THE 004 TRAP FLAG.
4711 020166 016700 162050 MOV CSRA,R0 ;SET UP CSR AS THE CKTRAP MOVE SOURCE.
4712 020172 012701 020406 MOV #52$,R1 ;SET UP DESTINATION LOCATION FOR CKTRAP MOVE.
4713 020176 004767 173254 JSR PC,CKTRAP ;MOVE AND CHECK FOR TRAP.
4714 020202 103402 BCS 4$ ;IF NO TRAP, BYPASS ERROR.
4715 020204 052705 100001 BIS #100001,R5 ;SET FATAL READ ERROR FLAGS.
4716 020210 042767 000017 000170 4$: BIC #17,52$ ;CLEAR THE I.A.R FIELD OF THE CSR DATA.
4717 020216 050467 000164 BIS R4,52$ ;OR IN THE LINE COUNTER TO THE I.A.R FIELD.
4718 020222 010100 MOV R1,R0 ;USE OLD DESTINATION FOR SOURCE OF CKTRAP MOVE.
4719 020224 016701 162012 MOV CSRA,R1 ;SET UP CSR AS THE CKTRAP MOVE DESTINATION.
4720 020230 004767 173222 JSR PC,CKTRAP ;MOVE AND CHECK FOR TRAP.
4721 020234 103403 BCS 6$ ;IF NO TRAP, BYPASS ERROR.
4722 020236 052705 100002 BIS #100002,R5 ;SET FATAL WRITE ERROR FLAGS.
4723 020242 000440 BR 40$ ;EXIT AND REPORT FATAL ERROR.
4724
4725 ;+
4726 ; Now, we test each register for this line.
4727 020244 012702 000010 6$: MOV #10,R2 ;INIT REGISTER COUNTER TO 8.
4728 020250 016767 161766 000126 MOV CSRA,50$ ;INITIALIZE THE REGISTER POINTER.
4729 020256 012700 020404 8$: MOV #50$,R0 ;SET UP REGISTER AS THE SOURCE FOR CKTRAP MOVE.
4730 020262 012701 020406 MOV #52$,R1 ;SET UP LOCAL STORAGE AS THE DES FOR CKTRAP.
4731 020266 004767 173164 JSR PC,CKTRAP ;PERFORM THE MOVE, CHECK FOR TRAP.

```

## HARDWARE TEST

- ADRA

```

4732 020272 103402          BCS 10$          ;IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS.
4733 020274 052705 100001  BIS #100001,R5    ;SET FATAL READ ERROR FLAGS.
4734 020300 010100          MOV R1,R0        ;USE OLD DEST AS SRC FOR CKTRAP MOVE.
4735 020302 012701 020404  MOV #50$,R1      ;SET UP REGISTER AS THE DEST FOR CKTRAP MOVE.
4736 020306 004767 173144  JSR PC,CKTRAP    ;PERFORM THE MOVE, CHECK FOR TRAP.
4737 020312 103402          BCS 12$          ;IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS.
4738 020314 052705 100002  BIS #100002,R5    ;SET FATAL WRITE ERROR FLAGS.
4739 020320 005267 000060  12$: INC 50$      ;INCREMENT THE REGISTER
4740 020324 005267 000054  INC 50$          ; POINTER BY 2.
4741 020330 005302          DEC R2            ;COUNT THE REGISTER.
4742 020332 001351          BNE 8$           ;LOOP TO TEST THE NEXT REGISTER ADDRESS.
4743
4744
4745          ;+
4746          ; Now we set up to test the next line, or to exit if we are done.
4747 020334 005204          INC R4            ;INCREMENT THE LINE COUNTER.
4748 020336 020427 000010  CMP R4,#NUMLNS   ;COMPARE LINE COUNTER AGAINST NUMBER OF LINES.
4749 020342 002707          BLT 2$             ;LOOP TO TEST THE NEXT LINE IF WE'RE NOT DONE.
4750
4751          ;+
4752          ; Done checking device register addresses.
4753          ; Report any errors and exit.
4754          ;-
4755 020344 016767 161740 157432 40$: MOV TP4VEC,4     ;RESTORE THE NORMAL 004 TRAP VECTOR.
4756 020352 005705          TST R5             ;CHECK THE ERROR FLAGS.
4757 020354 100015          BPL 60$          ;EXIT ROUTINE IF NO ERRORS.
4758          ; REPORT "DEVICE REGISTER ACCESS ERRORS"
4759 020356          ERRDF 101,EM0103,ER0101; >>>> ERROR #101 <<<<<.
          020356 104455          TRAP C$ERDF
          020360 000145          .WORD 101
          020362 005473          .WORD EM0103
          020364 011320          .WORD ER0101
4760
4761 020366          DODU UNITN          ;DROP THIS UNIT FROM FUTHER TESTING.
          020366 016700 161646          MOV UNITN,R0
          020372 104451          TRAP C$DODU
4762 020374 005067 161664          CLR CTRLCF      ;INDICATE NO CTRL C ABORT FROM TEST.
4763 020400          DOCLN             ;ABORT THIS SUB PASS.
          020400 104444          TRAP C$DCLN
4764 020402 000402          BR 60$
4765
4766          ;+
4767          ; Local storage.
4768 020404 000000          50$: .WORD 0          ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
4769 020406 000000          52$: .WORD 0          ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
4770 020410 005067 161650          60$: CLR CTRLCF    ;INDICATE THAT WE ARE NOT WITHIN A TEST.
4771 020414          ENDTST
          020414          L10023:
          020414 104401          TRAP C$ETST

```

HARDWARE TEST

- DMASTA -

```

4773 .SBTTL HARDWARE TEST          DMASTA -
4774 ;** *****
4775 ;*                               - DMA Start Bit Test -
4776 ;*   This test verifies that the DMA_START bit in the DUT's Line control
4777 ;*   registers will initiate DMA transmission on the selected line.
4778 ;*   This test is performed in internal loopback, on all active lines.
4779 ;*
4780 ;-- *****
4781 020416          BGNTST
4782                T2::
4783 020416 000002    TNUM == TNUM + 1    ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
4784 020424 012767 000002 161642    MOV #TNUM,TSTNUM    ;SET UP THE TEST NUMBER. (40)
4785 020432 012767 177777 161632    MOV #-1,CTRLCF     ;INDICATE THAT WE ARE IN A TEST.
4786 020440 012767 000001 163444    MOV #1,ERRTYP      ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
4787 020446 012767 007641 163440    MOV #4001,ERRNBR   ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
4788 020454 012767 005774 163434    MOV #EM4001,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRTABL.
4789                MOV #ER9101,ERRBLK   ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
4790 ;*
4791 ; Reset the DUT to a known state, remove the status codes from the fifo.
4792 ; Clear TX and RX interrupt enable bits in the CSR.
4793 ; This subroutine reports error >>>> 4001 <<<<<.
4794 020462 004767 173020    JSR PC,CLNRST      ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
4795 020466 103141          BCC 50$                ;RESET FAILURE?, ABORT THIS TEST.
4796
4797 020470 004767 173520    JSR PC,INDATP      ;INITIALSE THE 256 BYTE DATA PATTERN.
4798 ;*
4799 ; Set internal loopback,enable receiver functions on all active lines.
4800 ; Set LPR on all lines to 38.4k baud, 8 bits per character, odd parity,
4801 ; 2 stop bits.
4802 ; Enable transmitters on all active lines.
4803 ;--
4804 020474 016705 161534    MOV ACTLNS,R5      ;PASS THE ACTIVE LINE BIT MAP.
4805 020500 012700 000204    MOV #204,R0        ;PASS THE LNCTRL CONTENTS.
4806 020504 004767 176202    JSR PC,WTWLNC      ;INITIALISE THE LNCTRL REGISTERS.
4807 020510 012700 177670    MOV #177670,R0     ;PASS THE LPR CONTENTS.
4808 020514 004767 176222    JSR PC,WTWLPR      ;INITIALSE THE LPR REGISTERS ON ALL LINES.
4809 020520 004767 175506    JSR PC,TXENBL      ;ENABLE TRANSMITTERS ON ALL LINES.
4810 ;*
4811 ; Set up outer loop to test the DMA START bit on all active lines.
4812 ;
4813 020524 016705 161504    MOV ACTLNS,R5      ;GET THE ACTIVE LINE BIT MAP.
4814 020530 005001          CLR R1              ;CLEAR THE LINE NUMBER COUNTER.
4815 020532 012767 007642 163346 2$: MOV #4002.,ERRNBR   ;SET THE ERROR NUMBER TO 4002.
4816 020540 000241          CLC                ;CLEAR THE CARRY BIT PRIOR TO SHIFTING BIT MAP.
4817 020542 006005          ROR R5                    ;SHIFT THE BIT MAP INTO THE CARRY BIT.
4818 020544 103106          BCC 14$                ;DO NOT TEST THE LINE IF IT IS INACTIVE.
4819 020546 004767 174262    JSR PC,PUFIFO      ;PURGE THE FIFO.
4820 020552 103107          BCC 50$                ;GO REPORT ERROR IF FIFO WILL NOT CLEAR.
4821 ;*
4822 ; Perform DMA_START bit testing on each line individually.
4823 ; Test each DMA_START bit before TX'ing data pattern, report error if set.
4824 ; Set DMA_START bit on LUT, verify it is set, report error if clear.
4825 ; Wait for DMA to complete.
4826 ; Verify DMA_START bit is clear, report error if set.
4827 ; Verify correct number of chars were received, report error if < expected.
4828 ;--

```

## HARDWARE TEST

- DMASTA -

```

4829 020554 005267 163326      INC  ERRNBR      ;SET ERROR NUMBER TO 4003.
4830 020560 012702 002704      MOV  #BUFBAS,R2  ;PASS THE START OF THE DATA PATTERN TO TX.
4831 020564 012703 000144      MOV  #100.,R3    ;PASS THE LENGTH OF THE DATA PATTERN.
4832 020570 004767 173164      JSR  PC,DODMA    ;TRANSMIT THE DATA PATTERN.
4833 020574 103067              BCC  12$         ;GO REPORT ERROR IF DMA_START BIT SET.
4834
4835      ;+
4836      ; Test the state of the DMA_START bit on the line under test.
4837      ; Report error if DMA_START bit is clear.
4838 020576 005267 163304      INC  ERRNBR      ;INCREMENT ERROR NUMBER TO 4004.
4839 020602 010177 161434      MOV  R1,@CSRA    ;SELECT THE LINE CURRENTLY UNDER TEST.
4840 020606 105777 161444      TSTB @TXAD2A    ;TEST THE STATE OF THE DMA_START BIT.
4841 020612 100060              BPL  12$         ;GO REPORT ERROR IF BIT IS CLEAR.
4842
4843      ;+
4844      ; Wait for DMA transmission to complete.
4845 020614 005267 163266      4$: INC  ERRNBR      ;INCREMENT ERROR NUMBER TO 4005.
4846 020620 010103              MOV  R1,R3       ;SAVE THE LINE NUMBER.
4847 020622 012701 170226      MOV  #170226,R1  ;TEST BIT 15, TIMEOUT OF 150 MILLI SECS.
4848 020626 016702 161410      MOV  CSRA,R2     ;PASS THE ADDRESS OF THE REGISTER TO TEST.
4849 020632 004767 175740      JSR  PC,WAIBIS   ;WAIT FOR DMA TO COMPLETE.
4850 020636 103045              BCC  10$         ;GO REPORT ERROR IF TIMEOUT OCCURRED.
4851 020640 012704 000005      MOV  #5,R4       ;PASS DELAY OF 5 MILLI SECS.
4852 020644 004767 173050      JSR  PC,DELAY    ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
4853 020650 010301              MOV  R3,R1       ;RESTORE THE CURRENT LINE NUMBER.
4854
4855      ;+
4856      ; Test the state of the DMA_START bit on the line under test.
4857      ; Report error if DMA_START bit is set.
4858 020652 005267 163230      INC  ERRNBR      ;INCREMENT ERROR NUMBER TO 4006.
4859 020656 010177 161360      MOV  R1,@CSRA    ;SELECT THE LINE CURRENTLY UNDER TEST.
4860 020662 105777 161370      TSTB @TXAD2A    ;TEST THE STATE OF THE DMA_START BIT.
4861 020666 100432              BMI  12$         ;GO REPORT ERROR IF BIT IS STILL SET.
4862
4863      ;+
4864      ; Verify the number of chars received = number of chars expected.
4865      ; Report error if count is incorrect.
4866      ; If more than 128 BMP codes are found then report error and exit test.
4867 020670 005003              CLR  R3          ;CLEAR THE READ COUNTER.
4868 020672 012704 000200      MOV  #128.,R4    ;SET UP MAX BMP CODE READ COUNT.
4869 020676 012767 007647 163202 6$: MOV  #4007.,ERRNBR ;SET ERROR NUMBER TO 4007.
4870 020704 017702 161334      MOV  @RBUFA,R2   ;READ THE CHARACTER FROM THE FIFO.
4871 020710 100021              BPL  12$         ;GO REPORT ERROR IF FIFO EMPTY TOO SOON.
4872 020712 012700 170301      MOV  #170301,R0  ;SET-UP BIT MASK OF A BMP CODE.
4873 020716 040200              BIC  R2,R0       ;TRY TO CLEAR THE BMP CODE MASK.
4874 020720 001007              BNE  8$         ;BRANCH IF NOT A BMP CODE.
4875 020722 005267 103160      INC  ERRNBR      ;INCREMENT ERROR NUMBER TO 4008.
4876 020726 004767 174612      JSR  PC,SAVBMP   ;SAVE THE BMP CODE ON THE QUEUE.
4877 020732 005304              DEC  R4          ;DECREMENT MAX BMP CODE READ COUNT.
4878 020734 001416              BEQ  50$         ;GO REPORT ERROR IF TOO MANY BMP CODES FOUND.
4879 020736 000757              BR   6$         ;DO NOT COUNT THE BMP CODE AS A VALID CHAR.
4880 020740 005203 000144      8$: INC  R3        ;COUNT THIS CHARACTER.
4881 020742 020327              CMP  R3,#100.    ;HAVE WE RECIEVED 100 CHARACTERS?.
4882 020746 002753              BLT  6$         ;LOOP UNTIL 100 (NON BMP) CHARS ARE READ.
4883 020750 000404              BR   14$        ;SKIP AROUND THE ERROR REPORT.
4884
4885      ;+

```

HARDWARE TEST

- DMASTA -

```

4886 ; Report error, skip further testing on this line.
4887 ;
4888 020752 010301 10$: MOV R3,R1 ;RESTORE THE CURRENT LINE NUMBER.
4889 ;
4890 020754 012702 006017 12$: MOV #EM4002,R2 ;PASS THE ERROR MESSAGE TO BE REPORTED.
4891 ; "DMA_START BIT BAD ON LINE nn".
4892 020760 ERROR ; >>>> ERROR <<<<<.
4893 020760 104460 ; TRAP C$ERROR
4894 020762 005201 14$: INC R1 ;INCREMENT THE LINE NUMBER COUNTER.
4895 020764 005705 TST R5 ;ARE THERE ANY MORE ACTIVE LINES TO TEST?.
4896 020766 001261 BNE 2$ ;YES; BRANCH TO TEST THE NEXT LINE.
4897 020770 000402 BR 60$ ;NO; EXIT THIS TEST.
4898 ;
4899 020772 004767 175004 50$: JSR PC,TSABRT ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
4900 020776 005067 161262 60$: CLR CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.
4901 ;
4902 021002 ENDTST
021002 L10024:
021002 104401 TRAP C$ETST
    
```

HARDWARE TEST

- DMABRT -

```

4904 .SBTTL HARDWARE TEST - DMABRT -
4905 ;** *****
4906 ;* - DMA Abort/Restart Test -
4907 ;* This test verifies that each DMA_ABORT bit will correctly halt
4908 ;* a DMA transmission, and return a TX_ACTION.
4909 ;* It will also verify that the aborted DMA transmission can be resumed,
4910 ;* and that a TX_ACTION is returned upon completion.
4911 ;* This test is performed in internal loopback, on all active lines.
4912 ;*
4913 ;-- *****
4914 021004 BGNTST
      021004
4915 000003 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER
4916 021004 012767 000003 161254 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (41)
4917 021012 012767 177777 161244 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
4918 021020 012767 000001 163056 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
4919 021026 012767 010005 163052 MOV #4101,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
4920 021034 012767 006053 163046 MOV #EM4101,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRTABL.
4921 021042 012767 012572 163042 MOV #ER9101,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
4922
4923 ;+
4924 ; Reset the DUT to a known state, remove the status codes from the fifo.
4925 ; Clear TX and RX interrupt enable bits in the CSR.
4926 ; This subroutine reports error >>>> 4101 <<<<.
4927 021050 004767 172432 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
4928 021054 103160 BCC 60$ ;RESET FAILURE?, ABORT THIS TEST.
4929
4930 021056 004767 173132 JSR PC,INDATP ;INITIALISE 256 BYTE DATA PATTERN.
4931
4932 ;+
4933 ; Set internal loopback,enable receiver functions on all active lines.
4934 ; Set LPR on all lines to 38.4k baud, 8 bits per character, odd parity,
4935 ; 2 stop bits.
4936 ; Enable transmitters on all active lines.
4937 021062 016705 161146 MOV ACTLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
4938 021066 012700 000204 MOV #204,R0 ;PASS THE LNCTRL CONTENTS.
4939 021072 004767 175614 JSR PC,WTWLNLC ;INITIALISE THE LNCTRL REGISTERS.
4940 021076 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
4941 021102 004767 175634 JSR PC,WTWLPR ;INITIALISE THE LPR REGISTERS ON ALL LINES.
4942 021106 004767 175120 JSR PC,TXENBL ;ENABLE TRANSMITTERS ON ALL LINES.
4943
4944 ;+
4945 ; Perform DMA ABORT bit testing on each individual (active) line.
4946 021112 016705 161116 MOV ACTLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
4947 021116 005001 CLR R1 ;CLEAR THE LINE NUMBER COUNTER.
4948 021120 012767 010006 162760 2$: MOV #4102.,ERRNBR ;SET THE ERROR NUMBER TO 4102.
4949 021126 000241 CLC ;CLEAR THE CARRY BIT PRIOR TO SHIFTING BIT MAP.
4950 021130 006005 ROR R5 ;SHIFT THE BIT MAP INTO THE CARRY BIT.
4951 021132 103123 BCC 10$ ;DO NOT TEST THE LINE IF IT IS INACTIVE.
4952 021134 004767 173674 JSR PC,PUFIFO ;PURGE THE FIFO.
4953 021140 103124 BCC 50$ ;GO REPORT ERROR IF FIFO WILL NOT CLEAR.
4954
4955 ;+
4956 ; Check the DMA_ABORT bit before enabling DMA, report error if set.
4957 021142 005267 162740 INC ERRNBR ;INCREMENT ERROR NUMBER TO 4103.
4958 021146 010177 161070 MOV R1,@CSRA ;SELECT THE LINE CURRENTLY UNDER TEST.
4959 021152 032777 000001 161072 BIT #BIT0,@LNCTRA ;TEST THE STATE OF THE DMA ABORT BIT.

```



## HARDWARE TEST

- DMABRT -

```

4960 021160 001105      BNE      6$          ;GO REPORT ERROR IF BIT IS SET.
4961
4962      ;+
4963      ; Enable DMA TX on selected line, wait for DMA to TX approx 1/4 of data.
4964      ; Abort the DMA transmission. Wait for TX_ACTION to be returned.
4965 021162 005267 162720      INC      ERRNBR      ;SET ERROR NUMBER TO 4104.
4966 021166 012702 002704      MOV      #BUFBAS,R2  ;PASS THE START OF THE DATA PATTERN TO TX.
4967 021172 012703 000400      MOV      #256.,R3   ;PASS THE LENGTH OF THE DATA PATTERN.
4968 021176 004767 172556      JSR      PC,DODMA   ;TRANSMIT THE DATA PATTERN.
4969 021202 103103      BCC      50$        ;GO REPORT ERROR IF THERE ARE TX PROBLEMS.
4970
4971      ;+
4972      ; Wait for DMA to transmit 1/4 of the data before aborting.
4973 021204 010177 161032      MOV      R1,@CSRA   ;SELECT THE LINE CURRENTLY UNDER TEST.
4974 021210 012704 000062      MOV      #50.,R4   ;PASS THE DELAY TIME OF 40 MILLI SECONDS.
4975 021214 004767 172500      JSR      PC,DELAY   ;WAIT FOR APPROX 1/4 OF DATA TO BE TX'D.
4976 021220 052777 000001 161024      BIS      #BIT0,@LNCTRA ;ABORT THE DMA TRANSMISSION.
4977
4978      ;+
4979      ; Wait for TX_ACTION to be returned, report error if time-out occurs.
4980 021226 005267 162654      INC      ERRNBR      ;INCREMENT ERROR NUMBER TO 4105.
4981 021232 010103      MOV      R1,R3     ;SAVE THE LINE NUMBER.
4982 021234 012701 170012      MOV      #170012,R1 ;TEST BIT 15, TIMEOUT OF 10 MILLI SECS.
4983 021240 016702 160776      MOV      CSRA,R2   ;PASS THE ADDRESS OF THE REGISTER TO TEST.
4984 021244 004767 175326      JSR      PC,WAIBIS  ;WAIT FOR DMA TO COMPLETE.
4985 021250 103050      BCC      4$        ;GO REPORT ERROR IF TIMEOUT OCCURRED.
4986 021252 010301      MOV      R3,R1     ;RESTORE THE CURRENT LINE NUMBER.
4987
4988      ;+
4989      ; Verify DMA_START bit clear, report error if set.
4990 021254 005267 162626      INC      ERRNBR      ;INCREMENT ERROR NUMBER TO 4106.
4991 021260 012702 006132      MOV      #EM4103,R2 ;SELECT MESSAGE TO BE REPORTED.
4992
4993      ; "DMA_START BIT FOUND SET AFTER DMA ABORTED".
4994 021264 010177 160752      MOV      R1,@CSRA   ;SELECT THE LINE CURRENTLY UNDER TEST.
4995 021270 105777 160762      TSTB    @TXAD2A    ;TEST THE STATE OF THE DMA_START BIT.
4996 021274 100441      BMI      8$        ;GO REPORT ERROR IF IT IS SET.
4997
4998      ;+
4999      ; Resume DMA transmission by clearing DMA_ABORT and setting DMA START.
5000 021276 042777 000001 160746      BIC      #BIT0,@LNCTRA ;CLEAR THE DMA_ABORT BIT.
5001 021304 052777 000200 160744      BIS      #BIT7,@TXAD2A ;SET THE DMA_START BIT.
5002
5003      ;+
5004      ; Wait for DMA transmission to complete.
5005 021312 005267 162570      INC      ERRNBR      ;INCREMENT ERROR NUMBER TO 4107.
5006 021316 010103      MOV      R1,R3     ;SAVE THE LINE NUMBER.
5007 021320 012701 170536      MOV      #170536,R1 ;TEST BIT 15, TIMEGUT OF 350 MILLI SECS.
5008 021324 016702 160712      MOV      CSRA,R2   ;PASS THE ADDRESS OF THE REGISTER TO TEST.
5009 021330 004767 175242      JSR      PC,WAIBIS  ;WAIT FOR DMA TO COMPLETE.
5010 021334 103016      BCC      4$        ;GO REPORT ERROR IF TIMEOUT OCCURRED.
5011 021336 012704 000002      MOV      #2,R4     ;PASS TIME-OUT OF 2 MILLI SECS.
5012 021342 004767 172352      JSR      PC,DELAY   ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
5013 021346 010301      MOV      R3,R1     ;RESTORE THE CURRENT LINE NUMBER.
5014
5015      ;+
5016      ; Test the state of the DMA_ABORT bit on the line under test.
      ; Report error if DMA_ABORT bit is set.

```

## HARDWARE TEST

- DMABRT -

```

5017 021350 005267 162532          INC   ERRNBR      ;INCREMENT ERROR NUMBER TO 4108.
5018 021354 010177 160662          MOV   R1,@CSRA   ;SELECT THE LINE CURRENTLY UNDER TEST.
5019 021360 032777 000001 160664  BIT   #BIT0,@LNCTRA ;TEST THE STATE OF THE DMA_ABORT BIT.
5020 021366 001002                BNE   6$         ;GO REPORT ERROR IF BIT IS SET.
5021 021370 000404                BR    10$        ;BRANCH TO CHECK FOR ANY MORE LINES TO TEST.
5022
5023          ;+
          ; Report error, skip further testing on this line.
5024          ;-
5025 021372 010301          4$:   MOV   R3,R1      ;RESTORE THE CURRENT LINE NUMBER.
5026
5027 021374 012702 006076          6$:   MOV   #EM4102,R2 ;PASS THE ERROR MESSAGE TO BE REPORTED.
5028          ; "DMA_ABORT BIT BAD ON LINE nn".
5029 021400          8$:   ERROR          ;
          >>>> ERROR <<<<<.
          TRAP   C$ERROR
5030
5031          ;+
          ; Verify all active lines have been tested.
5032          ;-
5033 021402 005201          10$:  INC   R1         ;INCREMENT THE LINE NUMBER COUNTER.
5034 021404 005705          TST   R5         ;ARE THERE ANY MORE ACTIVE LINES TO TEST?.
5035 021406 001244          BNE   2$         ;YES; BRANCH TO TEST THE NEXT LINE.
5036 021410 000402          BR    60$        ;NO; EXIT THIS TEST.
5037
5038 021412 004767 174364          50$:  JSR   PC,TSABRT ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
5039 021416 005067 160642          60$:  CLR   CTRLCF   ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5040
5041 021422                ENDTST
          L10025:
          TRAP   C$ETST
021422
021422 104401

```

HARDWARE TEST

- OAUTOI -

```

5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054 021424
      021424
5055 021424 126727 160606 000002
5056 021432 001402
5057 021434 000167 000516
5058      000004
5059 021440 012767 000004 160620
5060 021446 012767 177777 160610
5061 021454 012767 000001 162422
5062 021462 012767 011445 162416
5063 021470 012767 006216 162412
5064 021476 012767 012572 162406
5065
5066
5067
5068
5069
5070 021504 004767 171776
5071 021510 103402
5072 021512 000167 000440
5073
5074
5075
5076 021516 004767 171330
5077
5078
5079
5080
5081
5082
5083 021522 016705 160506
5084 021526 012700 000004
5085 021532 004767 175154
5086 021536 012705 000377
5087 021542 012700 177670
5088 021546 004767 175170
5089 021552 004767 174454
5090
5091
5092
5093 021556 012703 100000
5094 021562 016705 160446
5095 021566 046705 160502
5096 021572 010567 000352
5097 021576 005067 000344
5098 021602 016701 000340

```

```

.SBTTL HARDWARE TEST - OAUTOI -
;*****
;*
;* - OAUTO BIT INACTIVE TEST -
;*
;* This test verifies that the DUT's OAUTO function behaves correctly
;* when inactive, ie OAUTO bit clear.
;* This test will only execute if staggered loopback mode is selected.
;* The special staggered loopback connector must be fitted.
;*
;*****
BGNTST
T4::
CMPB LOPBCK,#2 ;CHECK MODE SELECTED.
BEQ .+6 ;DO NOT EXIT IF STAGGERD LOPBCK MODE SELECTED.
JMP 60$ ;EXIT THIS TEST.
TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (49)
MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
MOV #4901,ERRNBR ;SET ERROR NUMBER TO 4901.
MOV #EM4901,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
MOV #ER9101,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
;+
; Reset the DUT to a known state, remove the status codes from the fifo.
; Clear TX and RX interrupt enable bits in the CSR.
; This subroutine reports error >>>> 4901 <<<<<.
;-
JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
BCS .+6 ;DO NOT EXIT IF RESET WAS SUCCESSFUL.
JMP 60$ ;EXIT THIS TEST.
;+
; Set-up the associated TX/RX line number tables.
;-
JSR PC,ASLNTL ;INITIALISE THE ASSOCIATED TX/RX TABLES.
;+
; Set external loopback, disable OAUTO and enable receiver on all active lines.
; Set LPR on all lines to 38.4k baud, 8 bits per character, odd parity,
; 2 stop bits.
; Enable transmitters on all lines.
;-
MOV ACTLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
MOV #4,R0 ;PASS THE LNCTRL CONTENTS.
JSR PC,WTWLNC ;INITIALISE THE LNCTRL REGISTERS.
MOV #MAPLNS,R5 ;PASS BIT MAP OF ALL LINES.
MOV #177670,R0 ;PASS THE LPR CONTENTS.
JSR PC,WTWLPR ;INITIALSE THE LPR REGISTERS ON ALL LINES.
JSR PC,TXENBL ;ENABLE TRANSMITTERS ON ALL LINES.
;+
; Set up outer loop for testing active lines in both line groups.
;-
MOV #100000,R3 ;SET-UP LOOP CONTROL FLAG.
MOV ACTLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
BIC LGRP2M,R5 ;REMOVE LINES IN GROUP 2.
2$: MOV R5,45$ ;SAVE THE CURRENT LINE GROUP.
CLR 40$ ;CLEAR THE LINE NUMBER COUNTER.
4$: MOV 40$,R1 ;COPY THE LINE NUMBER.

```

## HARDWARE TEST

- OAUTOI -

```

5099 021606 000241          CLC          ;CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
5100 021610 006005          ROR          R5          ;SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.
5101 021612 103054          BCC          8$          ;SKIP TESTING THIS LINE IF IT IS INACTIVE.
5102
5103          ;+
5104          ; Test the state of the OAUTO bit on the line,      ~ test.
5105          ; Report error if it is found set, and skip      ~r testing of that line.
5106 021614 012767 011446 162264          MOV          #4902.,ERRNBR ;SET THE ERROR NUMBER TO 4902.
5107 021622 010177 160414          MOV          R1,@CSRA      ;SELECT THE LINE TO BE TESTED.
5108 021626 032777 000020 160416          BIT          #BIT4,@LNCTRA ;TEST THE STATE OF THE OAUTO BIT.
5109 021634 001404          BEQ          6$          ;SKIP ERROR REPORT IF OAUTO BIT IS CLEAR.
5110 021636 012702 006250          MOV          #EM4902,R2    ;PASS THE ERROR MESSAGE.
5111
5112 021642          ERROR          ; "OAUTO BIT BAD ON LINE nn"
5113 021644 000437          BR          8$          ;>>>> ERROR #4902 <<<<<.
5114          TRAP          C$ERROR
5115          ;+
5116          ; Transmit the XOFF (ASCII DC3) on the associated line.
5117 021646 116177 004044 160366 6$:      MOV          TXRLNB(R1),@CSRA ;SELECT THE ASSOCIATED TX LINE.
5118 021654 012777 100023 160362          MOV          #100023,@TXCHA ;TRANSMIT THE XOFF CHARACTER TO THE LUT.
5119
5120          ;+
5121          ; Wait for transmission to complete.
5122 021662 005267 162220          INC          ERRNBR      ;INCREMENT ERROR NUMBER TO 4903.
5123 021666 012701 170012          MOV          #170012,R1   ;TEST BIT 15, TIMEOUT OF 10 MILLI SECS.
5124 021672 016702 160344          MOV          CSRA,R2      ;PASS THE ADDRESS OF THE REGISTER TO TEST.
5125 021676 004767 174674          JSR          PC,WAIBIS    ;WAIT FOR DMA TO COMPLETE.
5126 021702 103123          BCC          50$          ;ABORT TEST IF TIMEOUT OCCURRED.
5127 021704 012704 000005          MOV          #5,R4        ;PASS TIME-OUT OF 5 MILLI SECS.
5128 021710 004767 172004          JSR          PC,DELAY     ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
5129
5130          ;+
5131          ; Test the state of the TX_ENABLE bit on the line under test.
5132          ; Report error if TX_ENABLE bit is clear.
5133 021714 005267 162166          INC          ERRNBR      ;INCREMENT ERROR NUMBER TO 4904.
5134 021720 016701 000222          MOV          40$,R1       ;GET THE NUMBER OF THE LINE TEST.
5135 021724 010177 160312          MOV          R1,@CSRA     ;SELECT THE LINE CURRENTLY UNDER TEST.
5136 021730 005777 160322          TST          @TXAD2A     ;TEST THE STATE OF THE TX_ENABLE BIT.
5137 021734 100403          BMI          8$          ;SKIP ERROR REPORT IF BIT IS SET.
5138 021736 012702 006250          MOV          #EM4902,R2   ;PASS THE MESSAGE TO BE REPORTED.
5139
5140 021742          ERROR          ; "OAUTO BIT BAD ON LINE nn".
5141 021742 104460          ;>>>> ERROR #4904 <<<<<.
5142          TRAP          C$ERROR
5143 021744 005267 000176 8$:      INC          40$          ;INCREMENT THE LINE NUMBER.
5144 021750 005705          TST          R5          ;CHECK IF THERE ARE ANY MORE LINES TO TEST.
5145 021752 001313          BNE          4$          ;
5146
5147          ;+
5148          ; Disable transmitters on the selected lines in the current line group.
5148 021754 016705 000170          MOV          45$,R5       ;RESTORE THE CURRENT LINE ACTIVE LINE GROUP.
5149 021760 004767 174152          JSR          PC,TXDSBL    ;DISABLE TRANSMITTERS ON THE SELECTED LINES.
5150 021764 016705 000160          MOV          45$,R5       ;GET THE CURRENT ACTIVE LINE GROUP AGAIN.
5151 021770 005067 000152          CLR          40$          ;CLEAR THE LINE COUNTER.
5152 021774 012767 011451 162104 10$:      MOV          #4905.,ERRNBR ;SET ERROR NUMBER TO 4905.
5153 022002 016701 000140          MOV          40$,R1       ;COPY THE LINE NUMBER.

```

HARDWARE TEST

- OAUTOI -

```

5154 022006 000241          CLC          ;CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
5155 022010 006005          ROR          R5          ;SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.
5156 022012 103035          BCC          12$        ;SKIP TESTING THIS LINE IF IT IS INACTIVE.
5157
5158          ;*
5159          ; Transmit the XON (ASCII DC1) on the associated line.
5160 022014 116177 004044 160220  MOVB TXRLNB(R1),@CSRA ;SELECT THE ASSOCIATED TX LINE.
5161 022022 012777 100021 160214  MOV #100021,@TXCHA ;TRANSMIT THE XON CHARACTER TO THE LUT.
5162
5163          ;*
5164          ; Wait for transmission to complete.
5165 022030 012701 170012          MOV #170012,R1          ;TEST BIT 15, TIMEOUT OF 10 MILLI SECS.
5166 022034 016702 160202          MOV CSRA,R2           ;PASS THE ADDRESS OF THE REGISTER TO TEST.
5167 022040 004767 174532          JSR PC,WAIBIS         ;WAIT FOR DMA TO COMPLETE.
5168 022044 103042          BCC 50$              ;ABORT TEST IF TIMEOUT OCCURRED.
5169 022046 012704 000005          MOV #5,R4            ;PASS TIME-OUT OF 5 MILLI SECS.
5170 022052 004767 171642          JSR PC,DELAY          ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
5171
5172          ;*
5173          ; Test the state of the TX_ENABLE bit on the line under test.
5174          ; Report error if TX_ENABLE bit is set.
5175 022056 005267 162024          INC ERRNBR           ;INCREMENT ERROR NUMBER TO 4906.
5176 022062 016701 000060          MOV 40$,R1           ;GET THE NUMBER OF THE LINE UNDER TEST.
5177 022066 010177 160150          MOV R1,@CSRA         ;SELECT THE LINE CURRENTLY UNDER TEST.
5178 022072 005777 160160          TST @TXAD2A         ;TEST THE STATE OF THE TX_ENABLE BIT.
5179 022076 100003          BPL 12$              ;SKIP ERROR REPORT IF BIT IS CLEAR.
5180 022100 012702 006250          MOV #EM4902,R2       ;PASS THE MESSAGE TO BE REPORTED.
5181
5182 022104          ERROR          ; "OAUTO BIT BAD ON LINE nn".
5183 022104 104460          ; >>>>> ERROR #4906 <<<<<.
5184          ; TRAP C$ERROR
5184 022106 005267 000034 12$: INC 40$          ;INCREMENT THE LINE NUMBER,
5185 022112 005705          TST R5              ;CHECK IF THERE ARE ANY MORE LINES TO TEST.
5186 022114 001327          BNE 10$              ;
5187
5188          ;*
5189          ; Check loop control flag to determine if both sets of lines have been tested
5190          ; If this is the first time around, re enable TX on all lines, generate active
5191          ; bit map for second line group.
5192 022116 005703          ;*
5193 022120 001416          TST R3              ;HAVE BOTH LINE GROUPS BEEN TESTED?.
5194 022122 005003          BEQ 60$             ;YES; THEN EXIT THIS TEST.
5195 022124 012705 000377          CLR R3              ;NO; CLEAR THE LOOP CONTROL FLAG,
5196 022130 004767 174076          MOV #MAPLNS,R5      ;PASS THE BIT MAP OF ALL AVAILABLE LINE.
5197 022134 016705 160074          JSR PC,TXENBL       ;RE ENABLE TRANSMISSION ON ALL LINES.
5198 022140 046705 160126          MOV ACTLNS,R5       ;GET THE ACTIVE LINE BIT MAP.
5199 022144 000612          BIC LGRP1M,R5       ;REMOVE ALL ACTIVE LINES IN GROUP 1.
5200          BR 2$          ;ONCE MORE AROUND AND WE ARE DONE.
5201 022146 000000 40$: .WORD 0          ;STORAGE FOR CURRENT LINE NUMBER.
5202 022150 000000 45$: .WORD 0          ;STORAGE FOR CURRENT ACTIVE LINE BIT MAP.
5203 022152 004767 173624 50$: JSR PC,TSABRT         ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
5204 022156 005067 160102 60$: CLR CTRLCF         ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5205
5206 022162          ENDTST
022162
022162 104401          L10026: TRAP C$ETST

```

HARDWARE TEST - OAUTOI -

```

5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220 022164
      022164
5221 022164 126727 160046 000002
5222 022172 001402
5223 022174 000167 000516
5224      000005
5225 022200 012767 000005 160060
5226 022206 012767 177777 160050
5227 022214 012767 000001 161662
5228 022222 012767 011611 161656
5229 022230 012767 006302 161652
5230 022236 012767 012572 161646
5231
5232
5233
5234
5235
5236 022244 004767 171236
5237 022250 103402
5238 022252 000167 000440
5239
5240
5241
5242 022256 004767 170570
5243
5244
5245
5246
5247
5248
5249 022262 016705 157746
5250 022266 012700 000024
5251 022272 004767 174414
5252 022276 012705 000377
5253 022302 012700 177670
5254 022306 004767 174430
5255 022312 004767 173714
5256
5257
5258
5259 022316 012703 100000
5260 022322 016705 157706
5261 022326 046705 157742
5262 022332 010567 000352
5263 022336 005067 000344
    
```

```

.SBTTL HARDWARE TEST - OAUTOA -
;*****
;*
;* - OAUTO BIT ACTIVE TEST -
;*
;* This test verifies that the DUT's OAUTO function behaves correctly
;* when active, ie OAUTO bit asserted high.
;* This test will only execute if the staggered loopback mode is selected.
;* The special staggered loopback connector must be fitted.
;*
;--*****
      BGNTST
      T5::
      CMPB LOPBCK,#2 ;CHECK MODE SELECTED.
      BEQ .+6 ;DO NOT EXIT IF STAGGERD LOPBCK MODE SELECTED.
      JMP 60$ ;EXIT THIS TEST.
      TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (50)
      MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
      MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
      MOV #5001,ERRNBR ;SET ERROR NUMBER TO 5001.
      MOV #EM5001,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
      MOV #ER9101,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
;+
; Reset the DUT to a known state, remove the status codes from the fifo.
; Clear TX and RX interrupt enable bits in the CSR.
; This subroutine reports error >>>> 5001 <<<<.
;-
      JSR PC,CLNRST ;RESET THE DHV11 M, REPORT ANY ERRORS FOUND.
      BCS .+6 ;DO NOT EXIT IF RESET WAS SUCCESSFUL.
      JMP 60$ ;EXIT THIS TEST.
;+
; Set-up the associated TX/RX line number tables.
;-
      JSR PC,ASLNTL ;INITIALISE THE ASSOCIATED TX/RX TABLES.
;+
; Set external loopback,enable OAUTO and receiver functions on all active lines
; Set LPR on all lines to 38.4k baud, 8 bits per character, odd parity,
; 2 stop bits.
; Enable transmitters on all lines.
;-
      MOV ACTLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
      MOV #24,R0 ;PASS THE LNCTRL CONTENTS.
      JSR PC,WTWLNC ;INITIALISE THE LNCTRL REGISTERS.
      MOV #MAPLNS,R5 ;PASS BIT MAP OF ALL LINES.
      MOV #177670,R0 ;PASS THE LPR CONTENTS.
      JSR PC,WTWLPR ;INITIALISE THE LPR REGISTERS ON ALL LINES.
      JSR PC,TXENBL ;ENABLE TRANSMITTERS ON ALL LINES.
;+
; Set up outer loop for testing active lines in both line groups.
;
      MOV #100000,R3 ;SET UP LOOP CONTROL FLAG.
      MOV ACTLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
      BIC LGRP2M,R5 ;REMOVE LINES IN GROUP 2.
2$: MOV R5,45$ ;SAVE THE CURRENT LINE GROUP.
      CLR 40$ ;CLEAR THE LINE NUMBER COUNTER.
    
```

## HARDWARE TEST

## OAUTOA -

```

5264 022342 016701 000340      4$:  MOV    40$,R1      ;COPY THE LINE NUMBER.
5265 022346 000241              CLC                ;CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
5266 022350 006005              ROR     R5          ;SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.
5267 022352 103054              BCC    8$          ;SKIP TESTING THIS LINE IF IT IS INACTIVE.
5268
5269
5270
5271
5272 022354 012767 011612 161524      MOV    #5002.,ERRNBR ;SET THE ERROR NUMBER TO 5002.
5273 022362 010177 157654      MOV    R1,@CSRA     ;SELECT THE LINE TO BE TESTED.
5274 022366 032777 000020 157656      BIT    @BIT4,@LNCTRA ;TEST THE STATE OF THE OAUTO BIT.
5275 022374 001004              BNE    6$          ;SKIP ERROR REPORT IF OAUTO BIT IS SET.
5276 022376 012702 006250      MOV    #EM4902,R2   ;PASS THE ERROR MESSAGE.
5277
5278 022402              ERROR            ; "OAUTO BIT BAD ON LINE nn"
5279 022402 104460              >>>> ERROR #5002 <<<<.
5280 022404 000437              BR     8$          ;SKIP FURTHER TESTING OF THIS LINE.
5281
5282
5283 022406 116177 004044 157626 6$:  MOV    TXRLNB(R1),@CSRA ;SELECT THE ASSOCIATED TX LINE.
5284 022414 012777 100023 157622      MOV    #100023,@TXCHA ;TRANSMIT THE XOFF CHARACTER TO THE LUT.
5285
5286
5287
5288 022422 005267 161460      INC    ERRNBR       ;INCREMENT ERROR NUMBER TO 5003.
5289 022426 012701 170012      MOV    #170012,R1   ;TEST BIT 15, TIMEOUT OF 10 MILLI SECS.
5290 022432 016702 157604      MOV    CSRA,R2      ;PASS THE ADDRESS OF THE REGISTER TO TEST.
5291 022436 004767 174134      JSR    PC,WAIBIS    ;WAIT FOR DMA TO COMPLETE.
5292 022442 103123              BCC    50$         ;ABORT TEST IF TIMEOUT OCCURRED.
5293 022444 012704 000005      MOV    #5,R4        ;PASS TIME-OUT OF 5 MILLI SECS.
5294 022450 004767 171244      JSR    PC,DELAY     ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
5295
5296
5297
5298
5299 022454 005267 161426      INC    ERRNBR       ;INCREMENT ERROR NUMBER TO 5004.
5300 022460 016701 000222      MOV    40$,R1       ;GET THE NUMBER OF THE LINE TEST.
5301 022464 010177 157552      MOV    R1,@CSRA     ;SELECT THE LINE CURRENTLY UNDER TEST.
5302 022470 005777 157562      TST    @TXAD2A      ;TEST THE STATE OF THE TX_ENABLE BIT.
5303 022474 100003              BPL    8$          ;SKIP ERROR REPORT IF BIT IS CLEAR.
5304 022476 012702 006250      MOV    #EM4902,R2   ;PASS THE MESSAGE TO BE REPORTED.
5305
5306 022502              ERROR            ; "OAUTO BIT BAD ON LINE nn"
5307 022502 104460              >>>> ERROR #5004 <<<<.
5308 022504 005267 000176      8$:  INC    40$         ;INCREMENT THE LINE NUMBER,
5309 022510 005705              TST    R5          ;CHECK IF THERE ARE ANY MORE LINES TO TEST.
5310 022512 001313              BNE    4$          ;
5311
5312
5313
5314 022514 016705 000170      MOV    45$,R5       ;RESTORE THE CURRENT LINE ACTIVE LINE GROUP.
5315 022520 004767 173412      JSR    PC,TXDSBL    ;DISABLE TRANSMITTERS ON THE SELECTED LINES.
5316 022524 016705 000160      MOV    45$,R5       ;GET THE CURRENT LINE ACTIVE LINE GROUP AGAIN.
5317 022530 005067 000152      CLR    40$         ;CLEAR THE LINE COUNTER.
5318 022534 012767 011615 161344 10$:  MOV    #5005.,ERRNBR ;SET ERROR NUMBER TO 5005.

```

HARDWARE TEST

- OAUTOA -

```

5319 022542 016701 000140      MOV    40$,R1      ;COPY THE LINE NUMBER.
5320 022546 000241              CLC              ;CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
5321 022550 006005              ROR     R5        ;SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.
5322 022552 103035              BCC    12$       ;SKIP TESTING THIS LINE IF IT IS INACTIVE.
5323                          ;+
5324                          ; Transmit the XON (ASCII DC1) on the associated line.
5325                          ;-
5326 022554 116177 004044 157460  MOVB   TXRLNB(R1),@CSRA ;SELECT THE ASSOCIATED TX LINE.
5327 022562 012777 100021 157454  MOV    @100021,@TXCHA ;TRANSMIT THE XON CHARACTER TO THE LUT.
5328                          ;+
5329                          ; Wait for transmission to complete.
5330                          ;-
5331 022570 012701 170012      MOV    @170012,R1  ;TEST BIT 15, TIMEOUT OF 10 MILLI SECS.
5332 022574 016702 157442      MOV    CSRA,R2    ;PASS THE ADDRESS OF THE REGISTER TO TEST.
5333 022600 004767 173772      JSR    PC,WAIBIS  ;WAIT FOR DMA TO COMPLETE.
5334 022604 103042              BCC    50$       ;ABORT TEST IF TIMEOUT OCCURRED.
5335 022606 012704 000005      MOV    @5,R4     ;PASS TIME-OUT OF 5 MILLI SECS.
5336 022612 004767 171102      JSR    PC,DELAY  ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
5337                          ;+
5338                          ; Test the state of the TX_ENABLE bit on the line under test.
5339                          ; Report error if TX_ENABLE bit is clear.
5340                          ;-
5341 022616 005267 161264      INC    ERRNBR    ;INCREMENT ERROR NUMBER TO 5006.
5342 022622 016701 000060      MOV    40$,R1    ;GET THE NUMBER OF THE LINE UNDER TEST.
5343 022626 010177 157410      MOV    R1,@CSRA  ;SELECT THE LINE CURRENTLY UNDER TEST.
5344 022632 005777 157420      TST    @TXAD2A   ;TEST THE STATE OF THE TX_ENABLE BIT.
5345 022636 100403              BMI    12$       ;SKIP ERROR REPORT IF BIT IS CLEAR.
5346 022640 012702 006250      MOV    @EM4902,R2 ;PASS THE MESSAGE TO BE REPORTED.
5347                          ; "OAUTO BIT BAD ON LINE nn".
5348 022644              ERROR                          ; >>>> ERROR #5006 <<<<<.
5349 022644 104460              TRAP   C$ERROR
5350 022646 005267 000034      12$: INC    40$    ;INCREMENT THE LINE NUMBER.
5351 022652 005705              TST    R5        ;CHECK IF THERE ARE ANY MORE LINES TO TEST.
5352 022654 001327              BNE    10$      ;
5353                          ;+
5354                          ; Check loop control flag to determine if both sets of lines have been tested
5355                          ; If this is the first time around, re-enable TX on all lines, generate active
5356                          ; bit map for second line group.
5357                          ;-
5358 022656 005703              TST    R3        ;HAVE BOTH LINE GROUPS BEEN TESTED?.
5359 022660 001416              BEQ    60$       ;YES; THEN EXIT THIS TEST.
5360 022662 005003              CLR    R3        ;NO; CLEAR THE LOOP CONTROL FLAG.
5361 022664 012705 000377      MOV    @MAPLNS,R5 ;PASS THE BIT MAP OF ALL AVAILABLE LINE.
5362 022670 004767 173336      JSR    PC,TXENBL ;RE-ENABLE TRANSMISSION ON ALL LINES.
5363 022674 016705 157334      MOV    ACTLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
5364 022700 046705 157366      BIC    LGRP1M,R5 ;REMOVE ALL ACTIVE LINES IN GROUP 1.
5365 022704 000612              BR     2$       ;ONCE MORE AROUND AND WE ARE DONE.
5366                          ;+
5367 022706 000000      40$: .WORD 0      ;STORAGE FOR CURRENT LINE NUMBER.
5368 022710 000000      45$: .WORD 0      ;STORAGE FOR CURRENT ACTIVE LINE BIT MAP.
5369 022712 004767 173064      50$: JSR    PC,TSABRT ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
5370 022716 005067 157342      60$: CLR    CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5371                          ;-
5372 022722              ENDTST
022722              L10027: TRAP   C$ETST
022722 104401

```



HARDWARE TEST

- OAUTOA -

```

5374
5375 .SBTTL HARDWARE TEST - IAUTOI -
5376 ;*****
5377 ;*
5378 ;*
5379 ;* This test verifies that the DUT's IAUTO function behaves correctly
5380 ;* when inactive, ie. IAUTO bit clear.
5381 ;* All active lines are tested individually by filling the FIFO
5382 ;* then reading the received data checking for the presence of
5383 ;* XOFF(ASCII DC3) or XON (ASCII DC1) characters.
5384 ;* If any are found then appropriate errors are reported.
5385 ;* Any BMP codes that are found will be placed on the BMP code queue,
5386 ;* to be reported later.
5387 ;* The characters are transmitted on all active lines, in internal
5388 ;* loopback mode.
5389 ;*
5390 ; -*****
5391
5392 022724 BGNTST
5393 022724
5393 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
5394 022724 000006 000006 157334 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (51)
5395 022732 012767 177777 157324 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
5396 022740 012767 000001 161136 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE
5397 022746 012767 011755 161132 MOV #5101,ERRNBR ;SET ERROR NUMBER TO 5101.
5398 022754 012767 006332 161126 MOV #EM5101,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
5399 022762 012767 012572 161122 MOV #ER9101,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
5400
5401 ;+
5402 ; Reset the DUT to a known state, remove the status codes from the fifo.
5403 ; Clear TX and RX interrupt enable bits in the CSR.
5404 ; This subroutine reports error >>>> 5101 <<<<.
5405 022770 004767 170512 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
5406 022774 103146 BCC 60$ ;EXIT TEST IF FATAL ERROR FOUND.
5407
5408 ;+
5409 ; Initialize the 256 byte data pattern.
5410 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
5411 ; Note: the first two characters and the last two characters will be the same.
5412
5413 022776 004767 171242 JSR PC,INDTPX ;INITIALISE DATA PATTERN.
5414
5415 ;+
5416 ; Set internal loopback, disable IAUTO, enable receiver on the selected line.
5417 ; Set LPR to 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
5418 023002 016705 157226 MOV ACTLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
5419 023006 012700 000204 MOV #204,R0 ;PASS INT'L LOPBCK, ENABLE RX, DISABLE IAUTO.
5420 023012 004767 173674 JSR PC,WTWLNCR ;INITIALISE THE LINE CONTROL REGISTER.
5421 023016 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
5422 023022 004767 173714 JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
5423 023026 012704 000012 MOV #10,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
5424 023032 004767 170662 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
5425
5426 ;+
5427 ; Set up loop for all active lines.
5428 ; Test the state of the IAUTO bit prior to transmitting the data pattern.
5429 ; If the bit is set, then report the error and skip transmitting

```

## HARDWARE TEST

- IAUTOI -

```

5430 ; the data pattern on the selected line.
5431 ; Transmit a 256 character data pattern using DMA, on a single channel
5432 ; Empty the fifo, and verify no XOFF or XON chars were found.
5433 ;-
5434 023036 005001          CLR    R1          ;CLEAR THE LINE NUMBER COUNTER.
5435 023040 005067 000244  CLR    55$        ;CLEAR STORAGE FOR LINE NUMBER.
5436 023044 012767 011756 161034 2$:  MOV    #5102.,ERRNBR ;SET THE ERROR NUMBER TO 5102.
5437 023052 004767 171756          JSR    PC,PUFIFO    ;PURGE THE FIFO.
5438 023056 103111          BCC    50$        ;GO REPORT ERROR IF FIFO DID NOT PURGE.
5439 023060 000241          CLC                    ;CLEAR CARRY PRIOR TO ROTATING BIT MAP.
5440 023062 006005          ROR    R5          ;ROTATE THE BIT MAP INTO THE CARRY BIT.
5441 023064 103077          BCC    12$        ;BRANCH IF LINE IS INACTIVE.
5442 ;+
5443 ; Test the IAUTO bit on the selected active line.
5444 ; Report error if it is set.
5445 ; Do not transmit the data pattern on the selected line.
5446 ;-
5447 023066 005267 161014          INC    ERRNBR      ;SET ERROR NUMBER TO 5103.
5448 023072 010177 157144          MOV    R1,@CSRA   ;SELECT LINE TO TEST.
5449 023076 032777 000002 157146  BIT    #BIT1,@LNCTRA ;TEST THE STATE OF THE IAUTO BIT ON THIS LINE.
5450 023104 001404          BEQ    4$          ;SKIP ERROR IF IAUTO BIT CLEAR.
5451 023106 012702 006360          MOV    #EM5102,R2 ;PASS THE CORRECT ERROR MESSAGE.
5452 023112          ERROR          ;
5453 023114 000463          BR     12$        ;SKIP TRANSMITTING DATA PATTERN. TRAP    C$ERROR
5454
5455 ;+
5456 ; Transmit data pattern of 256 chars.
5457 ;-
5458 023116 005267 160764          INC    ERRNBR      ;SET ERROR NUMBER TO 5104.
5459 023122 012702 002704          MOV    #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
5460 023126 012703 000400          MOV    #256.,R3   ;PASS THE LENGTH OF THE DATA PATTERN.
5461 023132 004767 170622          JSR    PC,DODMA   ;TRANSMIT THE DATA PATTERN.
5462 023136 103061          BCC    50$        ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
5463
5464 ;+
5465 ; Wait for DMA to complete, then wait for the last character plus XOFF
5466 ; to arrive in the fifo.
5467 ;-
5468 023140 005267 160742          INC    ERRNBR      ;SET ERROR NUMBER TO 5105.
5469 023144 012701 170536          MOV    #170536,R1 ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
5470 023150 016702 157066          MOV    CSRA,R2   ;PASS THE ADDRESS OF THE CSR.
5471 023154 004767 173416          JSR    PC,WAIBIS  ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
5472 023160 103050          BCC    50$        ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
5473 023162 012704 000012          MOV    #10.,R4   ;PASS DELAY OF 10 MILLI SECS.
5474 023166 004767 170526          JSR    PC,DELAY   ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
5475
5476 ;+
5477 ; Read 256 chars from the fifo. Report error if any XOFF's or XON's
5478 ; are found.
5479 ;-
5480 023172 005267 160710          INC    ERRNBR      ;INCREMENT ERROR NUMBER TO 5106.
5481 023176 012701 000400          MOV    #256.,R1   ;INITIALISE THE READ COUNTER.
5482 023202 017702 157036          MOV    @RBUFA,R2  ;READ CHAR FROM THE FIFO.
5483 023206 100035          BPL    50$        ;GO REPORT ERROR IF FIFO EMPTY.
5484
5485 ;+
5485 ; Check for BMP code in the fifo. Save any found on the queue.

```

## HARDWARE TEST

- IAUTOI -

```

5486
5487 023210 012700 170301      ;
5488 023214 040200              MOV    #170301,R0      ;SET UP BMP BIT MASK.
5489 023216 001002              BIC    R2,R0          ;TRY TO CLEAR ALL THE BMP BITS.
5490 023220 004767 172320      BNE    8$             ;SKIP BMPSAV IF NOT A BMP CODE.
5491                               JSR    PC,SAVBMP       ;SAVE THE BMP CODE ON THE QUEUE.
5492                               ;+
5493                               ; Check for XOFF and XON characters.
5494 023224 120227 000023      8$:   CMPB   R2,#23      ;IS IT AN XOFF CHARACTER?.
5495 023230 001406              BEQ    10$            ;YES; GO REPORT ERROR.
5496 023232 120227 000021      CMPB   R2,#21      ;NO; IS IT AN XON CHARACTER?.
5497 023236 001403              BEQ    10$            ;YES; GO REPORT ERROR.
5498 023240 005301              DEC    R1            ;DECREMENT THE READ COUNT.
5499 023242 001357              BNE    6$             ;LOOP TO READ THE NEXT CHAR.
5500 023244 000407              BR     12$           ;GO CHECK FOR ANY UNTESTED ACTIVE LINES.
5501
5502 023246 005267 160634      10$:  INC    ERRNBR        ;SET ERROR NUMBER TO 5107.
5503 023252 016701 000032      MOV    55$,R1        ;PASS THE LINE NUMBER TO BE REPORTED.
5504 023256 012702 006416      MOV    #EM5103,R2    ;PASS THE ERROR MESSAGE TO BE REPORTED.
5505 023262 104460              ERROR                   ;
5506                               ;>>>> ERROR <<<<<<
5507                               ; TRAP    C$ERROR
5508                               ;+
5509                               ; Check if all active lines have been tested.
5509 023264 005267 000020      12$:  INC    55$           ;INCREMENT LINE NUMBER.
5510 023270 016701 000014      MOV    55$,R1        ;GET NUMBER OF THE NEXT LINE TO TEST.
5511 023274 005705              TST    R5            ;ARE THERE ANY MORE ACTIVE LINES TO TEST?.
5512 023276 001262              BNE    2$             ;LOOP TO CHECK NEXT LINE.
5513 023300 000404              BR     60$           ;EXIT TEST.
5514
5515 023302 004767 172474      50$:  JSR    PC,TSABRT     ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
5516 023306 000401              BR     60$           ;EXIT THIS TEST.
5517 023310 000000      55$:  .WORD  0            ;STORAGE FOR LINE NUMBER.
5518 023312 005067 156746      60$:  CLR    CTRLCF       ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5519
5520 023316              ENDTST
5520 023316
5520 023316 104401              L10030: TRAP    C$ETST

```

HARDWARE TEST

- IAUTOA -

```

5522 .SBTTL HARDWARE TEST - IAUTOA -
5523 ;*****
5524 ;
5525 ; IAUTO BIT ACTIVE TEST -
5526 ;
5527 ; This test verifies that the DUT's IAUTO function behaves correctly
5528 ; when active, ie IAUTO asserted high.
5529 ; All active lines are tested individually by filling the FIFO, and
5530 ; checking for the presence of at least one XOFF(ASCII DC3) character
5531 ; and one XON (ASCII DC1) character.
5532 ; Any BMP codes that are found will be placed on the BMP code queue,
5533 ; to be reported later.
5534 ; The characters are transmitted on all active lines, in internal
5535 ; loopback mode.
5536 ;
5537 ;--*****
5538 023320 BGNTST
5539 023320 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
5540 023320 000007 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (52)
5541 023326 012767 000007 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
5542 023334 012767 177777 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
5543 023342 012767 000001 MOV #5201,ERRNBR ;SET ERROR NUMBER TO 5201.
5544 023350 012767 012121 MOV #EM5201,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
5545 023356 012767 006446 MOV #ER9101,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
5546 ;
5547 ;+
5548 ; Reset the DUT to a known state, remove the status codes from the fifo.
5549 ; Clear TX and RX interrupt enable bits in the CSR.
5550 ; This subroutine reports error >>>> 5201 <<<<.
5551 023364 004767 170116 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
5552 023370 103161 BCC 60$ ;EXIT TEST IF FATAL ERROR FOUND.
5553 ;
5554 ;+
5555 ; Initialize the 256 byte data pattern.
5556 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
5557 ; Note: the first two characters and the last two characters will be the same.
5558 ;
5559 023372 004767 170646 JSR PC,INDTPX ;INITIALISE DATA PATTERN.
5560 ;+
5561 ; Set internal loopback, enable IAUTO and receiver on the selected line.
5562 ; Set LPR to 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
5563 ;-
5564 023376 016705 156632 MOV ACTLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
5565 023402 012700 000206 MOV #206,R0 ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.
5566 023406 004767 173300 JSR PC,WTWLNCR ;INITIALISE THE LINE CONTROL REGISTER.
5567 023412 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
5568 023416 004767 173320 JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
5569 023422 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
5570 023426 004767 170266 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
5571 ;
5572 ;+
5573 ; Set up loop for all active lines.
5574 ; Test the state of the IAUTO bit prior to transmitting the data pattern.
5575 ; If the bit is clear, then report the error and skip transmitting
5576 ; the data pattern on the selected line.
5577 ; Transmit a 224 character data pattern using DMA, on a single channel

```

## HARDWARE TEST

- IAUTOA -

```

5578 ; Empty the fifo, and count the XOFF and an XON chars found.
5579 ;
5580 023432 005001 CLR R1 ;CLEAR THE LINE NUMBER COUNTER.
5581 023434 005067 000272 CLR 55$ ;CLEAR STORAGE FOR LINE NUMBER.
5582 023440 012767 012122 160440 2$: MOV #5202.,ERRNBR ;SET THE ERROR NUMBER TO 5202.
5583 023446 004767 171362 JSR PC,PUFIFO ;PURGE THE FIFO.
5584 023452 103124 BCC 50$ ;GO REPORT ERROR IF FIFO DID NOT PURGE.
5585 023454 000241 CLC ;CLEAR CARRY PRIOR TO ROTATING BIT MAP.
5586 023456 006005 ROR R5 ;ROTATE THE BIT MAP INTO THE CARRY BIT.
5587 023460 103112 BCC 16$ ;BRANCH IF LINE IS INACTIVE.
5588 ;+
5589 ; Test the IAUTO bit on the selected active line.
5590 ; Report error if it is clear.
5591 ; Do not transmit the data pattern on the selected line.
5592 ;-
5593 023462 005267 160420 INC ERRNBR ;SET ERROR NUMBER TO 5203.
5594 023466 010177 156550 MOV R1,@CSRA ;SELECT LINE TO TEST.
5595 023472 032777 000002 156552 BIT #BIT1,@LNCTRA ;TEST THE STATE OF THE IAUTO BIT ON THIS LINE.
5596 023500 001004 BNE 4$ ;SKIP ERROR IF IAUTO BIT SET.
5597 023502 012702 006472 MOV #EM5202,R2 ;PASS THE CORRECT ERROR MESSAGE.
5598 ; "IAUTO BIT FOUND CLEAR ON LINE nn"
5599 023506 ERROR ; >>>> ERROR <<<<<. TRAP C$ERROR
023506 104460
5600 023510 000476 BR 16$ ;SKIP TRANSMITTING DATA PATTERN.
5601
5602 ;+
5603 ; Transmit data pattern to fill the fifo, 223 chars + 32 xoff's + xon.
5604 ;-
5605 023512 005267 160370 4$: INC ERRNBR ;SET ERROR NUMBER TO 5204.
5606 023516 012702 002704 MOV #BUF8AS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
5607 023522 012703 000337 MOV #223.,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
5608 023526 004767 170226 JSR PC,DODMA ;TRANSMIT THE DATA PATTERN.
5609 023532 103074 BCC 50$ ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
5610
5611 ;+
5612 ; Wait for DMA to complete, then wait for the last character plus XOFF
5613 ; to arrive in the fifo.
5614 ;-
5615 023534 005267 160346 INC ERRNBR ;SET ERROR NUMBER TO 5205.
5616 023540 012701 170536 MOV #170536,R1 ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
5617 023544 016702 156472 MOV CSRA,R2 ;PASS THE ADDRESS OF THE CSR.
5618 023550 004767 173022 JSR PC,WAIBIS ;WAIT FOR DMA TO COMPLETE, TX ACTION SET.
5619 023554 103063 BCC 50$ ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
5620
5621 ;+
5622 ; Read 256 chars from the fifo, count any XOFF or XON chars found.
5623 ;-
5624 023556 005003 CLR R3 ;CLEAR XOFF AND XON INDICATOR.
5625 023560 005267 160322 INC ERRNBR ;INCREMENT ERROR NUMBER TO 5206.
5626 023564 012701 000400 MOV #256.,R1 ;INITIALISE THE READ COUNTER.
5627 023570 017702 156450 6$: MOV @RBUFA,R2 ;READ CHAR FROM THE FIFO.
5628 023574 100053 BPL 50$ ;GO REPORT ERROR IF FIFO EMPTY.
5629 ;+
5630 ; Check for BMP code in the fifo. Save any found on the queue.
5631 ;-
5632 023576 012700 170301 MOV #170301,R0 ;SET UP BMP BIT MASK.
5633 023602 040200 BIC R2,R0 ;TRY TO CLEAR ALL THE BMP BITS.

```

## HARDWARE TEST

- IAUTOA -

```

5634 023604 001002      BNE      8$          ;SKIP BMPSAV IF NOT A BMP CODE.
5635 023606 004767 171732 JSR      PC,SAVBMP   ;SAVE THE BMP CODE ON THE QUEUE.
5636                                     ;+
5637                                     ; Check for XOFF and XON characters.
5638                                     ;-
5639 023612 120227 000023 8$:      CMPB     R2,#23   ;IS IT AN XOFF CHARACTER?.
5640 023616 001002      BNE      10$        ;NO, BRANCH TO SEE IF IT IS AN XON.
5641 023620 052703 000001      BIS      #BIT0,R3   ;INDICATE THE XOFF CHAR.
5642 023624 120227 000021 10$:     CMPB     R2,#21   ;IS IT AN XON CHARACTER?.
5643 023630 001002      BNE      12$        ;NO, SKIP THE NEXT INSTRUCTION.
5644 023632 052703 000002      BIS      #BIT1,R3   ;INDICATE THE XON CHAR.
5645 023636 005301 12$:     DEC      R1          ;DECREMENT THE READ COUNT. 25C READ?
5646 023640 001410      BEQ      14$        ;YES, EXIT THE LOOP.
5647 023642 020127 000300      CMP      R1,#192.   ;NO. HAVE WE READ EXACTLY 192 CHARS?
5648 023646 001350      BNE      6$          ;NO, LOOP TO READ MORE.
5649 023650 012704 000012      MOV      #10.,R4    ;YES. DELAY 10MS TO WAIT FOR THE XON CHAR
5650 023654 004767 170040      JSR      PC,DELAY    ; (WHICH IS LAST CHAR) TO GET INTO FIFO.
5651 023660 000743      BR       6$          ;LOOP TO READ MORE CHARS.
5652                                     ;+
5653                                     ; Verify than at least 1 XOFF and 1 XON was found in the fifo.
5654                                     ; Report error if none were found.
5655                                     ;-
5656 023662 020327 000003 14$:     CMP      R3,#3     ;DID WE GET AT LEAST 1 XON AND 1 XOFF?
5657 023666 001407      BEQ      16$        ;YES, SKIP ERROR REPORT.
5658 023670 005267 160212      INC      ERRNBR     ;NO, SET ERROR NUMBER TO 5207.
5659 023674 016701 000032      MOV      55$,R1    ;PASS THE LINE NUMBER TO BE REPORTED.
5660 023700 012702 006416      MOV      #EM5103,R2 ;PASS THE ERROR MESSAGE TO BE REPORTED.
5661                                     ; "IAUTO BIT BAD ON LINE nn".
5662 023704      ERROR      ; >>>> ERROR <<<<<.
5663 023704 104460      TRAP      C$ERROR
5664                                     ;+
5665                                     ; Check if all active lines have been tested.
5666 023706 005267 000020 16$:     INC      55$     ;INCREMENT LINE NUMBER.
5667 023712 016701 000014      MOV      55$,R1    ;GET NUMBER OF THE NEXT LINE TO TEST.
5668 023716 005705      TST      R5        ;ARE THERE ANY MORE ACTIVE LINES TO TEST?.
5669 023720 001247      BNE      2$        ;LOOP TO CHECK NEXT LINE.
5670 023722 000404      BR       60$      ;EXIT TEST.
5671
5672 023724 004767 172052 50$:     JSR      PC,TSABRT  ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.
5673 023730 000401      BR       60$      ;EXIT THIS TEST.
5674 023732 000000 55$:     .WORD     0          ;STORAGE FOR LINE NUMBER.
5675 023734 005067 156324 60$:     CLR      CTRLCF    ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5676
5677 023740      ENDTST
5678 023740
5679 023740 104401      L10031: TRAP      C$ETST

```

HARDWARE TEST - FIFDAT -

```

5679 .SBTTL HARDWARE TEST - FIFDAT -
5680 ;*****
5681 ;*
5682 ;*
5683 ;* This test verifies that the DUT is capable of holding 256 valid
5684 ;* characters in its fifo.
5685 ;* The characters are transmitted on the first available active line, in
5686 ;* internal loopback mode.
5687 ;* The data found in the fifo is compared with the expected data, and any
5688 ;* discrepancies are reported.
5689 ;* Any BMP code found will invalidate the test and cause it to be aborted.
5690 ;* However the BMP code will be placed on the BMP code queue, to be
5691 ;* reported later.
5692 ;*
5693 ;*
5694 ;*
5695 ;-----*****
5695 023742 BGNTST
5696 023742 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
5696 000010 T8::
5697 023742 012767 000010 156316 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (53)
5698 023750 012767 177777 156306 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
5699 023756 012767 000001 160120 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
5700 023764 012767 012265 160114 MOV #5301,ERRNBR ;SET ERROR NUMBER TO 5301.
5701 023772 012767 006530 160110 MOV #EM5301,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
5702 ;*
5703 ; Reset the DUT to a known state, remove the status codes from the fifo.
5704 ; Clear TX and RX interrupt enable bits in the CSR.
5705 ; This subroutine reports error >>>> 5301 <<<<<.
5706 ;-
5707 024000 004767 167502 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
5708 024004 103107 BCC 60$ ;EXIT TEST IF FATAL ERROR FOUND.
5709 ;*
5710 ; Find an active line on which to perform the test.
5711 ; Initialise 256 byte data pattern.
5712 ;-
5713 024006 004767 170122 JSR PC,FINACT ;FIND AN ACTIVE LINE.
5714 024012 103104 BCC 60$ ;EXIT IF NO ACTIVE LINES FOUND.
5715 024014 004767 170174 JSR PC,INDATP ;INITIALISE THE DATA PATTERN.
5716 ;*
5717 ; Transmit a 265 character data pattern using DMA, on a single channel
5718 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
5719 ;-
5720 ;*
5721 ; Set internal loopback on the selected line.
5722 ; Transmit the data pattern on the first available active line.
5723 ;-
5724 024020 012700 000204 MOV #204,R0 ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE RX.
5725 024024 004767 172662 JSR PC,WTWLNLC ;INITIALISE THE LINE CONTROL REGISTER.
5726 024030 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
5727 024034 004767 172702 JSR PC,WTWLPRLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
5728 024040 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
5729 024044 004767 167650 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
5730 024050 012702 002704 MOV #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
5731 024054 012703 000400 MOV #BUFMID-BUFBAS,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
5732 024060 005267 160022 INC ERRNBR ;SET ERROR NUMBER TO 5302.
5733 024064 004767 167670 JSR PC,DODMA ;TRANSMIT THE DATA PATTERN.
5734 024070 103053 BCC 50$ ;ABORT TEST IF ERROR FOUND DURING DMA TX.

```

HARDWARE TEST

- FIFDAT -

```

5735
5736      ; Wait for DMA to complete, then wait for the last character to arrive in
5737      ; the fifo.
5738      ;-
5739 024072 005267 160010      INC      ERRNBR      ;SET ERROR NUMBER TO 5303.
5740 024076 010103      MOV      R1,R3      ;SAVE THE NUMBER OF THE SELECTED ACTIVE LINE.
5741 024100 012701 170536      MOV      #170536,R1  ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
5742 024104 016702 156132      MOV      CSRA,R2    ;PASS THE ADDRESS OF THE CSR.
5743 024110 004767 172462      JSR      PC,WAIBIS  ;WAIT FOR DMA TO COMPLETE, TX ACTION SET.
5744 024114 103041      BCC     50$        ;BRANCH IF FIFO EMPTY, ABORT THE TEST.
5745 024116 012704 000005      MOV      #5,R4     ;PASS DELAY OF 5 MILLI SECS.
5746 024122 004767 167572      JSR      PC,DELAY   ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
5747
5748      ; Read the FIFO checking for data corruption, report any errors found.
5749      ; Abort the test if a BMP code was found in the fifo.
5750      ;-
5751 024126 006303      ASL     R3         ;MULTIPLY BY 2.
5752 024130 005004      CLR     R4         ;INITIALISE THE EXPECTED DATA.
5753 024132 016705 156106      MOV     RDUFA,R5   ;GET THE ADDRESS OF THE RECEIVER BUFFER REG.
5754 024136 012767 012270 157742 2$:  MOV     #5304,ERRNBR ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
5755 024144 011502      MOV     (R5),R2    ;GET THE ACTUAL DATA FROM THE FIFO.
5756 024146 100024      BPL    50$        ;ABORT THE TEST IF THE FIFO IS EMPTY.
5757
5758      ; Check if the read character is a BMP code.
5759      ; If it is a BMP code save it on the queue to be reported later, and
5760      ; abort the test.
5761      ;-
5762 024150 005267 157732      INC     ERRNBR     ;SET ERROR NUMBER TO 5305.
5763 024154 004767 167226      JSR     PC,CHKBMP  ;CHECK IF CHARACTER IS A BMP CODE.
5764 024160 103002      BCC    4$         ;BRANCH IF NOT A BMP CODE.
5765 024162 104460      ERROR  >>>> ERROR 5305 <<<<<.      TRAP    C$ERROR
5766 024164 000417      BR     60$        ;ABORT THIS TEST.
5767
5768 024166 005267 157714      4$:    INC     ERRNBR     ;SET ERROR NUMBER TO 5306.
5769 024172 120402      CMPB   R4,R2      ;COMPARE THE EXPECTED WITH THE ACTUAL DATA.
5770 024174 001406      BEQ    8$         ;SKIP ERROR REPORT IF DATA IS OK.
5771 024176 012767 012424 157706      MOV     #ER9002,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
5772 024204 012701 006555      MOV     #EM5302,R1 ;PASS THE MESSAGE TO BE REPORTED.
5773      ;REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"
5774 024210 6$:    ERROR  >>>> ERROR 5306 <<<<<.      TRAP    C$ERROR
5775 024212 104460      8$:    INCB   R4         ;INCREMENT THE EXPECTED DATA.
5776 024214 105204      BNE    2$         ;LOOP IF NOT DONE.
5777 024216 000402      BR     60$        ;EXIT
5778
5779 024220 004767 171556      50$:   JSR     PC,TSABRT  ;ABORT THE TEST, REASON SHOWN BY ERROR NUMBER.
5780 024224 005067 156034      60$:   CLR     CTRLCF    ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5781
5782 024230      ENDTST
5782 024230
5782 024230 104401      L10032:  TRAP    C$ETST

```



HARDWARE TEST - FI3QLI -

```

5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800 024232
      024232
5801      000011
5802 024232 012767 000011 156026
5803 024240 012767 177777 156016
5804 024246 012767 000001 157630
5805 024254 012767 012431 157624
5806 024262 012767 006705 157620
5807 024270 012767 011636 157614
5808
5809
5810
5811
5812
5813 024270 004767 167204
5814 024302 003111
5815
5816
5817
5818 024304 004767 167624
5819 024310 103106
5820
5821
5822
5823
5824
5825
5826 024312 004767 167726
5827
5828
5829
5830
5831
5832
5833
5834
5835 024316 012700 000206
5836 024322 004767 172364
5837 024326 012700 177670
5838 024332 004767 172404
5839 024336 012704 000012

```

```

.SBTTL HARDWARE TEST - FI3QLI -
;*****
; - FIFO 3/4 LEVEL INACTIVE TEST -
;
; This test verifies that the DUT's fifo 3/4 level alarm system
; remains inactive while it contains 191 characters or less.
; The test looks for an XOFF (ASCII DC3) character in the fifo.
; If any XOFF's are found an error will be reported and the test aborted.
; Any BMP code found will invalidate the test and cause it to be aborted.
; However the BMP code will be placed on the BMP code queue, to be
; reported later.
; The characters are transmitted on the first available active line, in
; internal loopback mode.
;*****
BGNTST
      T9::
      TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (54)
      MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
      MOV #1,ERRTYP ;SET FATAL ERROR TYPE IN ERROR TABLE.
      MOV #5401,ERRNBR ;SET ERROR NUMBER TO 5401.
      MOV #EM5401,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
      MOV #ER0503,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
;
; Reset the DUT to a known state, remove the status codes from the fifo.
; Clear TX and RX interrupt enable bits in the CSR.
; This subroutine reports error >>>> 5401 <<<<.
;
      JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
      BCC 60$ ;EXIT TEST IF FATAL ERROR FOUND.
;
; Find an active line on which to perform the test.
;
      JSR PC,FINACT ;FIND THE NUMBER OF THE FIRST ACTIVE LINE.
      BCC 60$ ;EXIT IF NO LINES ARE AVAILABLE.
;
; Initialize the 256 byte data pattern.
; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
; Note: the first two characters and the last two characters will be the same.
;
      JSR PC,INDTPX ;INITIALISE THE DATA PATTERN.
;
; Transmit a 191 character data pattern using DMA, on a single channel
; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
;
;
; Set internal loopback, enable IAUTO and RX on the selected line.
; Transmit the data pattern on the first available active line.
;
      MOV #206,R0 ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.
      JSR PC,WTWLNC ;INITILAISE THE LINE CONTROL REGISTER.
      MOV #177670,R0 ;PASS THE LPR CONTE'.TS.
      JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
      MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.

```

## HARDWARE TEST

- FI3QLI -

```

5840 024342 004767 167352      JSR    PC,DELAY      ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
5841 024346 012702 002704      MOV    #8UFBAS,R2   ;PASS THE START OF THE DATA PATTERN TO TX.
5842 024352 012703 000277      MOV    #191.,R3     ;PASS THE LENGTH OF THE DATA PATTERN.
5843 024356 004767 167376      JSR    PC,DODMA     ;TRANSMIT THE DATA PATTERN.
5844 024362 103057      BCC    50$          ;IF ERROR FOUND DURING DMA THEN ABORT TEST.
5845
5846
5847      ;+
5848      ; Wait for DMA to complete, then wait for the last character to arrive in
5849      ; the fifo.
5850 024364 005267 157516      INC    ERRNBR       ;SET ERROR NUMBER TO 5402.
5851 024370 012701 170454      MOV    #170454,R1   ;PASS TIME-OUT VALUE OF 300 MILLI SECS.
5852 024374 016702 155642      MOV    CSRA,R2      ;PASS THE ADDRESS OF THE CSR.
5853 024400 004767 172172      JSR    PC,WAIBIS    ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
5854 024404 103046      BCC    50$          ;IF FIFO EMPTY, REPORT ERROR, ABORT THE TEST.
5855 024406 012704 000005      MOV    #5,R4        ;PASS DELAY OF 5 MILLI SECS.
5856 024412 004767 167302      JSR    PC,DELAY     ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
5857
5858      ;+
5859      ; Read the contents of the fifo. If any of the following conditions occur
5860      ; report the error and abort the test;
5861      ; Fifo empty too soon.
5862      ; BMP code found.
5863      ; Xoff code found.
5864      ; Extra (192) character found in fifo.
5865 024416 005004
5866 024420 016705 155620      CLR    R4           ;CLEAR THE CHARACTER COUNT.
5867 024424 012767 012267 157454 2$:  MOV    RBUFA,R5     ;GET THE ADDRESS OF THE RECEIVER BUFFER REG.
5868 024432 011502      MOV    #5303.,ERRNBR ;SET ERROR NUMBER TO 5403.
5869 024434 100032      MOV    (R5),R2      ;GET THE ACTUAL DATA FROM THE FIFO.
5870 024436 005204      BPL    50$          ;FIFO EMPTY, ABORT TEST.
5871      INC    R4        ;COUNT THE CHARACTER.
5872
5873      ;+
5874      ; Check if the read character is a BMP code.
5875      ; If it is a BMP code save it on the queue to be reported later, and
5876      ; abort the test.
5877 024440 005267 157442      INC    ERRNBR       ;SET ERROR NUMBER TO 5404.
5878 024444 004767 166736      JSR    PC,CHKBMP    ;CHECK IF CHARACTER IS A BMP CODE.
5879 024450 103001      BCC    4$           ;BRANCH IF NOT A BMP CODE.
5880 024452 000421      ;REPORT ERROR "BMP CODE FOUND IN FIFO, TEST INVALIDATED".
5881      BR    8$         ;REPORT THE ERROR AND ABORT THE TEST.
5882
5883      ;+
5884      ; Check if the character is an XOFF. Report the error if one is found.
5885 024454 005267 157426      4$: INC    ERRNBR     ;SET ERROR NUMBER TO 5405.
5886 024460 122702 000023      CMPB   #23,R2       ;CHECK IF THE READ DATA IS AN XOFF.
5887 024464 001003      BNE    6$           ;BRANCH IF NOT AN XOFF.
5888 024466 012701 006744      MOV    #EM5402,R1   ;PASS THE MESSAGE TO BE REPORTED.
5889      ;REPORT THE ERROR "FIFO BAD, ALARM SIGNAL DEFECTIVE".
5890 024472 000411      BR    8$            ;GO REPORT THE ERROR AND ABORT THE TEST.
5891
5892 024474 005267 157406      6$: INC    ERRNBR     ;SET ERROR NUMBER TO 5406.
5893 024500 020427 000277      CMP    R4,#191.     ;CHECK IF WE HAVE READ ALL THE CHARACTERS.
5894 024504 001347      BNE    2$           ;LOOP BACK TO GET THE NEXT CHARACTER.
5895 024506 011502      MOV    (R5),R2      ;TRY TO READ AN EXTRA CHARACTER FROM THE FIFO.
5896 024510 100006      BPL    60$         ;EXIT IF NON FOUND.

```



HARDWARE TEST - FI3QLA -

```

5909 .SBTTL HARDWARE TEST - FI3QLA -
5910 ;*****
5911 ;
5912 ;
5913 ;
5914 ; This test verifies that the DUT's fifo 3/4 level alarm system
5915 ; becomes active when the fifo contains > 192 characters.
5916 ; The test compares the actual number of XOFF (ASCII DC3)
5917 ; characters that are found in the fifo with the expected number.
5918 ; An error will be reported, if the counts are found to differ.
5919 ; Any BMP code found will invalidate the test and cause it to be aborted.
5920 ; However the BMP code will be placed on the BMP code queue, to be
5921 ; reported later.
5922 ; The characters are transmitted on the first available active line, in
5923 ; internal loopback mode.
5924 ;
5925 ; *****
5926 024534 BGNTST
5927 024534 T10::
5928 000012 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
5929 012767 000012 155524 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (55)
5930 024542 012767 177777 155514 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
5931 024550 012767 000001 157326 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
5932 024556 012767 012575 157322 MOV #5501,ERRNBR ;SET ERROR NUMBER TO 5501.
5933 024564 012767 007005 157316 MOV #EM5501,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
5934 ;
5935 ; Reset the DUT to a known state, remove the status codes from the fifo.
5936 ; Clear TX and RX interrupt enable bits in the CSR.
5937 ; This subroutine reports error >>>> 5501 <<<<<.
5938 024572 004767 166710 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
5939 024576 103402 BCS .+6 ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
5940 024600 000167 000414 JMP 60$ ;EXIT TEST FATAL ERROR FOUND.
5941 ;
5942 ; Find an active line on which to perform the test.
5943 ;
5944 024604 004767 167324 JSR PC,FINACT ;FIND AN ACTIVE LINE.
5945 024610 103402 BCS .+6 ;SKIP EXIT OF TEST IF ACTIVE LINE FOUND.
5946 024612 000167 000402 JMP 60$ ;EXIT TEST.
5947 ;
5948 ; Initialize the 256 byte data pattern.
5949 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
5950 ; Note: the first two characters and the last two characters will be the same.
5951 ;
5952 024616 004767 167422 JSR PC,INDTPX ;INITIALISE DATA PATTERN.
5953 ;
5954 ; Transmit a 256 character data pattern using DMA, on a single channel
5955 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
5956 ;
5957 ;
5958 ; Set internal loopback, enable IAUTO and receiver on the selected line.
5959 ; Transmit the first 191 characters on the first available active line.
5960 ;
5961 024622 005267 157260 2$: INC ERRNBR ;SET ERROR NUMBER TO 5502.
5962 024626 012700 000206 MOV #206,R0 ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.
5963 024632 004767 172054 JSR PC,WTWLCN ;INITIALISE THE LINE CONTROL REGISTER.
5964 024636 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.

```

HARDWARE TEST

- FI3QLA -

```

5965 024642 004767 172074      JSR    PC,WTWLPB          ;SET THE LPR CONTENTS TO 38.4K BAUD.
5966 024646 012704 000012      MOV    #10.,R4           ;PASS DELAY TIME OF 10 MILLI SECONDS.
5967 024652 004767 167042      JSR    PC,DELAY          ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
5968 024656 010105              MOV    R1,R5             ;COPY THE LINE NUMBER.
5969 024660 012702 002704      MOV    #BUFBAS,R2       ;PASS THE START OF THE DATA PATTERN TO TX.
5970 024664 012703 000277      MOV    #191.,R3         ;PASS THE LENGTH OF THE DATA PATTERN.
5971 024670 004767 167064      JSR    PC,DODMA         ;TRANSMIT THE DATA PATTERN.
5972 024674 103147              BCC    50$              ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
5973
5974
5975      ;+
5975      ; Wait for DMA to complete, then wait for the last character to arrive in
5976      ; the fifo.
5976
5977      ;-
5978 024676 005267 157204      INC    ERRNBR           ;SET ERROR NUMBER TO 5503.
5979 024702 012701 170454      MOV    #170454,R1       ;PASS TIME-OUT VALUE OF 300 MILLI SECS.
5980 024706 016702 155330      MOV    CSRA,R2          ;PASS THE ADDRESS OF THE CSR.
5981 024712 004767 171660      JSR    PC,WAIBIS        ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
5982 024716 103136              BCC    50$              ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
5983 024720 012704 000005      MOV    #5,R4            ;PASS DELAY OF 5 MILLI SECS.
5984 024724 004767 166770      JSR    PC,DELAY         ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
5985
5986      ;+
5986      ; Transmit a null character which will cause an XOFF to be generated.
5987      ;-
5988 024730 005267 157152      INC    ERRNBR           ;SET ERROR NUMBER TO 5504.
5989 024734 010501              MOV    R5,R1            ;PASS THE LINE NUMBER.
5990 024736 012702 002704      MOV    #BUFBAS,R2       ;PASS THE START OF THE DATA PATTERN TO TX.
5991 024742 012703 000001      MOV    #1,R3            ;PASS THE NUMBER OF
5992 024746 004767 167006      JSR    PC,DODMA         ;TX A NULL CHARACTER TO CAUSE AN XOFF.
5993 024752 103120              BCC    50$              ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
5994
5995      ;+
5995      ; Wait for the XOFF to be received before TX the next 42 characters
5996      ; which will cause a further 21 XOFF's to be generated.
5997
5998      ;-
5999 024754 005267 157126      INC    ERRNBR           ;SET ERROR NUMBER TO 5505.
6000 024760 012701 170012      MOV    #170012,R1       ;PASS TIME OUT VALUE OF 10 MILLI SECS.
6001 024764 016702 155252      MOV    CSRA,R2          ;PASS THE ADDRESS OF THE CSR.
6002 024770 004767 171602      JSR    PC,WAIBIS        ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6003 024774 103107              BCC    50$              ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
6004 024776 012704 000005      MOV    #5,R4            ;PASS DELAY OF 5 MILLI SECS.
6005 025002 004767 166712      JSR    PC,DELAY         ;WAIT FOR XOFF TO GET INTO THE FIFO.
6006
6007      ;+
6007      ; Initialise the 256 byte data pattern to all nulls.
6008
6009 025006 012702 002704      MOV    #BUFBAS,R2       ;INITIALIZE THE DATA PATTERN TO BE
4$:      CLRB  (R2)+           ; ALL NULLS.
6010 025012 105022              CMP    R2,#BUFMID       ;
6011 025014 020227 003304      BLO    4$              ;
6012 025020 103774
6013
6014      ;+
6014      ; Transmit a further 31 null characters which will cause 31 XOFF's to be
6015      ; generated.
6016
6017      ;-
6018 025022 005267 157060      INC    ERRNBR           ;SET ERROR NUMBER TO 5506.
6019 025026 010501              MOV    R5,R1            ;PASS THE LINE NUMBER.
6020 025030 012702 002704      MOV    #BUFBAS,R2       ;PASS THE START OF THE DATA PATTERN TO TX.
6021 025034 012703 000037      MOV    #31.,R3         ;PASS THE LENGTH OF THE DATA PATTERN.

```

HARDWARE TEST

- FI3QLA -

```

6022 025040 004767 166714      JSR    PC,DODMA      ;TRANSMIT THE DATA PATTERN.
6023 025044 103063              BCC    50$           ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
6024                          ;+
6025                          ; Wait for the XOFF's and the null characters to be received.
6026                          ;-
6027 025046 005267 157034      INC    ERRNBR        ;SET ERROR NUMBER TO 5507.
6028 025052 012701 170454      MOV    #170454,R1    ;PASS TIME-OUT VALUE OF 300 MILLI SECS.
6029 025056 016702 155160      MOV    CSRA,R2       ;PASS THE ADDRESS OF THE CSR.
6030 025062 004767 171510      JSR    PC,WAIBIS     ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6031 025066 103052              BCC    50$           ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
6032 025070 012704 000005      MOV    #5,R4         ;PASS DELAY OF 5 MILLI SECS.
6033 025074 004767 166620      JSR    PC,DELAY      ;WAIT FOR XOFF TO GET INTO THE FIFO.
6034                          ;+
6035                          ; Read the fifo until empty, counting the number of XOFF characters
6036                          ; that are found.
6037                          ;-
6038 025100 005004              CLR    R4            ;CLEAR CHARACTER COUNTER.
6039 025102 005003              CLR    R3            ;CLEAR THE XOFF FOUND COUNTER.
6040 025104 012701 170001      MOV    #170001,R1    ;INDICATE TO TEST DATA.VALID BIT, TIME-OUT 1MS.
6041 025110 012767 012604 156770 6$: MOV    #5508,ERRNBR  ;SET UP ERROR NUMBER EACH TIME AROUND THE LOOP.
6042 025116 016702 155122      MOV    RBUFA,R2     ;INDICATE TO CHECK RECEIVE BUFFER REGISTER.
6043 025122 004767 171450      JSR    PC,WAIBIS     ;WAIT FOR RECEIVED CHAR OR TIME-OUT.
6044 025126 103032              BCC    50$           ;GO REPORT ERROR IF FIFO EMPTY.
6045 025130 005204              INC    R4            ;COUNT THE CHARACTER.
6046                          ;+
6047                          ; Check if for BMP codes in the fifo, abort the test if any are found.
6048                          ; Save the BMP code on the queue to be reported later.
6049                          ;-
6050 025132 005267 156750      INC    ERRNBR        ;SET ERROR NUMBER TO 5509.
6051 025136 004767 166244      JSR    PC,CHKBMP     ;CHECK IF WE HAVE GOT A BMP CODE.
6052 025142 103422              BCS    12$          ;GO REPORT THE ERROR IF WE FOUND A BMP CODE.
6053                          ;+
6054                          ; Check for XOFF character.
6055                          ;-
6056 025144 122702 000023      8$:   CMPB   #23,R2    ;CHECK IF THE RECEIVED CHARACTER WAS AN XOFF.
6057 025150 001001              BNE    10$          ;BRANCH IF CHARACTER WAS NOT AN XOFF.
6058 025152 005203              INC    R3            ;INCREMENT XOFF FOUND COUNT.
6059                          ;+
6060                          ; Check if all the characters including the XON have been removed.
6061                          ;-
6062 025154 020427 000400      10$:  CMP    R4,#256.   ;CHECK IF WE HAVE REMOVED ALL THE CHARACTERS.
6063 025160 002753              BLT    6$           ;GO GET THE NEXT CHAR IF WE HAVE NOT FINISHED.
6064                          ;+
6065                          ; Check if the correct number of XOFF's were found in the fifo,
6066                          ; report error if count is incorrect.
6067                          ;-
6068                          ;-
6069 025162 016767 165420 156716 MOV    5510,ERRNBR   ;SET UP THE ERROR NUMBER TO 5510.
6070 025170 022703 000040      CMP    #32.,R3      ;COMPARE EXPECTED XOFF COUNT WITH ACTUAL COUNT.
6071 025174 001411              BEQ    60$          ;EXIT TEST IF SUCCESS.
6072 025176 012767 011636 156706 MOV    #ER0503,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
6073 025204 012701 006744      MOV    #EM5402,R1   ;PASS THE MESSAGE TO BE REPORTED.
6074                          ;REPORT THE ERROR "FIFC BAD, ALARM SIGNAL DEFECTIVE".
6075 025210 104460 000402      12$:  ERROR  >>>>> ERROR <<<<<<.
6076 025212 000402              BR     60$          ;ABORT THE TEST.
6077

```

TRAP C\$ERROR

HARDWARE TEST - FI3QLA -

6078 025214 004767 170562  
6079 025220 005067 155040  
6080  
6081 025224  
025224  
025224 104401

50\$: JSR PC,TSABRT  
60\$: CLR CTRLCF  
  
ENDTST

;REPORT TEST ABORTED. ERROR # SHOWS REASON.  
;INDICATE THAT WE ARE NOT WITHIN A TEST.

L10034: TRAP C\$ETST

HARDWARE TEST

- FI3QAI -

```

6083 .SBTTL HARDWARE TEST - FI3QAI -
6084 ;+*****
6085 ;*
6086 ;* - FIFO 3/4 ALARM LEVEL ACTIVE/INACTIVE TEST
6087 ;*
6088 ;* This test verifies that the DUT's fifo 3/4 level alarm system
6089 ;* becomes active and inactive at the correct levels.
6090 ;* Any BMP code found will invalidate the test and cause it to be aborted.
6091 ;* However the BMP code will be placed on the BMP code queue, to be
6092 ;* reported later.
6093 ;* The characters are transmitted on the first available active line, in
6094 ;* internal loopback mode.
6095 ;*
6096 ;--*****
6097 025226 BGNTST
        025226
6098 000013 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6099 025226 012767 000013 155032 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (56)
6100 025234 012767 177777 155022 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6101 025242 012767 000001 156634 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
6102 025250 012767 012741 156630 MOV #5601,ERRNBR ;SET ERROR NUMBER TO 5601.
6103 025256 012767 007042 156624 MOV #EM5601,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6104 ;+
6105 ; Reset the DUT to a known state, remove the status codes from the fifo.
6106 ; Clear TX and RX interrupt enable bits in the CSR.
6107 ; This subroutine reports error >>>> 5601 <<<<.
6108 ;
6109 025264 004767 166216 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
6110 025270 103402 BCS 2$ ;SKIP EXITING TEST A SUCCESSFUL RESET.
6111 025272 000167 000412 JMP 60$ ;EXIT THIS TEST.
6112 025276
6113 2$:
6114 ;+
6115 ; Find an active line on which to perform the test.
6116 025276 004767 166632 JSR PC,FINACT ;FIND AN ACTIVE LINE.
6117 025302 103402 BCS .+6 ;SKIP EXIT OF TEST IF ACTIVE LINE FOUND.
6118 025304 000167 000400 JMP 60$ ;EXIT TEST.
6119 ;+
6120 ; Initialize the 256 byte data pattern.
6121 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
6122 ; Note: the first two characters and the last two characters will be the same.
6123 ;-
6124 025310 004767 166730 JSR PC,INDTPX ;INITIALISE THE DATA PATTERN.
6125 ;+
6126 ; Transmit a 256 character data pattern using DMA, on a single channel
6127 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
6128 ;-
6129 ;+
6130 ; Set internal loopback, enable IAUTO and receiver on the selected line.
6131 ; Transmit the first 191 characters on the first available active line.
6132 ;-
6133 025314 005267 156566 INC ERRNER ;SET ERROR NUMBER TO 5602.
6134 025320 012700 000206 MOV #206,R0 ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.
6135 025324 004767 171362 JSR PC,WTWLNCR ;INITIALISE THE LINE CONTROL REGISTER.
6136 025330 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
6137 025334 004767 171402 JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
6138 025340 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.

```



## HARDWARE TEST

- FI3QAI -

```

6139 025344 004767 166350      JSR    PC,DELAY      ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
6140 025350 010105              MOV    R1,R5        ;COPY THE LINE NUMBER.
6141 025352 012702 002704      MOV    #BUFBAS,R2   ;PASS THE START OF THE DATA PATTERN TO TX.
6142 025356 012703 000277      MOV    #191.,R3     ;PASS THE LENGTH OF THE DATA PATTERN.
6143 025362 004767 166372      JSR    PC,DODMA     ;TRANSMIT THE DATA PATTERN.
6144 025366 103146              BCC    50$          ;EXIT IF ERROR FOUND DURING DMA TX.
6145
6146                          ;+
6147                          ; Wait for DMA to complete, then wait for the last character to arrive in
6148                          ; the fifo.
6149 025370 005267 156512      :-
6150 025374 012701 170454      INC    ERRNBR       ;SET ERROR NUMBER TO 5603.
6151 025400 016702 154636      MOV    #170454,R1   ;PASS TIME-OUT VALUE OF 300 MILLI SECS.
6152 025404 004767 171166      MOV    CSRA,R2      ;PASS THE ADDRESS OF THE CSR.
6153 025410 103135              JSR    PC,WAIBIS    ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6154 025412 012704 000005      BCC    50$          ;BRANCH IF FIFO EMPTY, ABORT THE TEST.
6155 025416 004767 166276      MOV    #5,R4        ;PASS DELAY OF 5 MILLI SECS.
6156                          JSR    PC,DELAY     ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
6157
6158                          ;+
6159                          ; Transmit a null character which will cause an XOFF to be generated.
6160 025422 005267 156460      :-
6161 025426 010501              INC    ERRNBR       ;SET ERROR NUMBER TO 5604.
6162 025430 012702 002704      MOV    R5,R1        ;PASS THE LINE NUMBER.
6163 025434 012703 000001      MOV    #BUFBAS,R2   ;PASS THE START OF THE DATA PATTERN TO TX.
6164 025440 004767 166314      MOV    #1,R3        ;PASS THE NUMBER OF
6165 025444 103117              JSR    PC,DODMA     ;TX A NULL CHARACTER TO CAUSE AN XOFF.
6166                          BCC    50$          ;ABORT THE TEST IF ERROR FOUND DURING DMA TX.
6167
6168                          ;+
6169                          ; Wait for the XOFF to be received before continuing the test.
6170 025446 005267 156434      :-
6171 025452 012701 170012      INC    ERRNBR       ;SET ERROR NUMBER TO 5605.
6172 025462 004767 171110      MOV    #170012,R1   ;PASS TIME-OUT VALUE OF 10 MILLI SECS.
6173 025466 103106              MOV    CSRA,R2      ;PASS THE ADDRESS OF THE CSR.
6174 025470 012704 000005      JSR    PC,WAIBIS    ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6175 025474 004767 166220      BCC    50$          ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
6176                          MOV    #5,R4        ;PASS DELAY OF 5 MILLI SECS.
6177 025500 010577 154536      JSR    PC,DELAY     ;WAIT FOR XOFF TO GET INTO THE FIFO.
6178                          MOV    R5,@CSRA     ;SELECT THE LINE READY FOR TRANSMISSION.
6179
6180                          ;+
6181                          ; Read three characters, transmit one character until the first 192 characters
6182                          ; have been read from the fifo, ie until the half level is reached.
6183                          ; Then read the fifo until empty.
6184                          ; Count all XOFF's that are detected.
6184 025504 005005              :-
6185 025506 005004              CLR    R5           ;CLEAR THE TX FLAG.
6186 025510 012703 000300      CLR    R4           ;CLEAR THE CHARACTER COUNTER.
6187                          MOV    #192.,R3       ;SET UP READ COUNTER FOR THE FIRST 192 CHARS.
6188 025514 012700 000003      4$: MOV    #3,R0     ;SET READ COUNTER.
6189 025520 012701 170005      6$: MOV    #170005,R1 ;INDICATE TO TEST DATA.VALID BIT, TIME OUT SMS.
6190 025524 016702 154514      MOV    RBUFA,R2     ;INDICATE TO CHECK RECEIVE BUFFER REGISTER.
6191 025530 004767 171042      JSR    PC,WAIBIS    ;WAIT FOR RECEIVED CHAR OR TIME-OUT.
6192 025534 103046              BCC    14$         ;EXIT LOOP IF TIME-OUT, FIFO EMPTY.
6193 025536 005300              DEC    R0           ;DECREMENT READ COUNTER.
6194 025540 005303              DEC    R3           ;DECREMENT CHAR COUNTER.
6195 025542 003002              BGT    8$          ;SKIP DISBL'G TX IF FIRST 192 CHARS NOT READ.

```

HARDWARE TEST - FI3QAI -

```

6196 025544 052705 100000          BIS    #BIT15,R5      ;DISABLE ANY FURTHER TRANSMISSIONS.
6197
6198          ;+
6198          ; Check if the read character is a BMP code.
6199          ; If it is a BMP code save it on the queue to be reported later, and
6200          ; abort the test.
6201
6202 025550 012767 012746 156330 8$:  MOV    #5606.,ERRNBR  ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
6203 025556 004767 165624          JSR    PC,CHKBMP     ;CHECK IF CHARACTER IS A BMP CODE.
6204 025562 103446          BCS    16$          ;GO REPORT ERROR AND ABORT TEST IF BMP FOUND.
6205
6206          ;+
6206          ; Check for XOFF character. If one is found, count it.
6207          ; Transmit a null character until the first 192 chars have been read.
6208
6209 025564 122702 000023          10$:  CMPB   #23,R2      ;CHECK IF THE RECEIVED CHARACTER WAS AN XOFF.
6210 025570 001001          BNE    12$          ;BRANCH IF CHARACTER WAS NOT AN XOFF.
6211 025572 005204          INC    R4           ;INCREMENT THE XOFF CHAR FOUND COUNTER.
6212
6213 025574 005700          12$:  TST    R0           ;CHECK READ COUNT, TO SEE IF A CHAR CAN BE TX.
6214 025576 001350          BNE    6$           ;BRANCH IF 3 CHARS HAVE NOT YET BEEN READ.
6215 025600 005705          TST    R5           ;CHECK THE TRANSMISSION ENABLED FLAG.
6216 025602 100744          BMI    4$           ;SKIP TRANSMITTING A CHARACTER IF TX DISABLED.
6217 025604 012777 100000 154432  MOV    #100000,@TXCHA ;TX A NULL CHARACTER.
6218 025612 010446          MOV    R4,-(SP)     ;SAVE THE XOFF COUNT ON THE STACK.
6219
6220          ;+
6220          ; Wait for the character to be received before continuing the test.
6221          ;-
6222 025614 005267 156266          INC    ERRNBR      ;SET ERROR NUMBER TO 5607.
6223 025620 012701 170012          MOV    #170012,R1  ;PASS TIME-OUT VALUE OF 10 MILLI SECS.
6224 025624 016702 154412          MOV    CSRA,R2     ;PASS THE ADDRESS OF THE CSR.
6225 025630 004767 170742          JSR    PC,WAIBIS   ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6226 025634 103023          BCC    50$         ;IF NO TX_ACTION WAS RECEIVED, ABORT THE TEST.
6227 025636 012704 000005          MOV    #5,R4       ;PASS DELAY OF 5 MILLI SECS.
6228 025642 004767 166052          JSR    PC,DELAY    ;WAIT FOR XOFF TO GET INTO THE FIFO.
6229 025646 012604          MOV    (SP)+,R4    ;RESTORE THE XOFF COUNT.
6230 025650 000721          BR     4$          ;GO RESET THE READ COUNT AND GET NEXT CHAR.
6231
6232
6233          ;+
6233          ; Check if the correct number of XOFF's were found in the fifo
6234          ; Report error if count is incorrect.
6235
6236 025652 012767 012750 156226 14$:  MOV    #5608.,ERRNBR ;SET ERROR NUMBER TO 5608.
6237 025660 020427 000077          CMP    R4,#63      ;COMPARE THE EXPECTED AND ACTUAL XOFF COUNTS.
6238 025664 001411          BEQ    60$         ;EXIT TEST IF SUCCESS.
6239 025666 012767 011636 156216  MOV    #ER0503,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
6240 025674 012701 006744          MOV    #EM5402,R1  ;PASS THE MESSAGE TO BE REPORTED.
6241          ;REPORT THE ERROR "FIFO BAD, ALARM SIGNAL DEFECTIVE".
6242 025700          16$:  ERROR   ;>>>> ERROR <<<<<.
6243 025700 104460          TRAP   C$ERROR
6243 025702 000402          BR     60$         ;EXIT THIS TEST.
6244
6245 025704 004767 170072          50$:  JSR    PC,TSABRT   ;REPORT TEST ABORTED. ERROR # INDICATES FAULT.
6246 025710 005067 154350          60$:  CLR    CTRLCF     ;INDICATE THAT WE ARE NOT WITHIN A TEST.
6247
6248 025714          ENDTST
025714
025714 104401          L10035: TRAP   C$ETST

```

HARDWARE TEST

- FIHAVL -

```

6250 .SBTTL HARDWARE TEST - FIHAVL -
6251 ;*****
6252 ;*
6253 ;*
6254 ;* This test checks that the DUT's fifo half level alarm system
6255 ;* becomes active and inactive at the correct levels.
6256 ;* Any BMP code found will invalidate the test and cause it to be aborted.
6257 ;* However the BMP code will be placed on the BMP code queue, to be
6258 ;* reported later.
6259 ;* The characters are transmitted on the first available active line, in
6260 ;* internal loopback mode.
6261 ;*
6262 ;--*****
6263
6264 025716 BGNTST
        025716
6265 000014 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6266 025716 012767 000014 154342 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (57)
6267 025724 012767 177777 154332 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6268 025732 012767 000001 156144 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
6269 025740 012767 013105 156140 MOV #5701,ERRNBR ;SET ERROR NUMBER TO 5701.
6270 025746 012767 007110 156134 MOV #EM5701,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6271 025754 012767 011636 156130 MOV #ER0503,ERRBLK ;SELECT THE ERROR REPORTING ROUTINE.
6272
6273 ;+
6274 ; Reset the DUT to a known state, remove the status codes from the fifo.
6275 ; Clear TX and RX interrupt enable bits in the CSR.
6276 ; This subroutine reports error >>>> 5701 <<<<<.
6277 025762 004767 165520 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
6278 025766 103402 BCS 2$ ;SKIP EXITING TEST A SUCCESSFUL RESET.
6279 025770 000167 000360 JMP 60$ ;EXIT THIS TEST.
6280 025774
6281 2$:
6282 ;+
6283 ; Find an active line on which to perform the test.
6284 025774 004767 166134 JSR PC,FINACT ;FIND AN ACTIVE LINE.
6285 026000 103165 BCC 60$ ;EXIT IF NO ACTIVE LINES AVAILABLE.
6286
6287 ;+
6288 ; Initialize the 256 byte data pattern.
6289 ; Ensure the data pattern is free from XON's or XOFF's to prevent errors.
6290 ; Note: the first two characters and the last two characters will be the same.
6291 026002 004767 166236 JSR PC,INDTPX ;INITIALISE THE DATA PATTERN.
6292
6293 ;+
6294 ; Fill the fifo by transmitting 225 chars (ie 225 + 31 XOFF's).
6295 ; Transmit data pattern using DMA, on a single channel
6296 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
6297
6298 ;+
6299 ; Set internal loopback, enable IAUTO and receiver on the selected line.
6300 ; Transmit the 225 characters on the first available active line.
6301 026006 005267 156074 INC ERRNBR ;SET ERROR NUMBER TO 5702.
6302 026012 004767 167640 JSR PC,SETPAR ;SET UP PARAMETERS FOR TRANSMISSION.
6303 026016 012700 000341 MOV #225,R0 ;PASS LENGTH OF DATA PATTERN.
6304 026022 004767 170066 JSR PC,TXDATP ;TRANSMIT DATA PATTERN.
6305 026026 103150 BCC 50$ ;EXIT IF ERROR FOUND DURING TX.

```

## HARDWARE TEST

- FIHAVL -

```

6306 026030 010105          MOV    R1,R5          ;COPY THE LINE NUMBER.
6307
6308          ;+
6309          ; Wait for DMA to complete, then wait for the last character to arrive in
6310          ; the fifo.
6311 026032 005267 156050    INC    ERRNBR          ;SET ERROR NUMBER TO 5703.
6312 026036 004767 170610    JSR    PC,WAITTX      ;WAIT FOR TRANSMISSION TO COMPLETE.
6313 026042 10314?          BCC    50$            ;GO REPORT ERROR IF TX FAILED TO COMPLETE.
6314
6315          ;+
6316          ; Read the first 130 characters from the fifo, if any XON's are found
6317          ; report the error. If any BMP codes are found then save them on the queue
6318          ; and abort the test.
6319 026044 005267 156036    INC    ERRNBR          ;SET ERROR NUMBER TO 5704.
6320 026050 012700 000202    MOV    #130.,R0       ;PASS THE NUMBER OF CHARS TO READ.
6321 026054 004767 167230    JSR    PC,READBX      ;READ THE FIRST 130 CHARS FROM THE FIFO.
6322 026060 103133          BCC    50$            ;GO REPORT ERROR IF BMP CODE FOUND.
6323 026062 005267 156020    INC    ERRNBR          ;SET ERROR NUMBER TO 5705.
6324 026066 005701          TST    R1              ;CHECK IF AN XON WAS FOUND.
6325 026070 001125          BNE    40$            ;GO REPORT ERROR IF AN XON WAS FOUND.
6326
6327          ;+
6328          ; Transmit a null character.
6329          ;-
6330 026072 010577 154144    MOV    R5,@CSRA        ;SELECT THE LINE READY FOR TRANSMISSION.
6331 026076 012777 100000 154140  MOV    #100000,@TXCHA  ;TRANSMIT A NULL CHARACTER.
6332 026104 005267 155776    INC    ERRNBR          ;SET ERROR NUMBER TO 5706.
6333 026110 004767 170536    JSR    PC,WAITTX      ;WAIT FOR TX TO COMPLETE.
6334 026114 103115          BCC    50$            ;GO REPORT ERROR IF TX DID NOT COMPLETE.
6335
6336          ;+
6337          ; Read three characters, to cause the XON to be generated.
6338 026116 005267 155764    INC    ERRNBR          ;SET ERROR NUMBER TO 5707.
6339 026122 012700 000003    MOV    #3,R0           ;SET THE READ COUNT TO 3.
6340 026126 004767 167156    JSR    PC,READBX      ;READ 3 CHARACTERS FROM THE FIFO.
6341 026132 103106          BCC    50$            ;GO REPORT ERROR IF FIFO EMPTY.
6342 026134 005267 155746    INC    ERRNBR          ;SET ERROR NUMBER TO 5708.
6343 026140 005701          TST    R1              ;CHECK IF AN XON WAS FOUND.
6344 026142 001100          BNE    40$            ;GO REPORT ERROR IF AN XON WAS FOUND.
6345
6346          ;+
6347          ; Transmit 62 characters to bracket the XON and fill the fifo with 191 chars.
6348 026144 012700 000076    MOV    #62.,R0         ;PASS LENGTH OF DATA PATTERN.
6349 026150 010501          MOV    R5,R1           ;PASS THE LINE NUMBER.
6350 026152 005267 155730    INC    ERRNBR          ;SET ERROR NUMBER TO 5709.
6351 026156 004767 167732    JSR    PC,TXDATP      ;TRANSMIT DATA PATTERN.
6352 026162 103072          BCC    50$            ;EXIT IF ERROR FOUND DURING TX.
6353
6354          ;+
6355          ; Wait for DMA to complete, then wait for the last character to arrive in
6356          ; the fifo.
6357          ;-
6358 026164 005267 155716    INC    ERRNBR          ;SET ERROR NUMBER TO 5710.
6359 026170 004767 170456    JSR    PC,WAITTX      ;WAIT FOR TX TO COMPLETE.
6360 026174 103065          BCC    50$            ;GO REPORT ERROR IF TX FAILED TO COMPLETE.
6361
6362          ;+
6362          ; Read the first 126 characters.

```

HARDWARE TEST

- FIHAVL -

```

6363 ; Read the next 4 characters and check if they are in the following order
6364 ; NULL, XOFF, XON, NULL.
6365 ;-
6366 026176 005267 155704 INC ERRNBR ;SET ERROR NUMBER TO 5711.
6367 026202 012700 000176 MOV #126.,R0 ;SET UP READ COUNTER.
6368 026206 004767 167076 JSR PC,READBX ;READ THE FIRST 126 CHARS.
6369 026212 103056 BCC 50$ ;GO REPORT THE ERROR IF FIFO EMPTY.
6370 026214 005267 155666 INC ERRNBR ;SET ERROR NUMBER TO 5712.
6371 026220 005701 TST R1 ;CHECK IF AN XON WAS FOUND.
6372 026222 001050 BNE 40$ ;GO REPORT ERROR IF AN XON WAS FOUND.
6373 026224 005267 155656 INC ERRNBR ;SET ERROR NUMBER TO 5713.
6374 026230 012701 006744 MOV #EM5402,R1 ;PASS THE MESSAGE TO BE REPORTED.
6375 026234 016703 154004 MOV RBUFA,R3 ;GET THE RECEIVER BUFFER ADDRESS.
6376 026240 011302 MOV (R3),R2 ;READ THE NULL CHARACTER FROM THE FIFO.
6377 026242 120227 000000 CMPB R2,#000 ;CHECK IF IT IS A NULL CHARACTER.
6378 026246 001036 BNE 40$ ;GO REPORT THE ERROR IF NOT THE SAME.
6379 026250 005267 155632 INC ERRNBR ;SET ERROR NUMBER TO 5714.
6380 026254 011302 MOV (R3),R2 ;READ THE XOFF FROM THE FIFO.
6381 026256 120227 000023 CMPB R2,#23 ;CHECK IF THE READ CHAR IS AN XOFF.
6382 026262 001030 BNE 40$ ;GO REPORT THE ERROR IF NOT THE SAME.
6383 026264 011302 MOV (R3),R2 ;READ THE XON FROM THE FIFO.
6384 026266 005267 155614 INC ERRNBR ;SET ERROR NUMBER TO 5715.
6385 026272 120227 000021 CMPB R2,#21 ;CHECK IF THE READ CHARACTER IS AN XON.
6386 026276 001022 BNE 40$ ;GO REPORT THE ERROR IF NOT THE SAME.
6387 026300 005267 155602 INC ERRNBR ;SET ERROR NUMBER TO 5716.
6388 026304 011302 MOV (R3),R2 ;READ THE NULL CHARACTER FROM THE FIFO.
6389 026306 120227 000000 CMPB R2,#000 ;CHECK IF IT IS A NULL CHARACTER.
6390 026312 001014 BNE 40$ ;GO REPORT THE ERROR IF NOT THE SAME.
6391
6392 ;+
6393 ; Read the remaining characters from the fifo.
6394 ;-
6395 026314 012700 000075 60$: MOV #61.,R0 ;SET UP READ COUNTER.
6396 026320 005267 155562 INC ERRNBR ;SET ERROR NUMBER TO 5717.
6397 026324 004767 166760 JSR PC,READBX ;READ THE FIRST 61 CHARS.
6398 026330 103007 BCC 50$ ;GO REPORT THE ERROR IF FIFO EMPTY.
6399 026332 005267 155550 INC ERRNBR ;SET ERROR NUMBER TO 5718.
6400 026336 005701 TST R1 ;CHECK IF AN XON WAS FOUND.
6401 026340 001001 BNE 40$ ;GO REPORT ERROR IF AN XON WAS FOUND.
6402 026342 000404 BR 60$ ;EXIT THE TEST.
6403 026344 40$: ERROR ; >>>> ERROR <<<<<
6404 026346 104460 TRAP C$ERROR
6405 026346 000402 BR 60$ ;EXIT THE TEST.
6406 026350 004767 167426 50$: JSR PC,TSABRT ;REPORT TEST ABORTED. ERROR # INDICATES FAULT.
6407 026354 005067 153704 60$: CLR CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.
6408
6409 026360 ENDTST
026360
026360 104401 L10036: TRAP C$ETST

```

HARDWARE TEST

- BREAKB -

```

6411 .SBTTL HARDWARE TEST - BREAKB -
6412 ;*****
6413 ;* - BREAK generation test -
6414 ;* This test verifies that all serial transmit lines can generate a break
6415 ;* by setting the brk bit in the associated LNCTRL register.
6416 ;* Use of the internal loopback feature of the DUARTS is made to minimise
6417 ;* any external effects caused on the serial lines by this test.
6418 ;* Framing error detection is used to indicate the presence of a break,
6419 ;* by setting the appropriate bit in the RBUF register.
6420 ;--*****
6421
6422 026362 BGNTST
        026362 T13::
6423
6424 026362 012767 177777 153674 MOV # -1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6425 000015 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6426 026370 012767 000015 153670 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (64)
6427 026376 012767 000001 155500 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
6428 026404 012767 014401 155474 MOV #6401,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
6429 026412 012767 007156 155470 MOV #EM6401,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRTBL.
6430
6431 ;*
6432 ; Reset the DUT to a known state, remove the status codes from the fifo.
6433 ; Clear TX and RX interrupt enable bits in the CSR.
6434 ; This subroutine reports error >>>> 6401 <<<<<.
6435 026420 004767 165062 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
6436 026424 103165 BCC 60$ ;EXIT TEST IF FATAL ERROR FOUND.
6437
6438 ;*
6439 ; Set up Device Under Test (DUT) to:
6440 ; Disable transmission and reception interrupts.
6441 ; Delay for 10 milli-seconds to allow time to clear any breaks.
6442 026426 004767 167674 JSR PC,TXIE0 ;DISABLE TRANSMISSION INTERRUPTS.
6443 026432 004767 167046 JSR PC,RXIE0 ;DISABLE RECEPTION INTERRUPTS.
6444 026436 012705 000377 MOV #MAPLNS,R5 ;PASS ACTIVE LINE BIT MAP.
6445 026442 012700 000200 MOV #200,RU ;PASS INTERNAL LOOPBACK MODE.
6446 026446 004767 170240 JSR PC,WTWLN ;SELECT INTERNAL LOOPBACK,DISABLE DMA.
6447 026452 012704 000012 MOV #10,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
6448 026456 004767 165236 JSR PC,DELAY ;DELAY TO ALLOW ANY BREAKS TO BE CLEARED.
6449
6450 ;*
6451 ; Set up transmission an reception parameters for all lines.
6452 ; 9600 baud,8 char,1 stopbit,no parity.
6453 026462 012700 156430 MOV #156430,R0 ;SET UP BAUD RATE,ETC.
6454 026466 004767 170250 JSR PC,WTWLP ;SET COMMUNICATION PARAMETERS ON ALL LINES.
6455
6456 ;*
6457 ; Enable transmitters on all active lines.
6458 026472 016705 153536 MOV ACTLNS,R5 ;PASS ACTIVE LINE BIT MAP.
6459 026476 004767 167530 JSR PC,TXENBL ;ENABLE TRANSMISSIONS ON ALL LINES.
6460
6461 ;*
6462 ; Purge the FIFO of any unwanted characters.
6463 ; This routine reports errors with numbers >>>> 6402 thru 6404 <<<<<.
6464 026502 005267 155400 INC ERRNBR ;SET ERROR NUMBER TO 6402.
6465 026506 004767 166404 JSR PC,PUFIFR ;PURGE FIFO.
6466 026512 103132 BCC 60$ ;ABORT TEST IF FIFO WILL NOT CLEAR.

```

## HARDWARE TEST

- BREAKB -

```

6467
6468 ; Verify break generation on individual lines.
6469 ; Clear breaks on all lines.
6470 ; Delay for 10 milli-seconds to allow time for any breaks to be cleared.
6471 ; Select line, set break bit in LNCTRL register.
6472 ; Test for a character in the FIFO with frame error.
6473 ;-
6474 026514 005002 2$: CLR R2 ;CLEAR LINE COUNTER.
6475 026516 012703 000001 MOV #1,R3 ;SET UP ACTIVE LINE BIT MASK.
6476 026522 030367 153506 4$: BIT R3,ACTLNS ;CHECK IF THIS LINE IS ACTIVE.
6477 026526 001434 BEQ 8$ ;GO SELECT NEXT LINE IF THIS ONE IS INACTIVE.
6478 026530 012700 000200 MOV #200,R0 ;SET UP PARAMETER TO CLEAR BREAK BITS.
6479 026534 004767 170152 JSR PC,WTWLNLC ;CLEAR BREAK BIT,RESELECT INTERNAL LOOPBACK
6480 026540 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
6481 026544 004767 165150 JSR PC,DELAY ;DELAY TO ALLOW BREAKS TO BE CLEARED.
6482
6483 ;+
6484 ; Set break bit on selected line.
6485 ; Set up parameters to test for the frame error bit set in RBUF.
6486 ; Time-out = 5 milli seconds.
6487 ; Call routine to check for condition found.
6488 026550 010305 6$: MOV R3,R5 ;COPY ACTIVE LINE BIT MASK.
6489 026552 012700 000214 MOV #214,R0 ;SET BREAK,RESELECT LOOPBACK,ENABLE RECEPTION.
6490 026556 004767 170130 JSR PC,WTWLNLC ;SET BREAK ON SELECTED LINE.
6491
6492 ;+
6493 ; Delay for 5 ms to allow time for break to be generated and received.
6494 ; Verify reception of a character with frame error bit set.
6495 026562 012704 000005 ;-
6496 026566 004767 165126 MOV #5.,R4 ;SET DELAY VALUE TO 5 MILLI SECS.
6497 026572 017700 153446 JSR PC,DELAY ;ALLOW TIME FOR CHARACTER RECEPTION.
6498 026576 032700 020000 MOV @RBUFA,R0 ;GET CHARACTER FROM RBUF REGISTER.
6499 026602 001006 BIT #BIT13,R0 ;CHECK FOR FRAME ERROR BIT.
6500 026604 012701 007205 BNE 8$ ;SKIP ERROR REPORT IF SET.
6501 MOV #EM6402,R1 ;SELECT MESSAGE TO BE PRINTED.
6502 026610 ;REPORT ERROR "BREAK NOT RECEIVED ON LINE #nn"
026610 104455 ERRDF 6405,EM6401,ER6401 ; >>>>> ERROR #6405 <<<<<.
026612 014405 TRAP C$ERDF
026614 007156 .WORD 6405
026616 011742 .WORD EM6401
6503 026620 006303 8$: ASL R3 ;SHIFT BIT MASK FOR NEXT LINE.
6504 026622 005202 INC R2 ;NEXT LINE
6505 026624 020227 000010 CMP R2,#NUMLNS ;CHECK FOR MAX LINE COUNT.
6506 026630 001334 BNE 4$ ;IF <>,LOOP TO CHECK NEXT LINE
6507
6508 ;+
6509 ; Verify break generation on all lines simultaneously.
6510 ; Clear breaks on all lines.
6511 ; Delay for 10 milli-seconds to allow time for any breaks to be cleared.
6512 ; Purge the FIFO.
6513 ; Set break bit in LNCTRL registers on all active lines.
6514 ; Test for characters in the FIFO with frame error.
6515 026632 012705 000377 MOV #MAPLNS,R5 ;SET UP LINE TO CLEAR BREAKS ON.
6516 026636 012700 000200 MOV #200,R0 ;SET UP PARAMETER TO CLEAR BREAK BITS.
6517 026642 004767 170044 JSR PC,WTWLNLC ;CLEAR BREAK BIT,RESELECT INTERNAL LOOPBACK.
6518 026646 012704 000012 MOV #10.,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
6519 026652 004767 165042 JSR PC,DELAY ;DELAY TO ALLOW BREAKS TO BE CLEARED.

```

HARDWARE TEST - BREAKB -

```

6520
6521      ; Purge the FIFO of unwanted characters.
6522      ;
6523 026656 004767 166152      JSR    PC,PUFIFO      ;PURGE FIFO.
6524 026662 103044      BCC    50$           ;GO REPORT ERROR IF FAILED TO CLEAN_OUT FIFO.
6525
6526      ;
6527      ; Set up parameters for setting the break bit on all active lines.
6528      ; Then call routine to do it.
6529 026664 016705 153344 10$:  MOV    ACTLNS,R5      ;SET UP ACTIVE LINE BIT MASK.
6530 026670 012700 000214      MOV    #214,R0       ;SET BREAK,RESELECT LOOPBACK,ENABLE RECEPTION.
6531 026674 004767 170012      JSR    PC,WTWLCN     ;SET BREAK ON SELECTED LINES.
6532
6533      ;
6534      ; Delay for 10 milli seconds,to allow time for reception.
6535      ; Test for characters in fifo with frame error bit set.
6536 026700 012704 000012      ;
6537 026704 004767 165010      MOV    #10.,R4      ;SET DELAY VALUE TO 10 MILLI SECS.
6538 026710 010502      JSR    PC,DELAY     ;ALLOW TIME FOR CHARACTER RECEPTION.
6539 026712 004767 165424      MOV    R5,R2       ;COPY ACTIVE LINE BIT MAP.
6540 026716 017701 153322 12$:  JSR    PC,MAPCNT     ;COUNT THE NUMBER OF LINES AVAILABLE.
6541 026722 100011      MOV    @RBUFA,R1    ;GET CHARACTER FROM RBUF REGISTER.
6542 026724 032701 020000      BPL    14$         ;BRANCH IF DATA VALID NOT SET.
6543 026730 001406      BIT    #BIT13,R1    ;CHECK FOR FRAME ERROR BIT.
6544 026732 000301      BEQ    14$         ;DO NOT CLR FLG FOR THIS LINE IF FRAME BIT CLR.
6545 026734 042701 177400      SWAB   R1          ;GET LINE NUMBER IN LOW BYTE.
6546 026740 004767 165350      BIC    #177400,R1   ;CLEAR EVERYTHING BUT THE LINE NUMBER.
6547 026744 040005      JSR    PC,LINBIT    ;CALC BIT MASK FROM LINE NUMBER.
6548 026746 005302      BIC    R0,R5       ;CLEAR LINE FLAG.
6549 026750 001362 14$:  DEC    R2          ;DECREMENT THE LINE NUMBER COUNTER.
6550 026752 005705      BNE    12$         ;LOOP TO GET THE NEXT CHARACTER.
6551 026754 001411      TST    R5          ;CHECK IF ANY BREAKS NOT RECEIVED.
6552 026756 012701 007205      BEQ    60$         ;EXIT TEST IF ALL CLEAR.
6553      MOV    #EM6402,R1 ;SELECT MESSAGE TO BE PRINTED.
6554      ;REPORT ERROR "BREAK NOT RECEIVED ON LINE #nn".
        ERRDF 6406,EM6401,ER6401; >>>> ERROR #6407 <<<<<.
        TRAP  C$ERDF
        .WORD 6406
        .WORD EM6401
        .WORD ER6401
6555 026762 000402      BR     60$         ;EXIT THE TEST.
6556
6557 026774 004767 167002 50$:  JSR    PC,TSABRT    ;ABORT THE TEST.
6558 027000 005067 153260 60$:  CLR    CTRLCF      ;INDICATE THAT WE ARE NOT WITHIN A TEST.
6559 027004
        ENDTST
        L10037:
        TRAP  C$ETST
    
```



HARDWARE TEST

- NORERR -

```

6561 .SBTTL HARDWARE TEST - NORERR -
6562 ;+*****
6563 ;*
6564 ;* - NO OVERRUN ERROR TEST -
6565 ;*
6566 ;* This test verifies that the DUT will not report data overrun
6567 ;* errors when they do not occur.
6568 ;* This test puts 256 characters in the DUT FIFO plus 4 in each active
6569 ;* UART and verifies that no overrun errors are reported.
6570 ;* Any BMP code found will invalidate the test and cause it to be aborted.
6571 ;* However the BMP code will be placed on the BMP code queue, to be
6572 ;* reported later.
6573 ;*
6574 ; *****
6575 027006 BGNTST
027006
6576 000016 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6577 027006 012767 000016 153252 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (66)
6578 027014 012767 177777 153242 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6579 027022 012767 000001 155054 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
6580 027030 012767 014711 155050 MOV #6601,ERRNBR ;SET ERROR NUMBER TO 6601.
6581 027036 012767 007246 155044 MOV #EM6601,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6582 ;+
6583 ; Reset the DUT to a known state, remove the status codes from the fifo
6584 ; Clear TX and RX interrupt enable bits in the CSR.
6585 ; This subroutine reports error >>>> 6601 <<<<.
6586 ;-
6587 027044 004767 164436 JSR PC,CLNRST ;RESET THE DHV11 M, REPORT ANY ERRORS FOUND.
6588 027050 103402 BCS .+6 ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
6589 027052 000167 000422 JMP 60$ ;EXIT THE TEST, FATAL ERROR WAS FOUND.
6590 ;+
6591 ; Find an active line on which to perform the test.
6592 ; Initialize the 256 byte data pattern.
6593 ;
6594 027056 004767 165052 JSR PC,FINACT ;FIND AN ACTIVE LINE.
6595 027062 103402 BCS .+6 ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
6596 027064 000167 000410 JMP 60$ ;EXIT THE TEST, FATAL ERROR WAS FOUND.
6597 027070 004767 165120 JSR PC,INDATP ;INITIALISE DATA PATTERN.
6598 ;+
6599 ; Transmit a 265 character data pattern using DMA, on a single channel
6600 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
6601 ;-
6602 ;+
6603 ; Set internal loopback on the selected line.
6604 ; Transmit the data pattern on the first available active line.
6605 ;-
6606 027074 005267 155006 INC ERRNBR ;SET THE ERROR REPORT NUMBER TO 6602.
6607 027100 012700 000204 MOV #204,R0 ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE RX.
6608 027104 004767 167602 JSR PC,WTWLNCR ;INITIALISE THE LINE CONTROL REGISTER.
6609 027110 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
6610 027114 004767 167622 JSR PC,WTWLP ;SET THE LPR CONTENTS TO 38.4K BAUD.
6611 027120 012704 000012 MOV #10,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
6612 027124 004767 164570 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
6613 027130 012702 002704 MOV #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
6614 027134 012703 000400 MOV #BUFMID-BUFBAS,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
6615 027140 004767 164614 JSR PC,DODMA ;TRANSMIT THE DATA PATTERN.
6616 027144 103153 BCC 50$ ;EXIT IF ERROR FOUND DURING DMA TX.

```

HARDWARE TEST

- NCRERR -

```

6617
6618
6619
6620
6621 027146 005267 154734
6622 027152 012701 170536
6623 027156 016702 153060
6624 027162 004767 167410
6625 027166 103142
6626 027170 012704 000005
6627 027174 004767 164520
6628
6629
6630
6631
6632 027200 016705 153030
6633 027204 012700 000204
6634 027210 004767 167476
6635 027214 012700 177670
6636 027220 004767 167516
6637 027224 012704 000012
6638 027230 004767 164464
6639
6640 027234 012702 002704
6641 027240 012703 000004
6642 027244 0050C1
6643 027246 005267 154634
6644 027252 010100
6645 027254 006300
6646 027256 036067 002370 152750
6647 027264 001403
6648 027266 004767 164466
6649 027272 103100
6650 027274 005201
6651 027276 020127 000010
6652 027302 002763
6653
6654 027304 005267 154576
6655 027310 012701 170040
6656 027314 016702 152722
6657 027320 004767 167252
6658 027324 103063
6659 027326 012704 000005
6660 027332 004767 164362
6661
6662
6663
6664
6665 027336 016702 152672
6666 027342 004767 164774
6667 027346 006302
6668 027350 006302
6669 027352 012705 000400
6670 027356 060205
6671 027360 005004
6672 027362 012767 014716 154516 6$:
6673 027370 017702 152650

```

```

;+
; Wait for DMA to complete, then wait for the last character to arrive in
; the fifo.
;-
      INC      ERRNBR      ;SET ERROR NUMBER TO 503.
      MOV      #170536,R1  ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
      MOV      CSRA,R2    ;PASS THE ADDRESS OF THE CSR.
      JSR      PC,WAIBIS   ;WAIT FOR DMA TO COMPLETE, TX ACTION SET.
      BCC      50$        ;ABORT THE TEST IF TIME-OUT ON DMA COMPLETION.
      MOV      #5,R4      ;PASS DELAY OF 5 MILLI SECS.
      JSR      PC,DELAY    ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.

;+
; Transmit 4 characters on each active line.
;-
      MOV      ACTLNS,R5   ;ALTER PARAMETERS FOR ALL ACTIVE LINES.
      MOV      #204,R0    ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE RX.
      JSR      PC,WTWLNC  ;INITIALISE THE LINE CONTROL REGISTER.
      MOV      #177670,R0 ;PASS THE LPR CONTENTS.
      JSR      PC,WTWLPR  ;SET THE LPR CONTENTS TO 38.4K BAUD.
      MOV      #10,R4     ;PASS DELAY TIME OF 10 MILLI SECONDS.
      JSR      PC,DELAY   ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.

      MOV      #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
      MOV      #4,R3      ;PASS THE LENGTH OF THE DATA PATTERN.
      CLR      R1         ;CLEAR THE LINE COUNTER.
      INC      ERRNBR     ;SET ERROR NUMBER TO 6604.
2$:   MOV      R1,R0
      ASL      R0         ;CALCULATE THE LINE OFFSET FROM THE LINE #.
      BIT      BITTBL(R0),ACTLNS ;TEST FOR THIS LINE BEING ACTIVE.
      BEQ      4$        ;SKIP THE TX ON THIS LINE IF IT IS NOT ACTIVE.
      JSR      PC,DODMA   ;TRANSMIT THE 5 CHAR DATA PATTERN.
      BCC      50$        ;ABORT IF ERROR FOUND DURING DMA TX.
4$:   INC      R1         ;INCREMENT THE LINE COUNTER.
      CMP      R1,#NUMLNS ;TEST FOR ALL POSSIBLE LINES HANDLED
      BLT      2$        ;LOOP IF NOT ALL LINES HANDLED.

      INC      ERRNBR     ;SET ERROR NUMBER TO 6605.
      MOV      #170040,R1 ;PASS TIME-OUT VALUE OF 32 MILLI SECS.
      MOV      CSRA,R2    ;PASS THE ADDRESS OF THE CSR.
      JSR      PC,WAIBIS   ;WAIT FOR A DMA TO COMPLETE, TX ACTION SET.
      BCC      50$        ;ABORT THE TEST IF TIME OUT ON DMA COMPLETION.
      MOV      #5,R4      ;PASS DELAY OF 5 MILLI SECS.
      JSR      PC,DELAY    ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.

;+
; Read the FIFO checking for overrun errors. Report errors if found.
; Abort the test if a BMP code was found in the fifo.
;-
      MOV      ACTLNS,R2  ;GET THE NUMBER OF ACTIVE LINES.
      JSR      PC,MAPCNT
      ASL      R2
      ASL      R2        ;MULTIPLY NUMBER OF ACTIVE LINES BY 4.
      MOV      #256.,R5
      ADD      R2,R5     ;CALCULATE NUMBER OF CHARACTERS TO RX.
      CLR      R4        ;CLEAR THE CHARACTER COUNTER.
6$:   MOV      #6606.,ERRNBR ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
      MOV      @RBUFA,R2 ;READ A CHARACTER FROM THE FIFO.

```

## HARDWARE TEST

- NORERR -

```

6674 027374 100032          BPL      10$          ;EXIT THE READ LOOP IF THE FIFO IS EMPTY.
6675
6676          ;+
6677          ; Check if the read character is a BMP code.
6678          ; If it is a BMP code save it on the queue to be reported later, and
6679          ; abort the test.
6680 027376 004767 164004    JSR      PC,CHKBMP      ;CHECK IF CHARACTER IS A BMP CODE.
6681 027402 103002          BCC      8$          ;BRANCH IF NOT A BMP CODE.
6682 027404          ERROR          ;
6683 027406 000434          BR       60$          ;EXIT THIS TEST.
6684
6685 027410 005267 154472    8$:      INC      ERRNBR      ;SET ERROR NUMBER TO 6607.
6686 027414 005204          INC      R4          ;COUNT THIS CHARACTER.
6687 027416 020405          CMP      R4,R5        ;COMPARE # OF CHARS WITH MAX # OF CHARS.
6688 027420 003025          BGT      50$          ;ABORT TEST IF TOO MANY VALID CHARS READ.
6689 027422 032702 040000    BIT      #BIT14,R2     ;TEST THE OVERRUN BIT OF THE READ CHAR.
6690 027426 001755          BEQ      6$          ;LOOP TO READ THE NEXT CHAR IF NO ERROR.
6691 027430 005267 154452    INC      ERRNBR      ;SET ERROR NUMBER TO 6608.
6692 027434 012767 012040 154450  MOV      #ER7801,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
6693 027442 012701 007274    MOV      #EM6602,R1    ;PASS THE MESSAGE TO BE REPORTED.
6694 027446 010203          MOV      R2,R3
6695 027450 000303          SWAB    R3
6696 027452 042703 177760    BIC      #177760,R3    ;GET FAILING LINE NUMBER.
6697          ;REPORT "OVERRUN ERROR REPORTED WHEN NONE FORCED, ON LINE nn ..."
6698 027456          ERROR          ;
6699 027456 104460          BR       6$          ;LOOP TO READ THE NEXT CHAR.
6700 027460 000740
6701 027462 012767 014721 154416 10$:    MOV      #6609.,ERRNBR ;SET ERROR NUMBER TO 6609.
6702 027470 020405          CMP      R4,R5        ;COMPARE NUMBER OF CHARS READ WITH EXPECTED.
6703 027472 001402          BEQ      60$          ;EXIT TEST WITHOUT ABORT IF CORRECT # OF CHARS.
6704
6705 027474 004767 166302    50$:    JSR      PC,TSABRT    ;ABORT THE TEST, NON-RELATED TEST ERROR FOUND.
6706 027500 005067 152560    60$:    CLR      CTRLCF      ;INDICATE THAT WE ARE NOT WITHIN A TEST.
6707 027504          ENDTST
                                L10040:
                                TRAP    C$ETST

```

HARDWARE TEST

- ORERR -

```

6709 .SBTTL HARDWARE TEST - ORERR -
6710 ;*****
6711 ;
6712 ; - OVERRUN ERROR TEST -
6713 ;
6714 ; This test verifies that the DUT will report data overrun errors when
6715 ; they occur.
6716 ; This test puts 256 characters in the DUT FIFO plus 5 in each active
6717 ; UART and verifies that overrun errors are reported on all active lines.
6718 ; Any BMP code found will invalidate the test and cause it to be aborted.
6719 ; However the BMP code will be placed on the BMP code queue, to be
6720 ; reported later.
6721 ;
6722 ;-----*****
6723 027506 BGNTST
        027506
6724 000017 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6725 027506 012767 000017 152552 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (67)
6726 027514 012767 177777 152542 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6727 027522 012767 000001 154354 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
6728 027530 012767 015055 154350 MOV #6701,ERRNBR ;SET ERROR NUMBER TO 6701.
6729 027536 012767 007346 154344 MOV #EM6701,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6730 ;
6731 ; Reset the DUT to a known state, remove the status codes from the fifo.
6732 ; Clear TX and RX interrupt enable bits in the CSR.
6733 ; This subroutine reports error >>>> 6701 <<<<.
6734 ;
6735 027544 004767 163736 JSR PC,CLNRST ;RESET THE DHV11-M, REPORT ANY ERRORS FOUND.
6736 027550 103402 BCS .+6 ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
6737 027552 000167 000650 JMP 60$ ;EXIT THE TEST, FATAL ERROR WAS FOUND.
6738 ;
6739 ; Find an active line on which to perform the test.
6740 ; Initialize the 256 byte data pattern.
6741 ;
6742 027556 004767 164352 JSR PC,FINACT ;FIND AN ACTIVE LINE.
6743 027562 103402 BCS .+6 ;IF ACTIVE LINE IS FOUND, DON'T ABORT TEST.
6744 027564 000167 000636 JMP 60$ ;ABORT THE TEST, NO ACTIVE LINES WERE FOUND.
6745 027570 004767 164420 JSR PC,INDATP ;INITIALISE DATA PATTERN.
6746 ;
6747 ; Transmit a 265 character data pattern using DMA, on a single channel
6748 ; at 38.4k baud, 8 bits per character, odd parity, 2 stop bits.
6749 ;
6750 ;
6751 ; Set internal loopback on the selected line.
6752 ; Transmit the data pattern on the first available active line.
6753 ;
6754 027574 005267 154306 INC ERRNBR ;SET ERROR NUMBER TO 6702.
6755 027600 012700 000204 MOV #204,R0 ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE RX.
6756 027604 004767 167102 JSR PC,WTWLNCR ;INITILAISE THE LINE CONTROL REGISTER.
6757 027610 012700 177670 MOV #177670,R0 ;PASS THE LPR CONTENTS.
6758 027614 004767 167122 JSR PC,WTWLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
6759 027620 012704 000012 MOV #10,R4 ;PASS DELAY TIME OF 10 MILLI SECONDS.
6760 027624 004767 164070 JSR PC,DELAY ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
6761 027630 012702 002704 MOV #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
6762 027634 012703 000400 MOV #BUFMID-BUFBAS,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
6763 027640 004767 164114 JSR PC,DODMA ;TRANSMIT THE DATA PATTERN.
6764 027644 103402 BCS .+6 ;IF NO ERROR FOUND DURING DMA TX, DON'T ABORT.

```

HARDWARE TEST

- ORERR -

```

6765 027646 000167 000550          JMP      50$          ;ABORT TEST, ERROR FOUND DURING DMA TX.
6766                                     ;+
6767                                     ; Wait for DMA to complete, then wait for the last character to arrive in
6768                                     ; the fifo.
6769                                     ;-
6770 027652 005267 154230          INC      ERRNBR       ;SET ERROR NUMBER TO 6703.
6771 027656 012701 170536          MOV      #170536,R1  ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
6772 027662 016702 152354          MOV      CSRA,R2     ;PASS THE ADDRESS OF THE CSR.
6773 027666 004767 166704          JSR      PC,WAIBIS   ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
6774 027672 103402                   BCS      .+6         ;IF NO TIME-OUT ON DMA COMPLETION, DON'T ABORT.
6775 027674 000167 000522          JMP      50$         ;ABORT TEST, TIME-OUT ON DMA COMPLETION.
6776 027700 012704 000005          MOV      #5,R4      ;PASS DELAY OF 5 MILLI SECS.
6777 027704 004767 164010          JSR      PC,DELAY    ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
6778                                     ;+
6779                                     ; Transmit 5 characters on each active line.
6780                                     ;
6781 027710 016705 152320          MOV      ACTLNS,R5   ;ALTER PARAMETERS FOR ALL ACTIVE LINES.
6782 027714 012700 000204          MOV      #204,R0    ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE RX.
6783 027720 004767 166766          JSR      PC,WTWLNCR ;INITILAISE THE LINE CONTROL REGISTER.
6784 027724 012700 177670          MOV      #177670,P0 ;PASS THE LPR CONTENTS.
6785 027730 004767 167006          JSR      PC,WTWLPRL ;SET THE LPR CONTENTS TO 38.4K BAUD.
6786 027734 012704 000012          MOV      #10.,R4   ;PASS DELAY TIME OF 10 MILLI SECONDS.
6787 027740 004767 163754          JSR      PC,DELAY    ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.
6788
6789 027744 012702 002704          MOV      #BUFBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
6790 027750 012703 000005          MOV      #5,R3     ;PASS THE LENGTH OF THE DATA PATTERN.
6791 027754 005001                   CLR      R1         ;CLEAR THE LINE COUNTER.
6792 027756 005267 154124          INC      ERRNBR     ;SET ERROR NUMBER TO 6704.
6793 027762 010100                   2$: MOV      R1,R0
6794 027764 006300                   ASL      R0         ;CALCULATE LINE OFFSET FROM THE LINE #.
6795 027766 036067 002370 152240    BIT      BITTBL(R0),ACTLNS ;TEST FOR THIS LINE BEING ACTIVE.
6796 027774 001405                   BEQ      4$         ;SKIP THE TX ON THIS LINE IF IT IS NOT ACTIVE.
6797 027776 004767 163756          JSR      PC,DODMA   ;TRANSMIT THE 5 CHAR DATA PATTERN.
6798 030002 103402                   BCS      .+6         ;IF NO TIME-OUT ON DMA COMPLETION, DON'T ABORT.
6799 030004 000167 000412          JMP      50$         ;ABORT TEST, TIME-OUT ON DMA COMPLETION.
6800 030010                   4$: INC      R1         ;INCREMENT THE LINE NUMBER COUNTER.
6801 030012 020127 000010          CMP      R1,#NUMLNS ;TEST FOR ALL POSSIBLE LINES HANDLED
6802 030016 002761                   BLT      2$         ;LOOP IF NOT ALL LINES HANDLED.
6803
6804 030020 005267 154062          INC      ERRNBR     ;SET ERROR NUMBER TO 6705.
6805 030024 012701 170040          MOV      #170040,R1 ;PASS TIME-OUT VALUE OF 32 MILLI SECS.
6806 030030 016702 152206          MOV      CSRA,R2   ;PASS THE ADDRESS OF THE CSR.
6807 030034 004767 166536          JSR      PC,WAIBIS  ;WAIT FOR A DMA TO COMPLETE, TX_ACTION SET.
6808 030040 103170                   BCC      50$        ;ABORT THE TEST IF TIME-OUT ON DMA COMPLETION.
6809 030042 012704 000005          MOV      #5,R4     ;PASS DELAY OF 5 MILLI SECS.
6810 030046 004767 163646          JSR      PC,DELAY   ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
6811                                     ;+
6812                                     ; Read 256 chars from the FIFO checking for BMP codes.
6813                                     ; Abort the test if a BMP code was found in the fifo
6814                                     ;-
6815 030052 012704 000400                   MOV      #256.,R4   ;SET UP THE CHARACTER COUNTER.
6816 030056 012767 015062 154022 6$: MOV      #6706.,ERRNBR ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
6817 030064 017702 152154                   MOV      @RBUFA,R2  ;READ A CHARACTER FROM THE FIFO.
6818 030070 100154                   BPL      50$        ;ABORT THE TEST IF DATA.VALID IS CLEAR.
6819 030072 005267 154010          INC      ERRNBR     ;SET ERROR NUMBER TO 6707.
6820 030076 004767 163304          JSR      PC,CHKBMP  ;CHECK IF CHARACTER IS A BMP CODE.
6821 030102 103545                   BCS      24$        ;REPORT ERROR AND ABORT TEST IF A BMP CODE.

```

HARDWARE TEST

- ORERR

```

6822 030104 005304          DEC    R4          ;COUNT THIS CHARACTER.
6823 030106 001363          BNE    6$          ;LOOP IF NOT 256 CHARS READ FROM FIFO.
6824                                     ;+
6825                                     ; Read the remaining and verif 1 overrun plus 1 char from each line.
6826                                     ;-
6827 030110 005004          CLR    R4          ;CLEAR THE OVERRUN ERROR FLAGS.
6828 030112 012700 003744    MOV    #RXCNTB,R0
6829 030116 004767 163406    JSR    PC,CLR16W   ;CLEAR RX CHAR COUNT TABLE.
6830 030122 012767 015064 153756 8$:  MOV    #6708.,ERRNBR ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
6831 030130 017702 152110    MOV    @RBUFA,R2   ;READ A CHARACTER FROM THE FIFO.
6832 030134 100047          BPL    14$         ;GO ANALYZE THE RESULTS IF ALL CHARS READ.
6833 030136 004767 163244    JSR    PC,CHKBMP   ;CHECK IF CHAR IS A BMP CODE.
6834 030142 103525          BCS    24$         ;REPORT ERROR AND ABORT TEST IF A BMP CODE.
6835 030144 005267 153736    INC    ERRNBR      ;SET ERROR NUMBER TO 6709.
6836 030150 010200          MOV    R2,R0
6837 030152 000300          SWAB   R0
6838 030154 042700 177760    BIC    #177760,R0  ;CALCULATE THE LINE NUMBER OF THE CHAR.
6839 030160 006300          ASL    R0          ;FORM WORD TABLE OFFSET FOR TABLE ACCESS.
6840 030162 042702 007400    BIC    #7400,R2    ;REMOVE LINE NUMBER FROM THE READ CHAR.
6841 030166 036067 002370 152040    BIT    BITTBL(R0),ACTLNS ;TEST FOR ACTIVE LINE.
6842 030174 001512          BEQ    50$         ;ABORT TEST IF FOR INACTIVE LINE.
6843 030176 005267 153704    INC    ERRNBR      ;SET ERROR NUMBER TO 6710.
6844 030202 005760 003744    TST    RXCNTB(R0) ;CHECK THE RX CHAR COUNTER FOR THIS LINE.
6845 030206 001006          BNE    10$         ;IS THIS FIRST CHAR ON LINE?
6846 030210 020227 140000    CMP    R2,#140000 ;YES, TEST FOR NULL CHAR WITH OVERRUN.
6847 030214 001414          BEQ    12$         ;IS CHAR A NULL?
6848 030216 056004 002370    BIS    BITTBL(R0),R4 ;NO, SET THE OVERRUN BIT ERROR FLAG FOR LINE.
6849 030222 000411          BR     12$         ;GO COUNT THE CHAR AND CONTINUE.
6850 030224 026027 003744 000004 10$:  CMP    RXCNTB(R0),#4
6851 030232 002073          BGE    50$         ;5TH CHAR ON THIS LINE? YES, ABORT.
6852 030234 032702 040000    BIT    #BIT14,R2  ;NO, CHECK OVERRUN BIT.
6853 030240 001402          BEQ    12$         ;IS OVERRUN BIT CLEAR? YES, GO COUNT CHAR.
6854 030242 056004 002370    BIS    BITTBL(R0),R4 ;NO, SET THE OVERRUN BIT ERROR FLAG FOR LINE.
6855 030246 005260 003744 12$:  INC    RXCNTB(R0)  ;COUNT THIS CHARACTER.
6856 030252 000723          BR     8$          ;LOOP UNTIL ALL CHARS ARE READ FROM FIFO.
6857                                     ;+
6858                                     ; Test for abort conditions. Only none abort conditions are:
6859                                     ;   1) 2 chars RXed on a line and no overrun error bit failure detected.
6860                                     ;   2) 2 to 4 chars RXed on a line and an overrun bit failure detected.
6861                                     ;-
6862 030254 005001          14$:  CLR    R1          ;INITIALIZE LINE LOOP, CLEAR LINE OFFSET.
6863 030256 012767 015067 153622 16$:  MOV    #6711.,ERRNBR ;SET UP ERROR NUMBER EACH TIME AROUND LOOP.
6864 030264 036167 002370 151742    BIT    BITTBL(R1),ACTLNS
6865 030272 001415          BEQ    18$         ;LINE ACTIVE? NO, NEXT LINE.
6866 030274 026127 003744 000002    CMP    RXCNTB(R1),#2 ;YES.
6867 030302 002447          BLT    50$         ;FEWER THAN 2 CHARS RXED? YES, ABORT.
6868 030304 036104 002370    BIT    BITTBL(R1),R4 ;NO.
6869 030310 001006          BNE    18$         ;OVERRUN BIT ERROR FLAG SET? YES, NEXT LINE.
6870 030312 005267 153570    INC    ERRNBR      ;SET LINE NUMBER TO 6712.
6871 030316 026127 003744 000002    CMP    RXCNTB(R1),#2
6872 030324 001036          BNE    50$         ;NOT 2 CHARS RXED? YES, ABORT. NO, NEXT LINE.
6873 030326 062701 000002 18$:  ADD    #2,R1       ;SET LINE OFFSET TO THE NEXT LINE.
6874 030332 020127 000020    CMP    R1,#NUMLNS*2
6875 030336 002747          BLT    16$         ;ALL LINES DONE? NO, LOOP. YES, CONTINUE.
6876                                     ;+
6877                                     ; Check for overrun error bit failures, print error message if found.
6878                                     ;-

```



HARDWARE TEST

- DTRMCS -

```

6905 .SBTTL HARDWARE TEST - DTRMCS -
6906 ;*****
6907 ;*
6908 ;* - Data Terminal Ready Modem Control Signal Test
6909 ;*
6910 ;* This test verifies that the DTR modem control signal is working
6911 ;* correctly. It will only be performed if either 25 pin or staggered
6912 ;* loopback is specified. This test uses the looped back signals RI
6913 ;* and DSR to test the DTR signal. This test is performed on all
6914 ;* active lines.
6915 ;*
6916 ;-----*****
6917 030434 BGNTST
        030434 T16::
6918 ;+
6919 ; Only perform this test if the DUT is in external or staggered loopback mode.
6920 ;
6921 030434 032767 000002 151574 BIT #BIT1,LOPBACK ;CHECK TYPE OF LOOPBACK MODE SELECTED.
6922 030442 001002 BNE 2$
6923 030444 000167 000456 JMP 60$ ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
6924 030450 000020 2$: TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
6925 030450 012767 000020 151610 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (78)
6926 030456 012767 177777 151600 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
6927 030464 012767 000001 153412 MOV #1,ERRTYP ;SET ERROR TYPE IN ERROR TABLE.
6928 030472 012767 017171 153406 MOV #7801.,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
6929 030500 012767 007446 153402 MOV #EM7801,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6930 ;+
6931 ; Reset the DUT to a known state, remove status codes from the fifo.
6932 ; Clear TX and RX interrupt enable bits.
6933 ; This subroutine reports error >>>> 7801 <<<<.
6934 ;-
6935 030506 004767 162774 JSR PC,CLNRST ;RESET THE DUT.
6936 030512 103402 BCS 4$
6937 030514 000167 000406 JMP 60$ ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
6938 ;+
6939 ; Set up the TX/RX associated line number table.
6940 ;-
6941 030520 004767 162326 4$: JSR PC,ASLNTL ;SET UP THE ASSOCIATED LINE TABLES.
6942 ;+
6943 ; Set up a loop which handles one line per iteration.
6944 ; This loop clears all the DTRs and then sets them individually and waits for
6945 ; a response on the associated RI and DSR signals.
6946 ; This loop will clear the TX.IE and RX.IE bits if they are set.
6947 ;-
6948 030524 005003 CLR R3 ;CLEAR THE LINE COUNTER.
6949 030526 010300 6$: MOV R3,R0
6950 030530 006300 ASL R0
6951 030532 036067 002370 151474 BIT BITTBL(R0),ACTLNS
6952 030540 001465 BEQ 12$ ;DON'T TEST IF NOT ACTIVE LINE.
6953 ;+
6954 ; Clear all the DUT LNCTRL registers DTR bits.
6955 ;-
6956 030542 005000 CLR R0 ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
6957 030544 012705 000377 MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
6958 030550 004767 166136 JSR PC,WTLNC ;CLEAR ALL THE DUT DTR BITS.
6959 030554 012704 000074 MOV #60.,R4
6960 030560 004767 163134 JSR PC,DELAY ;DELAY FOR 60 MS TO ALLOW SIGNALS TO SETTLE.

```



HARDWARE TEST

- DTRMCS -

```

6961
6962 ; Check that at least one of associated DSR or RI is clear and record states.
6963 ;-
6964 030564 116304 004044      MOV      TXRLNB(R3),R4      ;GET THE ASSOCIATED LINE NUMBER.
6965 030570 010477 151446      MOV      R4,@CSRA          ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
6966 030574 017705 151450      MOV      @STATA,R5         ;GET THE STATE OF THE ASSOCIATED DSR, RI BITS.
6967 030600 012700 120000      MOV      #BIT15:BIT13,R0
6968 030604 040500              BIC      R5,R0             ;CHECK FOR BOTH DSR AND RI SET.
6969 030606 001431              BEQ      10$              ;GO REPORT DTR IS BAD IF BOTH ARE SET.
6970
6971 ; Set the DTR for the selected line and wait for either DSR or RI to set.
6972 ;-
6973 030610 010377 151426      MOV      R3,@CSRA          ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.
6974 030614 052777 001000 151430  BIS      #BIT9,@LNCTRA     ;SET THE SELECTED LINE DTR.
6975 030622 012701 150074      MOV      #150074,R1        ;SPECIFY TO WAIT UP TO 60 MS FOR RI TO SET.
6976 030626 032705 100000      BIT      #BIT15,R5         ;CHECK PREVIOUS STATE OF DSR BIT.
6977 030632 001002              BNE      8$              ;GO USE RI IF DSR BIT WAS NOT CLEAR.
6978 030634 012701 170074      MOV      #170074,R1        ;SPECIFY TO WAIT UP TO 60 MS FOR DSR SET.
6979 030640 016702 151404 8$:  MOV      STATA,R2          ;SPECIFY TO LOOK IN STAT REG FOR BIT TO SET.
6980 030644 010477 151372      MOV      R4,@CSRA          ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
6981 030650 004767 165722      JSR      PC,WAIBIS         ;WAIT UP TO 60 MS FOR SIGNAL TO GO SET.
6982 030654 103417              BCS      12$              ;SELECT NEXT LINE AND LOOP IF SIGNAL IS SET.
6983 030656 017700 151366      MOV      @STATA,R0         ;GET THE STATUS REGISTER CONTENTS.
6984 030662 042700 057777      BIC      #57777,R0         ;REMOVE ALL BUT THE DSR AND RI BITS.
6985 030666 040500              BIC      R5,R0            ;TEST FOR SIGNAL ONCE CLEAR, BUT NOW SET.
6986 030670 001011              BNE      12$              ;GO LOOP IF SIGNAL HAS GONE FROM CLR TO SET.
6987 030672              10$: ;Report DTR MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
6988 030672 012767 017172 153206  MOV      #7802.,ERRNBR     ;SELECT THE ERROR NUMBER.
6989 030700 012767 012040 153204  MOV      #ER7801,ERRBLK    ;SELECT THE ERROR PRINT ROUTINE.
6990 030706 012701 007501      MOV      #EM7802,R1        ;SELECT THE ERROR MESSAGE.
6991 030712              ERROR
6992 030712 104460              TRAP      C$ERROR
6993 030714 005203              12$:  INC      R3              ;SELECT THE NEXT LINE NUMBER.
6994 030716 020327 000010      CMP      R3,#NUMLNS        ;TEST FOR ALL LINES DONE.
6995 030722 002701              BLT      6$              ;LOOP IF NOT ALL LINES DONE.
6996
6997 ; Set up a loop which handles one line per iteration.
6998 ; This loop sets all the DTRs and then clears them individually and waits for
6999 ; a response on the associated RI and DSR signals.
7000 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7001 ;-
7002 030724 005003              CLR      R3              ;CLEAR THE LINE COUNTER.
7003 030726 010300 14$:  MOV      R3,R0
7004 030730 006300              ASL      R0
7005 030732 036067 002370 151274  BIT      BITTBL(R0),ACTLNS
7006 030740 001466              BEQ      20$              ;DON'T TEST IF NOT ACTIVE LINE.
7007
7008 ; Set all the DUT LNCTRL registers DTR bits.
7009 ;-
7010 030742 012700 001000      MOV      #BIT9,R0          ;SPECIFY THAT DTR BITS ARE TO BE SET.
7011 030746 012705 000377      MOV      #MAPLNS,R5        ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7012 030752 004767 165734      JSR      PC,WTWLNLC        ;SET ALL THE DUT DTR BITS.
7013 030756 012704 000074      MOV      #60.,R4
7014 030762 004767 162732      JSR      PC,DELAY          ;DELAY FOR 60 MS TO ALLOW SIGNALS TO SETTLE.
7015
7016 ; Check that at least one of associated DSR or RI is set and record states.

```



HARDWARE TEST

- RTSMCS -

```

7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064
7065 031134
      031134
7066
7067
7068
7069 031134 032767 000002 151074
7070 031142 001002
7071 031144 000167 000456
7072 031150 000021
7073 031150 012767 000021 151110
7074 031156 012767 177777 151100
7075 031164 012767 000001 152712
7076 031172 012767 017335 152706
7077 031200 012767 007532 152702
7078
7079
7080
7081
7082
7083 031206 004767 162274
7084 031212 103402
7085 031214 000167 000406
7086
7087
7088
7089 031220 004767 161626
7090
7091
7092
7093
7094
7095
7096 031224 005003
7097 031226 010300
7098 031230 00630C
7099 031232 036067 002370 150774
7100 031240 001465
7101
7102
7103
7104 031242 005000
7105 031244 012705 000377
7106 031250 004767 165436
7107 031254 012704 000074
7108 031260 004767 162434

```

```

.SBTTL HARDWARE TEST - RTSMCS -
;*****
; - Request to Send Modem Control Signal Test -
;
; This test verifies that the RTS modem control signal is working
; correctly. It will only be performed if either 25 pin or staggered
; loopback is specified. This test uses the looped back signals CTS
; and DCD to test the RTS signal. This test is performed on all
; active lines.
;
;-----*****
      BGNTST
                                          T17::
;+
; Only perform this test if the DUT is in external or staggered loopback mode.
;-
      BIT    #BIT1,LOPBCK    ;CHECK TYPE OF LOOPBACK MODE SELECTED.
      BNE    1$
      JMP    60$
1$:    TNUM  == TNUM + 1    ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV    #TNUM,TSTNUM   ;SET UP THE TEST NUMBER. (79)
      MOV    #-1,CTRLCF     ;INDICATE THAT WE ARE IN A TEST.
      MOV    #1,ERRTYP      ;SET ERROR TYPE IN ERROR TABLE.
      MOV    #7901,ERRNBR   ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV    #EM7901,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;+
; Reset the DUT to a known state, remove status codes from the fifo.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 7901 <<<<.
;-
      JSR    PC,CLNRST      ;RESET THE DUT.
      BCS    3$
      JMP    60$           ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;+
; Set up the TX/RX associated line number table.
;-
3$:    JSR    PC,ASLNTL     ;SET UP THE ASSOCIATED LINE TABLES.
;+
; Set up a loop which handles one line per iteration.
; This loop clears all the RTSs and then sets them individually and waits for
; a response on the associated CTS and DCD signals.
; This loop will clear the TX.IE and RX.IE bits if they are set.
;-
      CLR    R3             ;CLEAR THE LINE COUNTER.
2$:    MOV    R3,R0
      ASL    R0
      BIT    BITiBL(R0),ACTLNS
      BEQ    8$            ;DON'T TEST IF NOT ACTIVE LINE.
;+
; Clear all the DUT LNCTRL registers RTS bits.
;
      CLR    R0             ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
      MOV    #MAPLNS,R5     ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
      JSR    PC,WTWLNLC     ;CLEAR ALL THE DUT RTS BITS.
      MOV    #60,R4
      JSR    PC,DELAY       ;DELAY FOR 60 MS TO ALLOW SIGNALS TO SETTLE.

```

HARDWARE TEST

- RTSMCS -

```

7109
7110 ;+
7111 ; Check that at least one of associated DCD or CTS is clear and record states.
7112 031264 116304 004044 ;-
7113 031270 010477 150746          MOV    TXPLNB(R3),R4 ;GET THE ASSOCIATED LINE NUMBER.
7114 031274 017705 150750          MOV    R4,@CSRA      ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
7115 031300 012700 014000          MOV    @STATA,R5    ;GET THE STATE OF THE ASSOCIATED DCD, CTS BITS.
7116 031304 040500                MOV    #BIT12:BIT11,R0
7117 031306 001431                BIC    R5,R0        ;CHECK FOR BOTH DCD AND CTS SET.
7118                                BEQ    6$           ;GO REPORT RTS IS BAD IF BOTH ARE SET.
7119 ;+
7120 ; Set the RTS for the selected line and wait for either DCD or CTS to set.
7121 031310 010377 150726          ;-
7122 031314 052777 010000 150730  MOV    R3,@CSRA    ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.
7123 031322 012701 130074          BIS    #BIT12,@LNCTRA ;SET THE SELECTED LINE RTS.
7124 031326 032705 010000          MOV    #130074,R1  ;SPECIFY TO WAIT UP TO 60 MS FOR CTS TO SET.
7125 031332 001002                BIT    #BIT12,R5    ;CHECK PREVIOUS STATE OF DCD BIT.
7126 031334 012701 140074          BNE    4$          ;GO USE CTS IF DCD BIT WAS NOT CLEAR.
7127 031340 016702 150704          MOV    #140074,R1  ;SPECIFY TO WAIT UP TO 60 MS FOR DCD SET.
7128 031344 010477 150672          4$: MOV    STATA,R2    ;SPECIFY TO LOOK IN STAT REG FOR BIT TO SET.
7129 031350 004767 165222          MOV    R4,@CSRA    ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
7130 031354 103417                JSR    PC,WAIBIS    ;WAIT UP TO 60 MS FOR SIGNAL TO GO SET.
7131 031356 017700 150666          BCS    8$          ;SELECT NEXT LINE AND LOOP IF SIGNAL IS SET.
7132 031362 042700 163777          MOV    @STATA,R0   ;GET THE STATUS REGISTER CONTENTS.
7133 031366 040500                BIC    #163777,R0  ;REMOVE ALL BUT THE DCD AND CTS BITS.
7134 031370 001011                BIC    R5,R0        ;TEST FOR SIGNAL ONCE CLEAR, BUT NOW SET.
7135 031372                BNE    8$          ;GO LOOP IF SIGNAL HAS GONE FROM CLR TO SET.
7136 031372 012767 017336 152506  6$: ;Report RTS MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7137 031400 012767 012040 152504  MOV    #7902,ERRNBR ;SELECT THE ERROR NUMBER.
7138 031406 012701 007565          MOV    #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7139 031412 104460                MOV    #EM7902,R1  ;SELECT THE ERROR MESSAGE.
7140 031414 005203                ERROR ;>>>> ERROR <<<<<.
7141 031416 020327 000010          ; TRAP C$ERROR
7142 031422 002701                8$: INC    R3          ;SELECT THE NEXT LINE NUMBER.
7143                                CMP    R3,#NUMLNS  ;TEST FOR ALL LINES DONE.
7144                                BLT    2$          ;LOOP IF NOT ALL LINES DONE.
7145 ;+
7146 ; Set up a loop which handles one line per iteration.
7147 ; This loop sets all the RTSs and then clears them individually and waits for
7148 ; a response on the associated CTS and DCD signals.
7149 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7150 ;-
7151 031424 005003                CLR    R3          ;CLEAR THE LINE COUNTER.
7152 031426 010300 10$: MOV    R3,R0
7153 031430 006300                ASL    R0
7154 031432 036067 002370 150574  BIT    BITTBL(R0),ACTLNS
7155 031440 001466                BEQ    16$         ;DON'T TEST IF NOT ACTIVE LINE.
7156 ;+
7157 ; Set all the DUT LNCTRL registers RTS bits.
7158 ;-
7159 031442 012700 010000          MOV    #BIT12,R0   ;SPECIFY THAT RTS BITS ARE TO BE SET.
7160 031446 012705 000377          MOV    #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7161 031452 004767 165234          JSR    PC,WTWLNLC ;SET ALL THE DUT RTS BITS.
7162 031456 012704 000074          MOV    #60,R4
7163 031462 004767 162232          JSR    PC,DELAY    ;DELAY FOR 60 MS TO ALLOW SIGNALS TO SETTLE.
7164 ;+
7165 ; Check that at least one of associated DCD or CTS is set and record states.
7166 ;-

```

HARDWARE TEST

- RTSMCS -

```

7165 031466 116304 004044      MOVB  TXRLNB(R3),R4      ;GET THE ASSOCIATED LINE NUMBER.
7166 031472 010477 150544      MOV   R4,@CSRA          ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
7167 031476 017705 150546      MOV   @STATA,R5         ;GET THE STATE OF THE ASSOCIATED DCD, CTS BITS.
7168 031502 010500              MOV   R5,R0
7169 031504 042700 163777      BIC   #163777,R0        ;CHECK FOR BOTH DCD AND CTS CLEAR.
7170 031510 001431              BEQ   14$               ;GO REPORT RTS IS BAD IF BOTH ARE CLEAR.
7171
7172                               ;*
7173                               ; Clear the RTS for the selected line and wait for either DCD or CTS to clear.
7174 031512 010377 150524      MOV   R3,@CSRA          ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.
7175 031516 042777 010000 150526  BIC   #BIT12,@LNCTRA    ;CLEAR THE SELECTED LINE RTS.
7176 031524 012701 130074      MOV   #130074,R1        ;SPECIFY TO WAIT UP TO 60 MS FOR CTS TO CLEAR.
7177 031530 032705 010000      BIT   #BIT12,R5         ;CHECK PREVIOUS STATE OF DCD BIT.
7178 031534 001402              BEQ   12$               ;GO USE CTS IF DCD BIT WAS NOT SET.
7179 031536 012701 140074      MOV   #140074,R1        ;SPECIFY TO WAIT UP TO 60 MS FOR DCD CLEAR.
7180 031542 016702 150502 12$:  MOV   STATA,R2          ;SPECIFY TO LOOK IN STAT REG FOR BIT TO CLR.
7181 031546 010477 150470      MOV   R4,@CSRA          ;SELECT ASSOCIATED LINE IND.ADR.REG FIELD.
7182 031552 004767 164744      JSR   PC,WAIBIC         ;WAIT UP TO 60 MS FOR SIGNAL TO GO CLEAR.
7183 031556 103417              BCS   16$               ;SELECT NEXT LINE AND LOOP IF SIGNAL IS CLEAR.
7184 031560 017700 150464      MOV   @STATA,R0         ;GET THE STATUS REGISTER CONTENTS.
7185 031564 042705 163777      BIC   #163777,R5
7186 031570 040005              BIC   R0,R5
7187 031572 001011              BNE   16$               ;TEST FOR SIGNAL ONCE SET, BUT NOW CLEAR.
7188 031574              14$: ;Report RTS MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7189 031574 012767 017337 152304  MOV   #7903.,ERRNBR     ;SELECT THE ERROR NUMBER.
7190 031602 012767 012040 152302  MOV   #ER7801,ERRBLK    ;SELECT THE ERROR PRINT ROUTINE.
7191 031610 012701 007565      MOV   #EM7902,R1        ;SELECT THE ERROR MESSAGE.
7192 031614              ERROR                  ;
7193 031614 104460              >>>>> ERROR <<<<<.
7194 031616 005203              TRAP   C$ERROR
7195 031620 020327 000010 16$:  INC   R3                ;SELECT THE NEXT LINE NUMBER.
7196 031624 002700              CMP   R3,#NUMLNS        ;TEST FOR ALL LINES DONE.
7197 031626 005067 150432 60$:  BLT   10$               ;LOOP IF NOT ALL LINES DONE.
7198              CLR   CTRLCF          ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7199 031632              ENDTST
031632
031632 104401              L10043:
                                TRAP   C$ETST

```

HARDWARE TEST - DSRMS -

```

7201 .SBTTL HARDWARE TEST - DSRMS -
7202 ;*****
7203 ;*
7204 ;* - Data Set Ready Modem Signal Test -
7205 ;*
7206 ;* This test verifies that the DSR modem status signal is working
7207 ;* correctly. It will only be performed if either 25 pin or staggered
7208 ;* loopback is specified. This test uses the looped back DTR signals
7209 ;* to test the DSR signal. This test is performed on all the active
7210 ;* lines.
7211 ;*
7212 ;-----*****
7213 031634 BGNTST
031634
7214 ;* T18::
7215 ; Only perform this test if the DUT is in external or staggered loopback mode.
7216 ;-
7217 031634 032767 000002 150374 BIT #BIT1,LOPBC ;CHECK TYPE OF LOOPBACK MODE SELECTED.
7218 031642 001002 BNE 2$
7219 031644 000167 000372 JMP 60$ ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
7220 031650 000022 2$: TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
7221 031650 012767 000022 150410 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (80)
7222 031656 012767 177777 150400 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
7223 031664 012767 000001 152212 MOV #1,ERRTYP ;SET ERROR TYPE IN ERROR TABLE.
7224 031672 012767 017501 152206 MOV #8001,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
7225 031700 012767 007616 152202 MOV #EM8001,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
7226 ;*
7227 ; Reset the DUT to a known state, remove status codes from the FIFO.
7228 ; Clear TX and RX interrupt enable bits.
7229 ; This subroutine reports error >>>> 8001 <<<<.
7230 ;-
7231 031706 004767 161574 JSR PC,CLNRST ;RESET THE DUT.
7232 031712 103402 BCS 4$
7233 031714 000167 000322 JMP 60$ ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
7234 ;*
7235 ; Set up the TX/RX associated line number table.
7236 ;-
7237 031720 004767 161126 4$: JSR PC,ASLNTL ;SET UP THE ASSOCIATED LINE TABLES.
7238 ;*
7239 ; Set up a loop which handles one line per iteration.
7240 ; This loop clears all the DTRs and then sets them individually and waits for
7241 ; a response on the associated DSR signal.
7242 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7243 ;
7244 031724 005003 CLR R3 ;CLEAR THE LINE COUNTER.
7245 031726 010300 6$: MOV R3,R0
7246 031730 006300 ASL R0
7247 031732 036067 002370 150274 BIT BITTBL(R0),ACTLNS
7248 031740 001450 BEQ 10$ ;DON'T TEST IF NOT ACTIVE LINE.
7249 ;*
7250 ; Clear all the DUT LNCTRL registers DTR bits.
7251 ;-
7252 031742 005000 CLR R0 ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
7253 031744 012705 000377 MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7254 031750 004767 164736 JSR PC,WTWLN ;CLEAR ALL THE DUT DTR BITS.
7255 031754 012704 000050 MOV #40,R4
7256 031760 004767 161734 JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.

```

HARDWARE TEST

- DSRMS -

```

7257
7258 ; Check that the specified DSR is clear.
7259 ;-
7260 031764 010377 150252      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7261 031770 032777 100000 150252  BIT    #BIT15,@STATA
7262 031776 001027          BNE    8$            ;GO REPORT DSR IS BAD IF BIT IS NOT CLEAR.
7263
7264 ; Set the DTR for the associated line.
7265 ; NOTE: If the associated line is not selected, DTR will not have been tested
7266 ;       in the DTR test (Only an issue in staggered loopback).
7267 ;-
7268 032000 116304 004044      MOVB   TXRLNB(R3),R4  ;GET THE ASSOCIATED LINE NUMBER.
7269 032004 010477 150232      MOV    R4,@CSRA      ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7270 032010 052777 001000 150234  BIS    #BIT9,@LNCTRA ;SET THE ASSOCIATED LINE DTR.
7271
7272 ; Check that the selected line DSR is active.
7273 ;-
7274 032016 010377 150220      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7275 032022 012701 170050      MOV    #170050,R1    ;PASS TIMEOUT OF 40 MILLI SEC, AND BIT TO TEST.
7276 032026 016702 150216      MOV    STATA,R2      ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7277 032032 004767 164540      JSR    PC,WAIBIS     ;WAIT FOR DSR TO BECOME SET OR TIMEOUT.
7278 032036 103411          BCS    10$          ;SKIP ERROR REPORT IF SELECTED DSR IS SET.
7279
7280
7281 032040          8$: ;Report DSR MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7282 032040 012767 017502 152040  MOV    #8002,ERRNBR  ;SELECT THE ERROR NUMBER.
7283 032046 012767 012040 152036  MOV    #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7284 032054 012701 007654          MOV    #EM8002,R1    ;SELECT THE ERROR MESSAGE.
7285 032060          ERROR
7286 032062 104460          TRAP   C$ERRJR
7287 032064 005203          10$: INC    R3            ;SELECT THE NEXT LINE NUMBER.
7288 032070 020327 000010      CMP    R3,#NUMLNS   ;TEST FOR ALL LINES DONE.
7289          BLT    6$            ;LOOP IF NOT ALL LINES DONE.
7290
7291 ; Set up a loop which handles one line per iteration.
7292 ; This loop sets all the DTRs and then clears them individually and waits for
7293 ; a response on the selected DSR signal.
7294 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7295 ;-
7295 032072 005003          CLR    R3            ;CLEAR THE LINE COUNTER.
7296 032074 010300          12$: MOV    R3,R0
7297 032076 006300          ASL   R0
7298 032100 036067 002370 150126  BIT    BITBL(R0),ACTLNS
7299 032106 001451          BEQ   16$            ;DON'T TEST IF NOT ACTIVE LINE.
7300
7301 ; Set all the DUT LNCTRL registers DTR bits.
7302 ;-
7303 032110 012700 001000      MOV    #BIT9,R0      ;SPECIFY THAT DTR BITS ARE TO BE SET.
7304 032114 012705 000377      MOV    #MAPLNS,R5    ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7305 032120 004767 164566      JSR    PC,WTWLNLC    ;SET ALL THE DUT DTR BITS.
7306 032124 012704 000050      MOV    #40,R4
7307 032130 004767 161564      JSR    PC,DELAY      ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
7308
7309 ; Check that the specified DSR is set.
7310 ;-
7311 032134 010377 150102      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7312 032140 032777 100000 150102  BIT    #BIT15,@STATA

```





HARDWARE TEST

- RINGI -

```

7344 .SBTTL HARDWARE TEST - RINGI
7345 ;*****
7346 ;* Ring Indicator Modem Signal Test -
7347 ;*
7348 ;* This test verifies that the RI modem status signal is working
7349 ;* correctly. It will only be performed if either 25 pin or staggered
7350 ;* loopback is specified. This test uses the looped back DTR signals
7351 ;* to test the RI signal. This test is performed on all the active
7352 ;* lines.
7353 ;*
7354 ;--*****
7355
7356 032250 BGNTST
032250 T19::
7357 ;+
7358 ; Only perform this test if the DUT is in external or staggered loopback mode.
7359 ;-
7360 032250 032767 000002 147760 BIT #BIT1,LOPBC ;CHECK TYPE OF LOOPBACK MODE SELECTED.
7361 032256 001002 BNE 2$
7362 032260 000167 000372 JMP 60$ ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
7363 032264 000023 2$: TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
7364 032264 012767 000023 147774 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (81)
7365 032272 012767 177777 147764 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
7366 032300 012767 000001 151576 MOV #1,ERRTP ;SET ERROR TYPE IN ERROR TABLE.
7367 032306 012767 017645 151572 MOV #8101.,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
7368 032314 012767 007720 151566 MOV #EM8101,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
7369 ;+
7370 ; Reset the DUT to a known state, remove status codes from the FIFO.
7371 ; Clear TX and RX interrupt enable bits.
7372 ; This subroutine reports error >>>> 8101 <<<<<.
7373 ;-
7374 032322 004767 161160 JSR PC,CLNRST ;RESET THE DUT.
7375 032326 103402 BCS 4$
7376 032330 000167 000322 JMP 60$ ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
7377 ;+
7378 ; Set up the TX/RX associated line number table.
7379 ;-
7380 032334 004767 160512 4$: JSR PC,ASLNTL ;SET UP THE ASSOCIATED LINE TABLES.
7381 ;+
7382 ; Set up a loop which handles one line per iteration.
7383 ; This loop clears all the DTRs and then sets them individually and waits for
7384 ; a response on the associated RI signal.
7385 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7386 ;-
7387 032340 005003 CLR R3 ;CLEAR THE LINE COUNTER.
7388 032342 010300 6$: MOV R3,R0
7389 032344 006300 ASL R0
7390 032346 036067 002370 147660 BIT BITTBL(R0),ACTLNS
7391 032354 001450 BEQ 10$ ;DON'T TEST IF NOT ACTIVE LINE.
7392 ;+
7393 ; Clear all the DUT LNCTRL registers DTR bits.
7394 ;-
7395 032356 005000 CLR R0 ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
7396 032360 012705 000377 MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7397 032364 004767 164322 JSR PC,WTWLN ;CLEAR ALL THE DUT DTR BITS.
7398 032370 012704 000050 MOV #40.,R4
7399 032374 004767 161320 JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.

```

## HARDWARE TEST

- RINGI -

```

7400
7401      ;+
7402      ; Check that the specified RI is clear.
7403 032400 010377 147636      ;-
7404 032404 032777 020000 147636      MOV R3,@CSRA ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7405 032412 001020      BIT #BIT13,@STATA
7406      BNE 8$ ;GO REPORT RI IS BAD IF BIT IS NOT CLEAR.
7407      ;+
7408      ; Set the DTR for the associated line.
7409      ; NOTE: If the associated line is not selected, DTR will not have been tested
7410      ; in the DTR test (Only an issue in staggered loopback).
7411 032414 116304 004044      ;-
7412 032420 010477 147616      MOV B TXRLNB(R3),R4 ;GET THE ASSOCIATED LINE NUMBER.
7413 032424 052777 001000 147620      MOV R4,@CSRA ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7414      BIS #BIT9,@LNCTRA ;SET THE ASSOCIATED LINE DTR.
7415      ;+
7416      ; Check that the selected line RI is active.
7417 032432 010377 147604      ;-
7418 032436 012701 150050      MOV R3,@CSRA ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7419 032442 016702 147602      MOV #150050,R1 ;PASS TIMEOUT OF 40 MILLI-SEC, AND BIT TO TEST.
7420 032446 004767 164124      MOV STATA,R2 ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7421 032452 103411      JSR PC,WAIBIS ;WAIT FOR RI TO BECOME SET OR TIMEOUT.
7422      BCS 10$ ;SKIP ERROR REPORT IF SELECTED RI IS SET.
7423
7424 032454      8$: ;Report RI MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7425 032454 012767 017646 151424      MOV #8102.,ERRNBR ;SELECT THE ERROR NUMBER.
7426 032462 012767 012040 151422      MOV #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7427 032470 012701 007755      MOV #EM8102,R1 ;SELECT THE ERROR MESSAGE.
7428 032474      ERROR
7429 032476 005203      TRAP C$ERROR
7430 032500 020327 000010      10$: INC R3 ;SELECT THE NEXT LINE NUMBER.
7431 032504 002716      CMP R3,#NUMLNS ;TEST FOR ALL LINES DONE.
7432      BLT 6$ ;LOOP IF NOT ALL LINES DONE.
7433      ;+
7434      ; Set up a loop which handles one line per iteration.
7435      ; This loop sets all the DTRs and then clears them individually and waits for
7436      ; a response on the selected RI signal.
7437      ; This loop will clear the TX.IE and RX.IE bits if they are set.
7438 032506 005003      ;-
7439 032510 010300      CLR R3 ;CLEAR THE LINE COUNTER.
7440 032512 006300      12$: MOV R3,R0
7441 032514 036067 002370 147512      ASL R0
7442 032522 001451      BIT BITTBL(R0),ACTLNS
7443      BEQ 16$ ;DON'T TEST IF NOT ACTIVE LINE.
7444      ;+
7445      ; Set all the DUT LNCTRL registers DTR bits.
7446 032524 012700 001000      ;-
7447 032530 012705 000377      MOV #BIT9,R0 ;SPECIFY THAT DTR BITS ARE TO BE SET.
7448 032534 004767 164152      MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7449 032540 012704 000050      JSR PC,WTLNC ;SET ALL THE DUT DTR BITS.
7450 032544 004767 161150      MOV #40.,R4
7451      JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
7452      ;+
7453      ; Check that the specified RI is set.
7454 032550 010377 147466      ;-
7455 032554 032777 020000 147466      MOV R3,@CSRA ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7455      BIT #BIT13,@STATA

```

HARDWARE TEST

- RINGI -

```

7456 032562 001420          BEQ      14$          ;GO REPORT RI IS BAD IF BIT IS NOT SET.
7457
7458          ;+
7459          ; Clear the DTR for the associated line.
7460          ; NOTE: If the associated line is not selected, DTR will not have been tested
7461          ;       in the DTR test (Only an issue in staggered loopback).
7462 032564 116304 004044          MOVB   TXRLNB(R3),R4      ;GET THE ASSOCIATED LINE NUMBER.
7463 032570 010477 147446          MOV    R4,@CSRA         ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7464 032574 042777 001000 147450  BIC    #BIT9,@LNCTRA    ;CLEAR THE ASSOCIATED LINE DTR.
7465
7466          ;+
7467          ; Check that the selected line RI is clear.
7468 032602 010377 147434          ;--
7469 032606 012701 150050          MOV    R3,@CSRA         ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7470 032612 016702 147432          MOV    #150050,R1      ;PASS TIMEOUT OF 40 MILLI-SEC. AND BIT TO TEST.
7471 032616 004767 163700          MOV    STATA,R2        ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7472 032622 103411          JSR    PC,WAIBIC        ;WAIT FOR RI TO BECOME CLEAR OR TIMEOUT.
7473          BCS    16$          ;SKIP ERROR REPORT IF SELECTED RI IS CLEAR.
7474 032624          14$: ;Report RI MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7475 032624 012767 017647 151254  MOV    #8103.,ERRNBR    ;SELECT THE ERROR NUMBER.
7476 032632 012767 012040 151252  MOV    #ER7801,ERRBLK   ;SELECT THE ERROR PRINT ROUTINE.
7477 032640 012701 007755          MOV    #EM8102,R1      ;SELECT THE ERROR MESSAGE.
7478 032644          ERROR
7479 032646 104460          TRAP   C$ERROR
7480 032646 005203          16$: INC    R3          ;SELECT THE NEXT LINE NUMBER.
7481 032650 020327 000010          CMP    R3,#NUMLNS      ;TEST FOR ALL LINES DONE.
7482 032654 002715          BLT    12$          ;LOOP IF NOT ALL LINES DONE.
7483 032656 005067 147402          60$: CLR    CTRLCF        ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7484
7485 032662          ENDTST
032662
032662 104401          L10045: TRAP   C$ETST

```

HARDWARE TEST - CTSMS -

```

7487 .SBTTL HARDWARE TEST - CTSMS -
7488 ;*****
7489 ;* - Clear to Send Modem Signal Test -
7490 ;*
7491 ;* This test verifies that the CTS modem status signal is working
7492 ;* correctly. It will only be performed if either 25 pin or staggered
7493 ;* loopback is specified. This test uses the looped back RTS signals
7494 ;* to test the CTS signal. This test is performed on all the active
7495 ;* lines.
7496 ;*
7497 ;*
7498 ;*****
7499 032664 BGNTST
032664 T20::
7500 ;+
7501 ; Only perform this test if the DUT is in external or staggered loopback mode.
7502 ;-
7503 032664 032767 000002 147344 BIT #BIT1,LOPCK ;CHECK TYPE OF LOOPBACK MODE SELECTED.
7504 032672 001002 BNE 2$
7505 032674 000167 000372 JMP 60$ ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
7506 032700 000024 2$: TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
7507 032700 012767 000024 147360 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (82)
7508 032706 012767 177777 147350 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
7509 032714 012767 000001 151162 MOV #1,ERRTYP ;SET ERROR TYPE IN ERROR TABLE.
7510 032722 012767 020011 151156 MOV #8201,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
7511 032730 012767 010020 151152 MOV #EM8201,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
7512 ;+
7513 ; Reset the DUT to a known state, remove status codes from the FIFO.
7514 ; Clear TX and RX interrupt enable bits.
7515 ; This subroutine reports error >>>> 8201 <<<<.
7516 ;-
7517 032736 004767 160544 JSR PC,CLNRST ;RESET THE DUT.
7518 032742 103402 BCS 4$
7519 032744 000167 000322 JMP 60$ ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
7520 ;+
7521 ; Set up the TX/RX associated line number table.
7522 ;-
7523 032750 004767 160076 4$: JSR PC,ASLNTL ;SET UP THE ASSOCIATED LINE TABLES.
7524 ;+
7525 ; Set up a loop which handles one line per iteration.
7526 ; This loop clears all the RTS's and then sets them individually and waits for
7527 ; a response on the associated CTS signal.
7528 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7529 ;
7530 032754 005003 CLR R3 ;CLEAR THE LINE COUNTER.
7531 032756 010300 6$: MOV R3,R0
7532 032760 006300 ASL R0
7533 032762 036067 002370 147244 BIT BITTBL(R0),ACTLNS
7534 032770 001450 BEQ 10$ ;DON'T TEST IF NOT ACTIVE LINE.
7535 ;+
7536 ; Clear all the DUT LNCTRL registers RTS bits.
7537 ;-
7538 032772 005000 CLR R0 ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
7539 032774 012705 000377 MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7540 033000 004767 163706 JSR PC,W1WLNLC ;CLEAR ALL THE DUT RTS BITS.
7541 033004 012704 000050 MOV #40,R4
7542 033010 004767 160704 JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.

```

HARDWARE TEST

- CTSMS -

```

7543
7544 ; Check that the specified CTS is clear.
7545 ;-
7546 033014 010377 147222      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7547 033020 032777 004000 147222  BIT    #BIT11,@STATA
7548 033026 001020          BNE    8$           ;GO REPORT CTS IS BAD IF BIT IS NOT CLEAR.
7549
7550 ;+
7551 ; Set the RTS for the associated line.
7552 ; NOTE: If the associated line is not selected, RTS will not have been tested
7553 ;       in the RTS test (Only an issue in staggered loopback).
7554 033030 116304 004044      MOVB   TXRLNB(R3),R4 ;GET THE ASSOCIATED LINE NUMBER.
7555 033034 010477 147202      MOV    R4,@CSRA      ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7556 033040 052777 010000 147204  BIS    #BIT12,@LNCTRA ;SET THE ASSOCIATED LINE RTS.
7557
7558 ;+
7559 ; Check that the selected line CTS is active.
7560 033046 010377 147170      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7561 033052 012701 130050      MOV    #130050,R1    ;PASS TIMEOUT OF 40 MILLI-SEC, AND BIT TO TEST.
7562 033056 016702 147166      MOV    STATA,R2      ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7563 033062 004767 163510      JSR    PC,WAIBIS     ;WAIT FOR CTS TO BECOME SET OR TIMEOUT.
7564 033066 103411          BCS    10$          ;SKIP ERROR REPORT IF SELECTED CTS IS SET.
7565
7566
7567 033070          8$: ;Report CTS MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7568 033070 012767 020012 151010  MOV    #8202.,ERRNBR ;SELECT THE ERROR NUMBER.
7569 033076 012767 012040 151006  MOV    #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7570 033104 012701 010056          MOV    #EM8202,R1    ;SELECT THE ERROR MESSAGE.
7571 033110          ERROR
7572 033110 104460          TRAP   C$ERROR
7573 033112 005203          10$: INC    R3           ;SELECT THE NEXT LINE NUMBER.
7574 033114 020327 000010      CMP    R3,#NUMLNS   ;TEST FOR ALL LINES DONE.
7575 033120 002716          BLT    6$           ;LOOP IF NOT ALL LINES DONE.
7576
7577 ;+
7578 ; Set up a loop which handles one line per iteration.
7579 ; This loop sets all the RTSs and then clears them individually and waits for
7580 ; a response on the selected CTS signal.
7581 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7582 ;-
7583 033122 005003          CLR    R3           ;CLEAR THE LINE COUNTER.
7584 033124 010300          12$: MOV    R3,R0
7585 033126 006300          ASL    R0
7586 033130 036067 002370 147076  BIT    BITTBL(R0),ACTLNS
7587 033136 001451          BEQ    16$          ;DON'T TEST IF NOT ACTIVE LINE.
7588
7589 ;+
7590 ; Set all the DUT LNCTRL registers RTS bits.
7591 ;-
7592 033140 012700 010000      MOV    #BIT12,R0    ;SPECIFY THAT RTS BITS ARE TO BE SET.
7593 033144 012705 000377      MOV    #MAPLNS,R5   ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7594 033150 004767 163536      JSR    PC,WTWLNLC   ;SET ALL THE DUT RTS BITS.
7595 033154 012704 000050      MOV    #40.,R4
7596 033160 004767 160534      JSR    PC,DELAY     ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
7597
7598 ;+
7599 ; Check that the specified CTS is set.
7600 ;-
7601 033164 010377 147052      MOV    R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7602 033170 032777 004000 147052  BIT    #BIT11,@STATA

```

## HARDWARE TEST

- CTSMS -

```

7599 033176 001420          BEQ      14$          ;GO REPORT CTS IS BAD IF BIT IS NOT SET.
7600
7601          ;+
7602          ; Clear the RTS for the associated line.
7603          ; NOTE: If the associated line is not selected, RTS will not have been tested
7604          ;      in the RTS test (Only an issue in staggered loopback).
7605 033200 116304 004044          MOVB   TXRLNB(R3),R4      ;GET THE ASSOCIATED LINE NUMBER.
7606 033204 010477 147032          MOV    R4,@CSRA         ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7607 033210 042777 010000 147034  BIC    #BIT12,@LNCTRA   ;CLEAR THE ASSOCIATED LINE RTS.
7608
7609          ;+
7610          ; Check that the selected line CTS is clear.
7611 033216 010377 147020          MOV    R3,@CSRA         ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7612 033222 012701 130050          MOV    #130050,R1      ;PASS TIMEOUT OF 40 MILLI-SEC, AND BIT TO TEST.
7613 033226 016702 147016          MOV    STATA,R2       ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7614 033232 004767 163264          JSR    PC,WAIBIC       ;WAIT FOR CTS TO BECOME CLEAR OR TIMEOUT.
7615 033236 103411          BCS    16$            ;SKIP ERROR REPORT IF SELECTED CTS IS CLEAR.
7616
7617 033240          14$: ;Report CTS MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7618 033240 012767 020013 150640  MOV    #8203,ERRNBR    ;SELECT THE ERROR NUMBER.
7619 033246 012767 012040 150636  MOV    #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7620 033254 012701 010056          MOV    #EM8202,R1     ;SELECT THE ERROR MESSAGE.
7621 033260          ERROR
7622 033260 104460          TRAP   C$ERROR
7623 033262 005203          16$: INC    R3          ;SELECT THE NEXT LINE NUMBER.
7624 033264 020327 000010          CMP    R3,#NUMLNS    ;TEST FOR ALL LINES DONE.
7625 033270 002715          BLT    12$          ;LOOP IF NOT ALL LINES DONE.
7626 033272 005067 146766          60$: CLR    CTRLCF      ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7627
7628 033276          ENDTST
033276
033276 104401          L10046: TRAP   C$ETST

```

HARDWARE TEST

- DCDMS -

```

7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642 033300
      033300
7643
7644
7645
7646 033300 032767 000002 146730
7647 033306 001002
7648 033310 000167 000372
7649 033314 000025
7650 033314 012767 000025 146744
7651 033322 012767 177777 146734
7652 033330 012767 000001 150546
7653 033336 012767 020155 150542
7654 033344 012767 010122 150536
7655
7656
7657
7658
7659
7660 033352 004767 160130
7661 033356 103402
7662 033360 000167 000322
7663
7664
7665
7666 033364 004767 157462
7667
7668
7669
7670
7671
7672
7673 033370 005003
7674 033372 010300
7675 033374 006300
7676 033376 036067 002370 146630
7677 033404 001450
7678
7679
7680
7681 033406 005000
7682 033410 012705 000377
7683 033414 004767 163272
7684 033420 012704 000050
7685 033424 004767 160270
    
```

```

.SBTTL HARDWARE TEST - DCDMS -
;*****
; - Data Carrier Detected Modem Signal Test -
;
; This test verifies that the DCD modem status signal is working
; correctly. It will only be performed if either 25 pin or staggered
; loopback is specified. This test uses the looped back RTS signals
; to test the DCD signal. This test is performed on all the active
; lines.
;*****
      BGNTST
      T21::
; Only perform this test if the DUT is in external or staggered loopback mode.
;
      BIT    #BIT1,LOPBCK ;CHECK TYPE OF LOOPBACK MODE SELECTED.
      BNE    2$
      JMP    60$          ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
2$:      TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV    #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (83)
      MOV    #-1,CTRLCF  ;INDICATE THAT WE ARE IN A TEST.
      MOV    #1,ERRTYP  ;SET ERROR TYPE IN ERROR TABLE.
      MOV    #8301,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV    #EM8301,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;
; Reset the DUT to a known state, remove status codes from the FIFO.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 8301 <<<<<.
;
      JSR    PC,CLNRST  ;RESET THE DUT.
      BCS    4$
      JMP    60$          ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;
; Set up the TX/RX associated line number table.
;
4$:      JSR    PC,ASLNTL ;SET UP THE ASSOCIATED LINE TABLES.
;
; Set up a loop which handles one line per iteration.
; This loop clears all the RTSs and then sets them individually and waits for
; a response on the associated DCD signal.
; This loop will clear the TX.IE and RX.IE bits if they are set.
;
      CLR    R3          ;CLEAR THE LINE COUNTER.
6$:      MOV    R3,R0
      ASL    R0
      BIT    BITTBL(R0),ACTLNS
      BEQ    10$        ;DON'T TEST IF NOT ACTIVE LINE.
;
; Clear all the DUT LNCTRL registers RTS bits.
;
      CLR    R0
      MOV    #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
      JSR    PC,WTWLNLC ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
      MOV    #40,R4     ;CLEAR ALL THE DUT RTS BITS.
      JSR    PC,DELAY   ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
    
```

HARDWARE TEST

- DCDMS -

```

7686
7687 ;+
7688 ; Check that the specified DCD is clear.
7689 033430 010377 146606          MOV   R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7690 033434 032777 010000 146606  BIT   #BIT12,@STATA
7691 033442 001020          BNE   8$           ;GO REPORT DCD IS BAD IF BIT IS NOT CLEAR.
7692 ;+
7693 ; Set the RTS for the associated line.
7694 ; NOTE: If the associated line is not selected, RTS will not have been tested
7695 ;       in the RTS test (Only an issue in staggered loopback).
7696 ;-
7697 033444 116304 004044          MOVB  TXRLNB(R3),R4 ;GET THE ASSOCIATED LINE NUMBER.
7698 033450 010477 146566          MOV   R4,@CSRA      ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7699 033454 052777 010000 146570  BIS   #BIT12,@LNCTRA ;SET THE ASSOCIATED LINE RTS.
7700 ;+
7701 ; Check that the selected line DCD is active.
7702 ;-
7703 033462 010377 146554          MOV   R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7704 033466 012701 140050          MOV   #140050,R1    ;PASS TIMEOUT OF 40 MILLI-SEC, AND BIT TO TEST.
7705 033472 016702 146552          MOV   STATA,R2      ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7706 033476 004767 163074          JSR   PC,WAIBIS     ;WAIT FOR DCD TO BECOME SET OR TIMEOUT.
7707 033502 103411          BCS   10$          ;SKIP ERROR REPORT IF SELECTED DCD IS SET.
7708
7709
7710 033504          8$: ;Report DCD MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7711 033504 012767 020156 150374.  MOV   #8302.,ERRNBR ;SELECT THE ERROR NUMBER.
7712 033512 012767 012040 150372  MOV   #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7713 033520 012701 010160          MOV   #EM8302,R1    ;SELECT THE ERROR MESSAGE.
7714 033524          ERROR
7715 033526 104460          TRAP  C$ERROR
7716 033530 005203          10$: INC   R3           ;SELECT THE NEXT LINE NUMBER.
7717 033534 020327 070010          CMP   R3,#NUMLNS   ;TEST FOR ALL LINES DONE.
7718          BLT   6$           ;LOOP IF NOT ALL LINES DONE.
7719 ;+
7720 ; Set up a loop which handles one line per iteration.
7721 ; This loop sets all the RTSs and then clears them individually and waits for
7722 ; a response on the selected DCD signal.
7723 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7724 033536 005003          ;-
7725 033540 010300          12$: CLR   R3           ;CLEAR THE LINE COUNTER.
7726 033542 006300          MOV   R3,R0
7727 033544 036067 002370 146462  ASL   R0
7728 033552 001451          BIT   BITTBL(R0),ACTLNS
7729          BEQ   16$          ;DON'T TEST IF NOT ACTIVE LINE.
7730 ;+
7731 ; Set all the DUT LNCTRL registers RTS bits.
7732 033554 012700 010000          ;-
7733 033560 012705 000377          MOV   #BIT12,R0    ;SPECIFY THAT RTS BITS ARE TO BE SET.
7734 033564 004767 163122          MOV   #MAPLNS,R5   ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
7735 033570 012704 000050          JSR   PC,WTWLNLC   ;SET ALL THE DUT RTS BITS.
7736 033574 004767 160120          MOV   #40.,R4
7737          JSR   PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
7738 ;+
7739 ; Check that the specified DCD is set.
7740 033600 010377 146436          ;-
7741 033604 032777 010000 146436  MOV   R3,@CSRA      ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7741          BIT   #BIT12,@STATA

```



## HARDWARE TEST

- DCDMS -

```

7742 033612 001420          BEQ      14$          ;GO REPORT DCD IS BAD IF BIT IS NOT SET.
7743
7744          ;+
7745          ; Clear the RTS for the associated line.
7746          ; NOTE: If the associated line is not selected, RTS will not have been tested
7747          ; in the RTS test (Only an issue in staggered loopback).
7748 033614 116304 004044          MOVB   TXRLNB(R3),R4      ;GET THE ASSOCIATED LINE NUMBER.
7749 033620 010477 146416          MOV    R4,@CSRA         ;SET IND.ADR.REG FIELD TO ASSOCIATED LINE.
7750 033624 042777 010000 146420          BIC    #BIT12,@LNCTRA   ;CLEAR THE ASSOCIATED LINE RTS.
7751
7752          ;+
7753          ; Check that the selected line DCD is clear.
7754 033632 010377 146404          ;-
7755 033636 012701 140050          MOV    R3,@CSRA         ;SET IND.ADR.REG FIELD TO SELECTED LINE.
7756 033642 016702 146402          MOV    #140050,R1      ;PASS TIMEOUT OF 40 MILLI-SEC, AND BIT TO TEST.
7757 033646 004767 162650          MOV    STATA,R2        ;PASS THE ADDRESS OF THE REGISTER TO TEST.
7758 033652 103411          JSR    PC,WAIBIC       ;WAIT FOR DCD TO BECOME CLEAR OR TIMEOUT.
7759          BCS    16$          ;SKIP ERROR REPORT IF SELECTED DCD IS CLEAR.
7760 033654          14$: ;Report DCD MODEM CONTROL SIGNAL DEFECTIVE ON LINE nn.
7761 033654 012767 020157 150224          MOV    #8303.,ERRNBR   ;SELECT THE ERROR NUMBER.
7762 033662 012767 012040 150222          MOV    #ER7801,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7763 033670 012701 010160          MOV    #EM8302,R1     ;SELECT THE ERROR MESSAGE.
7764 033674          ERROR
7765 033674 104460          TRAP   C$ERROR
7766 033676 005203          16$: INC    R3          ;SELECT THE NEXT LINE NUMBER.
7767 033700 020327 000010          CMP    R3,#NUMLNS     ;TEST FOR ALL LINES DONE.
7768 033704 002715          BLT   12$          ;LOOP IF NOT ALL LINES DONE.
7769 033706 005067 146352          60$: CLR    CTRLCF       ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7770
7771 033712          ENDTST
033712
033712 104401          L10047: TRAP   C$ETST

```

HARDWARE TEST

- DTRINT -

```

7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784 033714
      033714
7785
7786
7787
7788 033714 032767 000002 146314
7789 033722 001002
7790 033724 000167 000352
7791 033730 000026
7792 033730 012767 000026 146330
7793 033736 012767 177777 146320
7794 033744 012767 000001 150132
7795 033752 012767 020321 150126
7796 033760 012767 010224 150122
7797
7798
7799
7800
7801
7802 033766 004767 157514
7803 033772 103402
7804 033774 000167 000302
7805
7806
7807
7808 034000 004767 157046
7809
7810
7811
7812
7813
7814
7815 034004 005003
7816 034006 010300
7817 034010 006300
7818 034012 036067 002370 146214
7819 034020 001444
7820
7821
7822
7823 034022 005000
7824 034024 012705 000377
7825 034030 004767 162656
7826 034034 012704 000050
7827 034040 004767 157654
7828

```

```

.SBTTL HARDWARE TEST - DTRINT
;*****
;*
;* Data Terminal Ready Signal Interactions Test -
;*
;* This test verifies that the DTR signal (and the looped back DSR and
;* RI status signals) do not interact with any other modem status signals.
;* It will only be performed if either 25 pin or staggered loopback is
;* specified. This test is performed on all active lines.
;*
;-----*****
                BGNTST
                                T22::
;+
; Only perform this test if the DUT is in external or staggered loopback mode.
;-
                BIT    #BIT1,LOPBCK    ;CHECK TYPE OF LOOPBACK MODE SELECTED.
                BNE    2$
                JMP    60$
2$:              TNUM == TNUM + 1    ;EXIT THIS TEST IF IN INTERNAL LOOPBACK.
                MOV    #TNUM,TSTNUM    ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
                MOV    #-1,CTRLCF      ;SET UP THE TEST NUMBER. (84)
                MOV    #1,ERRTYF      ;INDICATE THAT WE ARE IN A TEST.
                MOV    #8401,ERRNBR    ;SET ERROR TYPE IN ERROR TABLE.
                MOV    #EM8401,ERRMSG  ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
                MOV    #EM8401,ERRMSG  ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;+
; Reset the DUT to a known state, remove status codes from the fifo.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 8401 <<<<<.
;
                JSR    PC,CLNRST      ;RESET THE DUT.
                BCS    4$
                JMP    60$           ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;+
; Set up the TX/RX associated line number table.
;
4$:              JSR    PC,ASLNTL     ;SET UP THE ASSOCIATED LINE TABLES.
;+
; Set up a loop which handles one line per iteration.
; This loop clears all the DTRs and then sets them individually and checks
; for any responses on signals other than the associated RI and DSR signals.
; This loop will clear the TX.IE and RX.IE bits if they are set.
;-
                CLR    R3             ;CLEAR THE LINE COUNTER.
6$:              MOV    R3,R0
                ASL    R0
                BIT    BITTBL(R0),ACTLNS
                BEQ    8$             ;DON'T TEST IF NOT ACTIVE LINE.
;+
; Clear all the DUT LNCTRL registers DTR bits.
;-
                CLR    R0             ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
                MOV    #MAPLNS,R5    ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
                JSR    PC,WTWLNC     ;CLEAR ALL THE DUT DTR BITS.
                MOV    #40.,R4
                JSR    PC,DELAY      ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
;+

```

HARDWARE TEST

- DTRINT -

```

7829 ; Record the states of the modem status signals.
7830 ;
7831 034044 004767 161542 ; JSR PC,SAVMST ;SAVE THE PRESENT MODEM STATUS STATES.
7832 ;
7833 ; Set the DTR for the selected line.
7834 ;
7835 034050 010377 146166 ; MOV R3,@CSRA ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.
7836 034054 052777 001000 146170 ; BIS #BIT9,@LNCTRA ;SET THE SELECTED LINE DTR.
7837 034062 012704 000050 ; MOV #40.,R4
7838 034066 004767 157626 ; JSR PC,DELAY ;ALLOW 40 MS FOR STATUS SIGNALS TO STABILIZE.
7839 ;
7840 ; Check the present DUT STAT register contents against previous.
7841 ; If any undesired changes have taken place, report the errors.
7842 ;
7843 034072 116301 004044 ; MOVB TXRLNB(R3),R1 ;SELECT SPECIAL TREATMENT FOR ASSOCIATED LINE.
7844 034076 012702 120000 ; MOV #BIT15!BIT13,R2 ;IGNORE DSR AND RI ON ASSOCIATED LINE.
7845 034102 004767 157444 ; JSR PC,CMPMST ;COMPARE OLD AND NEW STAT CONTENTS.
7846 034106 103411 ; BCS 8$ ;SKIP ERROR REPORT IF NO DISCREPANCIES FOUND.
7847 ;Report INTERACTIONS FOUND BETWEEN DTR FOR LINE nn AND THE FOLLOWING SIGNALS:
7848 034110 012767 020322 147770 ; MOV #8402.,ERRNBR ;SELECT THE ERROR NUMBER.
7849 034116 012767 012066 147766 ; MOV #ER8401,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7850 034124 012701 010300 ; MOV #EM8402,R1 ;SELECT THE DTR ERROR MESSAGES.
7851 034130 ; ERROR ;ER8401 USES R1, R2, AND R3 VALUES.
7851 034130 104460 ; TRAP C$ERROR
7852 ;
7853 ; Select the next line and loop if not all possible lines have been handled.
7854 ;
7855 034132 005203 8$: INC R3 ;SELECT THE NEXT LINE NUMBER.
7856 034134 020327 000010 ; CMP R3,#NUMLNS ;TEST FOR ALL LINES DONE.
7857 034140 002722 ; BLT 6$ ;LOOP IF NOT ALL LINES DONE.
7858 ;
7859 ; Set up a loop which handles one line per iteration.
7860 ; This loop sets all the DTRs and then clears them individually and checks
7861 ; for any responses on signals other than the associated RI and DSR signals.
7862 ; This loop will clear the TX.IE and RX.IE bits if they are set.
7863 ;
7864 034142 005003 ; CLR R3 ;CLEAR THE LINE COUNTER.
7865 034144 010300 10$: MOV R3,R0
7866 034146 006300 ; ASL R0
7867 034150 036067 002370 146056 ; BIT BITTBL(R0),ACTLNS
7868 034156 001445 ; BEQ 12$ ;DON'T TEST IF NOT ACTIVE LINE.
7869 ;
7870 ; Set all the DUT LNCTRL registers DTR bits.
7871 ;
7872 034160 012700 001000 ; MOV #BIT9,R0 ;SPECIFY THAT DTR BIT'S ARE TO BE SET.
7873 034164 012705 000377 ; MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTPLS ARE TO BE CHANGED.
7874 034170 004767 162516 ; JSR PC,WTWLNC ;SET ALL THE DUT DTR BITS.
7875 034174 012704 000050 ; MOV #40.,R4
7876 034200 004767 157514 ; JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
7877 ;
7878 ; Record the states of the modem status signals.
7879 ;
7880 034204 004767 161402 ; JSR PC,SAVMST ;SAVE THE PRESENT MODEM STATUS STATES.
7881 ;
7882 ; Clear the DTR for the selected line.
7883 ;
7884 034210 010377 146026 ; MOV R3,@CSRA ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.

```

HARDWARE TEST

- DTRINT -

```

7885 034214 042777 001000 146030      BIC  #BIT9,@LNCTRA ;CLEAR THE SELECTED LINE DTR.
7886 034222 012704 000050      MOV  #40.,R4
7887 034226 004767 157466      JSR  PC,DELAY      ;ALLOW 40 MS FOR STATUS SIGNALS TO STABILIZE.
7888
7889      ;+
7890      ; Check the present DUT STAT register contents against previous.
7891      ; If any undesired changes have taken place, report the errors.
7892      ;-
7892 034232 116301 004044      MOV  TXRLNB(R3),R1 ;SELECT SPECIAL TREATMENT FOR ASSOCIATED LINE.
7893 034236 012702 120000      MOV  #BIT15!BIT13,R2 ;IGNORE DSR AND RI ON ASSOCIATED LINE.
7894 034242 004767 157304      JSR  PC,CMPMST    ;COMPARE OLD AND NEW STAT CONTENTS.
7895 034246 103411      BCS  12$          ;SKIP ERROR REPORT IF NO DISCREPANCIES FOUND.
7896      ;Report INTERACTIONS FOUND BETWEEN DTR FOR LINE nn AND THE FOLLOWING SIGNALS:
7897 034250 012767 020323 147630      MOV  #8403.,ERRNBR ;SELECT THE ERROR NUMBER.
7898 034256 012767 012066 147626      MOV  #ER8401,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7899 034264 012701 010300      MOV  #EM8402,R1   ;SELECT THE DTR ERROR MESSAGES.
7900 034270      ERROR      ;ER8401 USES R1, R2, AND R3 VALUES.
7901      ;+
7902      ; Select the next line and loop if not all possible lines have been handled.
7903      ;-
7904 034272 005203      12$: INC  R3          ;SELECT THE NEXT LINE NUMBER.
7905 034274 020327 000010      CMP  R3,#NUMLNS  ;TEST FOR ALL LINES DONE.
7906 034300 002721      BLT  10$          ;LOOP IF NOT ALL LINES DONE.
7907
7908 034302 005067 145756      60$: CLR  CTRLCF    ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7909
7910 034306      ENDTST
      034306
      034306 104401      L10050: TRAP  C$ETST
    
```

HARDWARE TEST

- RTSINT -

```

7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923 034310
      034310
7924
7925
7926
7927 034310 032767 000002 145720
7928 034316 001002
7929 034320 000167 000352
7930 034324 000027
7931 034324 012767 000027 145734
7932 034332 012767 177777 145724
7933 034340 012767 000001 147536
7934 034346 012767 020465 147532
7935 034354 012767 010323 147526
7936
7937
7938
7939
7940
7941 034362 004767 157120
7942 034366 103402
7943 034370 000167 000302
7944
7945
7946
7947 034374 004767 156452
7948
7949
7950
7951
7952
7953
7954 034400 005003
7955 034402 010300
7956 034404 006300
7957 034406 036067 002370 145620
7958 034414 001444
7959
7960
7961
7962 034416 005000
7963 034420 012705 000377
7964 034424 004767 162262
7965 034430 012704 000050
7966 034434 004767 157260
7967

```

```

.SBTTL HARDWARE TEST - RTSINT -
;*****
;* - Request To Send Signal Interactions Test
;*
;* This test verifies that the RTS signal (and the looped back DCD and CTS
;* status signals) do not interact with any other modem status signals.
;* It will only be performed if either 25 pin or staggered loopback is
;* specified. This test is performed on all active lines.
;*****
BGNTST
T23::
;+
; Only perform this test if the DUT is in external or staggered loopback mode.
;-
      BIT    #BIT1,LOPBCK    ;CHECK TYPE OF LOOPBACK MODE SELECTED.
      BNE    2$
      JMP    60$
2$:   TNUM == TNUM + 1      ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV    #TNUM,TSTNUM   ;SET UP THE TEST NUMBER. (85)
      MOV    #-1,CTRLCF     ;INDICATE THAT WE ARE IN A TEST.
      MOV    #1,ERRTYP      ;SET ERROR TYPE IN ERROR TABLE.
      MOV    #8501,ERRNBR   ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV    #EM8501,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;+
; Reset the DUT to a known state, remove status codes from the fifo.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 8501 <<<<<.
;-
      JSR    PC,CLNRST      ;RESET THE DUT.
      BCS    4$
      JMP    60$           ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;+
; Set up the TX/RX associated line number table.
;-
4$:   JSR    PC,ASLNTL      ;SET UP THE ASSOCIATED LINE TABLES.
;+
; Set up a loop which handles one line per iteration.
; This loop clears all the RTSs and then sets them individually and checks
; for any responses on signals other than the associated DCD and CTS signals.
; This loop will clear the TX.IE and RX.IE bits if they are set.
;-
      CLR    R3             ;CLEAR THE LINE COUNTER.
6$:   MOV    R3,R0
      ASL    R0
      BIT    BITTBL(R0),ACTLNS
      BEQ    8$             ;DON'T TEST IF NOT ACTIVE LINE.
;+
; Clear all the DUT LNCTRL registers RTS bits.
;-
      CLR    R0             ;SPECIFY THAT ALL LNCTRL BITS TO BE CLEARED.
      MOV    #MAPLNS,R5     ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
      JSR    PC,WTWLNC      ;CLEAR ALL THE DUT RTS BITS.
      MOV    #40.,R4
      JSR    PC,DELAY       ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
;+

```

HARDWARE TEST

- RTSINT -

```

7968 ; Record the states of the modem status signals.
7969 ;
7970 034440 004767 161146 JSR PC,SAVMST ;SAVE THE PRESENT MODEM STATUS STATES.
7971 ;
7972 ; Set the RTS for the selected line.
7973 ;
7974 034444 010377 145572 MOV R3,@CSRA ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.
7975 034450 052777 010000 145574 BIS #BIT12,@LNCTRA ;SET THE SELECTED LINE RTS.
7976 034456 012704 000050 MOV #40.,R4
7977 034462 004767 157232 JSR PC,DELAY ;ALLOW 40 MS FOR STATUS SIGNALS TO STABILIZE.
7978 ;
7979 ; Check the present DUT STAT register contents against previous.
7980 ; If any undesired changes have taken place, report the errors.
7981 ;
7982 034466 116301 004044 MOVB TXRLNB(R3),R1 ;SELECT SPECIAL TREATMENT FOR ASSOCIATED LINE.
7983 034472 012702 014000 MOV #BIT12!BIT11,R2 ;IGNORE DCD AND CTS ON ASSOCIATED LINE.
7984 034476 004767 157050 JSR PC,CMPMST ;COMPARE OLD AND NEW STAT CONTENTS.
7985 034502 103411 BCS 8$ ;SKIP ERROR REPORT IF NO DISCREPANCIES FOUND.
7986 ;Report INTERACTIONS FOUND BETWEEN RTS FOR LINE nn AND THE FOLLOWING SIGNALS:
7987 034504 012767 020466 147374 MOV #8502.,ERRNBR ;SELECT THE ERROR NUMBER.
7988 034512 012767 012066 147372 MOV #ER8401,ERRBLK ;SELECT THE ERROR PRINT ROUTINE.
7989 034520 012701 010377 MOV #EM8502,R1 ;SELECT THE RTS ERROR MESSAGES.
7990 034524 104460 ERROR ;ER1901 USES R1, R2, AND R3 VALUES.
7991 ; TRAP C$ERROR
7992 ;
7993 ; Select the next line and loop if not all possible lines have been handled.
7994 034526 005203 8$: INC R3 ;SELECT THE NEXT LINE NUMBER.
7995 034530 020327 000010 CMP R3,#NUMLNS ;TEST FOR ALL LINES DONE.
7996 034534 002722 BLT 6$ ;LOOP IF NOT ALL LINES DONE.
7997 ;
7998 ; Set up a loop which handles one line per iteration.
7999 ; This loop sets all the RTSS and then clears them individually and checks
8000 ; for any responses on signals other than the associated DCD and CTS signals.
8001 ; This loop will clear the TX.IE and RX.IE bits if they are set.
8002 ;
8003 034536 005003 CLR R3 ;CLEAR THE LINE COUNTER.
8004 034540 010300 10$: MOV R3,R0
8005 034542 006300 ASL R0
8006 034544 036067 002370 145462 BIT BITTBL(R0),ACTLNS
8007 034552 001445 BEQ 12$ ;DON'T TEST IF NOT ACTIVE LINE.
8008 ;
8009 ; Set all the DUT LNCTRL registers RTS bits.
8010 ;
8011 034554 012700 010000 MOV #BIT12,R0 ;SPECIFY THAT RTS BITS ARE TO BE SET.
8012 034560 012705 000377 MOV #MAPLNS,R5 ;SPECIFY THAT ALL LNCTRLS ARE TO BE CHANGED.
8013 034564 004767 162122 JSR PC,WTWLNCR ;SET ALL THE DUT RTS BITS.
8014 034570 012704 000050 MOV #40.,R4
8015 034574 004767 157120 JSR PC,DELAY ;DELAY FOR 40 MS TO ALLOW SIGNALS TO SETTLE.
8016 ;
8017 ; Record the states of the modem status signals.
8018 ;
8019 034600 004767 161006 JSR PC,SAVMST ;SAVE THE PRESENT MODEM STATUS STATES.
8020 ;
8021 ; Clear the RTS for the selected line.
8022 ;
8023 034604 010377 145432 MOV R3,@CSRA ;SELECT THE SELECTED LINE IND.ADR.REG FIELD.

```



HARDWARE TEST - TXLNS -

```

8051
8052
8053
8054
8055
8056
8057
8058
8059
8060
8061
8062
8063
8064 034704
      034704
8065
8066
8067
8068 034704 032767 000002 145324
8069 034712 001002
8070 034714 000167 000274
8071 034720 000030
8072 034720 012767 000030 145340
8073 034726 012767 177777 145330
8074 034734 012767 000001 147142
8075 034742 012767 020631 147136
8076 034750 012767 010403 147132
8077
8078
8079
8080
8081
8082 034756 004767 156524
8083 034762 103402
8084 034764 000167 000224
8085
8086
8087
8088 034770 004767 156056
8089
8090
8091
8092
8093 034774 012705 000377
8094 035000 005000
8095 035002 004767 161704
8096 035006 012700 156430
8097 035012 004767 161724
8098
8099
8100
8101 035016 016705 145212
8102 035022 004767 161204
8103
8104
8105
8106 035026 016705 145202

```

```

.SBTTL HARDWARE TEST - TXLNS -
;*****
;
; TX Lines Test -
;
; This test verifies that the TX lines and RX lines are working correctly
; through the device cables, distribution panel (if installed), and
; loopback connector(s). It will only be performed if either 25 pin or
; staggered loopback is specified. This test sends a character on each
; active line and verifies that the proper character is detected on each
; receive line.
;*****
      BGNTST
      T24::
;+
; Only perform this test if the DUT is in external or staggered loopback mode.
;-
      BIT    #BIT1,LOPBCK    ;CHECK TYPE OF LOOPBACK MODE SELECTED.
      BNE    2$
      JMP    60$
2$:    TNUM  == TNUM + 1    ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV    #TNUM,TSTNUM  ;SET UP THE TEST NUMBER. (86)
      MOV    #-1,CTRLCF    ;INDICATE THAT WE ARE IN A TEST.
      MOV    #1,ERRTYP     ;SET ERROR TYPE IN ERROR TABLE.
      MOV    #8601,ERRNBR  ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV    #EM8601,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;+
; Reset the DUT to a known state, remove status codes from the fifo.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 8601 <<<<<.
;
      JSR    PC,CLNRST     ;RESET THE DUT.
      BCS    4$
      JMP    60$          ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;+
; Set up the TX/RX associated line number table.
;
4$:    JSR    PC,ASLNTL    ;SET UP THE ASSOCIATED LINE TABLES.
;+
; Set up transmission an reception parameters for all lines.
; 9600 baud,8 char,1 stopbit,no parity.
;-
      MOV    #MAPLNS,R5   ;INDICATE ALL LINES TO BE SET UP.
      CLR    R0           ;SPECIFY ALL LNCTRL BITS TO BE CLEARED.
      JSR    PC,WTWLNC    ;CLEAR ALL LNCTRL BITS.
      MOV    #156430,R0   ;SET UP BAUD RATE,ETC.
      JSR    PC,WTWLPR    ;SET COMMUNICATION PARAMETERS ON ALL LINES.
;+
; Enable transmitters on all active lines.
;-
      MOV    ACTLNS,R5    ;PASS ACTIVE LINE BIT MAP.
      JSR    PC,TXENBL    ;ENABLE TRANSMISSIONS ON ALL LINES.
;+
; Enable reception on all lines associated with active lines.
;-
      MOV    ACTLNS,R5    ;GET ACTIVE LINES BIT MAP TO CONVERT.

```



HARDWARE TEST

- TXLNS -

```

8107 035032 004767 156606      JSR    PC,CONMAP      ;CONVERT TO ASSOCIATED LINE MAP.
8108 035036 012700 000004      MOV    #BIT2,R0      ;INDICATE RX.ENABLE BIT TO BE SET.
8109 035042 004767 161644      JSR    PC,WTWLNC     ;ENABLE RX ON THE ASSOCIATED LINES.
8110
8111      ;+
8112      ; Set up a loop which sends a character on each active line.
8113 035046 005003      CLR    R3            ;CLEAR THE LINE COUNTER.
8114 035050 010300      6$:  MOV    R3,R0
8115 035052 006300      ASL    R0
8116 035054 036067 002370 145152  BIT    BITTBL(R0),ACTLNS
8117 035062 001406      BEQ    8$           ;DON'T SEND IF NOT ACTIVE LINE.
8118 035064 010377 145152      MOV    R3,@CSRA     ;SET UP THE IND.ADR.REG FIELD FOR PROPER LINE.
8119 035070 052700 100000      BIS    #BIT15,R0    ;OR IN THE TX.DATA.VALID BIT TO THE TX CHAR.
8120 035074 010077 145144      MOV    R0,@TXCHA    ;SEND THE CHARACTER.
8121 035100 005203      8$:  INC    R3           ;SELECT THE NEXT LINE NUMBER.
8122 035102 020327 000010      CMP    R3,#NUMLNS   ;TEST FOR ALL LINES DONE.
8123 035106 002760      BLT    6$           ;LOOP IF NOT ALL LINES DONE.
8124
8125      ;+
8126      ; Set up loop which waits for all chars to be received or time-out to occur.
8127 035110 016705 145120      MOV    ACTLNS,R5
8128 035114 012701 000062      MOV    #50.,R1      ;SELECT 50 MS TIME-OUT VALUE.
8129 035120 012702 100000      MOV    #BIT15,R2    ;SELECT DATA.VALID BIT TO BE TESTED.
8130 035124 016704 145114      MOV    RBUFA,R4     ;SPECIFY THAT BIT IS FOUND IN RBUF REG.
8131 035130 010203      10$: MOV    R2,R3        ;SPECIFY TO WAIT FOR BIT TO BE SET.
8132 035132 004767 157236      JSR    PC,MSLGET    ;WAIT FOR A VALID CHAR TO BE RECEIVED.
8133 035136 103016      BCC    12$          ;GO ANALYSE PROBLEM IF TIME-OUT.
8134 035140 010003      MOV    R0,R3        ;TEST THE RX CHARACTER TO VERIFY THAT
8135 035142 000303      SWAB   R3           ; IT IS A VALID COMBINATION OF LINE
8136 035144 042700 177600      BIC    #177600,R0   ; NUMBER AND DATA. IGNORE ANY ERROR
8137 035150 042703 177760      BIC    #177760,R3   ; BITS WHICH ARE SET.
8138 035154 006303      ASL    R3           ;
8139 035156 026300 004004      CMP    TXRXLB(R3),R0 ;
8140 035162 001362      BNE    10$          ;RX CHAR A VALID COMBINATION? NO, IGNORE IT.
8141 035164 046005 002370      BIC    BITTBL(R0),R5 ;YES, CLEAR THE BIT FOR THE ASSOCIATED LINE.
8142 035170 001357      BNE    10$          ;ALL ACTIVE LINES DONE? NO, LOOP.
8143 035172 000410      BR     60$         ;YES, SUCCESSFUL COMPLETION. EXIT THE TEST.
8144
8145      ;+
8146      ; Not all active lines received correct characters.
8147      ; Report "TX/RX LINE ERROR ON THE FOLLOWING LOOPED BACK TX LINES:"
8148      ; Error number          >>>>> 8602 <<<<<.
8149
8150 035174 005267 146706      12$: INC    ERRNBR    ;SELECT PROPER ERROR NUMBER (8602).
8151 035200 012767 011742 146704  MOV    #ER6401,ERRBLK ;SELECT PROPER ERROR REPORTING ROUTINE.
8152 035206 012701 010464      MOV    #EM8602,R1   ;SELECT PROPER ERROR MESSAGE.
8153 035212 104460      ERROR
8154      TRAP    C$ERROR
8155 035214 005067 145044      60$: CLR    CTRLCF    ;INDICATE THAT WE ARE NOT WITHIN A TEST.
8156
8157 035220      ENDTST
      L10052: TRAP    C$ETST
035220
035220 104401

```

HARDWARE TEST

- TXLINT -

```

8159
8160
8161
8162
8163
8164
8165
8166
8167
8168
8169
8170
8171
8172 035222
      035222
8173
8174
8175
8176 035222 032767 000002 145006
8177 035230 001002
8178 035232 000' 07 000376
8179 035236 000031
8180 035236 012767 000031 145022
8181 035244 012767 177777 145012
8182 035252 012767 000001 146624
8183 035260 012767 020775 146620
8184 035266 012767 010554 146614
8185
8186
8187
8188
8189
8190 035274 004767 156206
8191 035300 103402
8192 035302 000167 000326
8193
8194
8195
8196 035306 004767 ;55540
8197
8198
8199
8200
8201 035312 012705 000377
8202 035316 005000
8203 035320 004767 161366
8204 035324 012700 156430
8205 035330 004767 161406
8206
8207
8208
8209 035334 016705 144674
8210 035340 004767 160666
8211
8212
8213
8214 035344 016705 144664

```

```

.SBTTL HARDWARE TEST - TXLINT -
;*****
;* - TX Lines Interactions Test -
;*
;* This test verifies that each TX line does not interact with any other
;* TX lines or modem control signals. It will only be performed if
;* either 25 pin or staggered loopback is specified. This test causes
;* a BREAK condition on each active line individually and looks for any
;* characters generated on other lines or any changes to modem control
;* signals on active lines.
;*****
      BGNTST
      T25::
;+
; Only perform this test if the DUT is in external or staggered loopback mode.
;-
      BIT      #BIT1,LOPBCK      ;CHECK TYPE OF LOOPBACK MODE SELECTED.
      BNE      2$
      JMP      60$
2$:      TNUM == TNUM + 1      ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV      #TNUM,TSTNUM      ;SET UP THE TEST NUMBER. (87)
      MOV      #-1,CTRLCF      ;INDICATE THAT WE ARE IN A TEST.
      MOV      #1,ERRTYP      ;SET ERROR TYPE IN ERROR TABLE.
      MOV      #8701,ERRNBR      ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV      #EM8701,ERRMSG      ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
;+
; Reset the DUT to a known state, remove status codes from the fifo.
; Clear TX and RX interrupt enable bits.
; This subroutine reports error >>>> 8701 <<<<<.
;-
      JSR      PC,CLNRST      ;RESET THE DUT.
      BCS      4$
      JMP      60$      ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;+
; Set up the TX/RX associated line number table.
;-
4$:      JSR      PC,ASLNTL      ;SET UP THE ASSOCIATED LINE TABLES.
;+
; Set up transmission an reception parameters for all lines.
; 9600 baud,8 char,1 stopbit,no parity.
;-
      MOV      #MAPLNS,R5      ;INDICATE ALL LINES TO BE SET UP.
      CLR      R0      ;SPECIFY ALL LNCTRL BITS TO BE CLEARED.
      JSR      PC,WTWLNC      ;CLEAR ALL LNCTRL BITS.
      MOV      #156430,R0      ;SET UP BAUD RATE,ETC.
      JSR      PC,WTWLPR      ;SET COMMUNICATION PARAMETERS ON ALL LINES.
;+
; Enable transmitters on all active lines.
;-
      MOV      ACTLNS,R5      ;PASS ACTIVE LINE BIT MAP.
      JSR      PC,TXENBL      ;ENABLE TRANSMISSIONS ON ALL LINES.
;+
; Enable reception on all lines associated with active lines.
;-
      MOV      ACTLNS,R5      ;GET THE ACTIVE LINE MAP TO CONVERT.

```

HARDWARE TEST

- TXLINT -

```

8215 035350 004767 156270      JSR    PC,CONMAP      ;CONVERT TO ASSOCIATED LINE MAP.
8216 035354 012700 000004      MOV    #BIT2,R0      ;INDICATE RX.ENABLE BIT TO BE SET.
8217 035360 004767 161326      JSR    PC,WTWLNLC    ;ENABLE RX ON THE ASSOCIATED LINES.
8218
8219      ;*
8219      ; Set up a loop which sets the BREAK bit on each active line individually and
8220      ; checks for interactions.
8221      ;-
8222 035364 005003      CLR    R3            ;CLEAR THE LINE COUNTER.
8223 035366 010300      6$:    MOV    R3,R0
8224 035370 006300      ASL    R0
8225 035372 036067 002370 144634  BIT    BITTBL(R0),ACTLNS
8226 035400 001503      BEQ    16$          ;LINE ACTIVE? NO, SKIP THE LINE.
8227
8228      ;*
8228      ; Clear the BREAK bits for all lines.
8229      ; Delay 10 ms to all lines to get out of BREAK condition.
8230      ;-
8231 035402 005000      CLR    R0            ;CLEAR LINE COUNTER.
8232 035404 010077 144632      8$:    MOV    R0,@CSRA    ;SET UP THE IND.ADR.REG FILED FOR PROPER LINE.
8233 035410 042777 000010 144634  BIC    #BIT3,@LNCTRA ;CLEAR A BREAK BIT.
8234 035416 005200      INC    R0            ;SET LINE COUNTER TO NEXT LINE.
8235 035420 020027 000010      CMP    R0,#NUMLNS
8236 035424 001367      BNE    8$           ;DONE? NO, LOOP TO DO NEXT LINE.
8237 035426 012704 000012      MOV    #10.,R4      ;SELECT 10 MS DELAY.
8238 035432 004767 156262      JSR    PC,DELAY     ;DELAY TO ALLOW BREAKS TO BE CLEARED.
8239
8240      ;*
8240      ; Record the states of the modem control signals.
8241      ; Purge the FIFO of any unwanted characters.
8242      ; This routine reports errors with numbers >>>> 8702 thru 8704 <<<<.
8243      ;-
8244 035436 004767 160150      JSR    PC,SAVMST    ;RECORD THE STATES OF MODEM STATUS SIGNALS.
8245 035442 012767 020776 146436  MOV    #8702.,ERRNBR
8246 035450 004767 157442      JSR    PC,PUFIFR    ;PURGE THE FIFO.
8247 035454 103067      BCC    60$          ;PURGE SUCCESSFUL? NO, EXIT TEST.
8248
8249      ;*
8249      ; Set the break on the selected line.
8250      ;-
8251 035456 010377 144560      MOV    R3,@CSRA    ;SET UP THE IND.ADR.REG FIELD FOR PROPER LINE.
8252 035462 052777 000010 144562  BIS    #BIT3,@LNCTRA ;SET THE BREAK BIT IN LNCTRL REGISTER.
8253 035470 012704 000012      MOV    #10.,R4      ;SELECT 10 MS DELAY.
8254 035474 004767 156220      JSR    PC,DELAY     ;DELAY TO ALLOW BREAK TO BE SET.
8255
8256      ;*
8256      ; Verify that only the associated line receives characters.
8257      ;-
8258 035500 012700 000013      10$:   MOV    #11.,R0      ;ALLOW UP TO 10 "LEGAL" CHARACTERS.
8259 035504 017701 144534      MOV    @RBUFA,R1    ;READ A CHARACTER FROM THE DUT.
8260 035510 100022      BPL    14$          ;DATA VALID? NO, CHECK MODEM SIGNALS.
8261 035512 000301      SWAB   R1
8262 035514 042701 177760      BIC    #177760,R1    ;GET LINE NUMBER OF RX CHAR.
8263 035520 126103 004044      CMPB   TXRLNB(R1),R3 ;COMPARE AGAINST TX ASSOCIATED LINE NUMBER.
8264 035524 001003      BNE    12$          ;IS CHARACTER ON ASSOCIATED LINE? NO, ERROR.
8265 035526 005300      DEC    R0            ;COUNT THIS CHARACTER.
8266 035530 001365      BNE    10$          ;TOO MANY RXED? NO, LOOP TO LOOK FOR MORE.
8267 035532 000433      BR    50$          ;YES, REPORT ERROR AND ABORT.
8268
8269      ;*
8269      ; Character received on wrong line.
8270      ; Report "DATA LINE INTERACTIONS ON LINE nn DECIMAL."
8271      ; Error number >>>> 8705 <<<<.

```



HARDWARE TEST

- REP8MP -

8311  
8312  
8313  
8314  
8315  
8316  
8317  
8318  
8319  
8320  
8321 035642  
035642  
8322 000032  
8323 035642 012767 000032 144416  
8324 035650 012767 177777 144406  
8325 035656 016702 144560  
8326 035662 012703 002444  
8327 035666 020203  
8328 035670 001411  
8329  
8330  
8331  
8332  
8333  
8334 035672 012701 011244  
8335 035676  
035676 104455  
035700 022125  
035702 011127  
035704 012620  
8336  
8337 035706 012767 002444 144526  
8338  
8339 035714 005067 144344  
8340 035720  
035720  
035720 104401

```

.SBTTL HARDWARE TEST - REP8MP -
;* *****
;* - Report any BMP codes in the queue -
;* This is a pseudo-test used to report any BMP codes that were found
;* in the DUT's FIFO during previous test, and logged in the BMP code
;* queue.
;* It is unlikely that running this pseudo-test alone will produce any
;* error reports.
;*
;-- *****
      BGNST
                                T26::
      TNUM == TNUM + 1          ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV #TNUM,TSTNUM         ;SET UP THE TEST NUMBER. (93)
      MOV #-1,CTRLCF           ;INDICATE THAT WE ARE IN A TEST.
      MOV BMPCQP,R2            ;GET THE CONTENTS OF THE POINTER.
      MOV #BMPCQB,R3          ;GET THE START ADDRESS OF THE QUEUE.
      CMP R2,R3                ;SEE IF THE POINTER HAS MOVED FROM THE BASE.
      BEQ 60$                  ;EXIT NO CODES IN THE QUEUE.
;*
;* There is at least one BMP code in the queue. Report the error.
;--
      ;Report error BMP CODE FOUND IN TEST nn, BMP CODE:nnnnnn"
      MOV #EM9304,R1           ;PASS THE FIRST MESSAGE TO BE REORTED.
      ERRDF 9301,EM9301,ER9301 ; >>>> ERROR #9301 <<<<<.
                                TRAP C$ERDF
                                .WORD 9301
                                .WORD EM9301
                                .WORD ER9301
      MOV #BMPCQB,BMPCQP      ;SET POINTER BACK TO THE BEGINING OF THE QUE.
      CLR CTRLCF              ;INDICATE THAT WE ARE NOT WITHIN A TEST.
      ENDTST
                                L10054:
                                TRAP C$ETST

```

HARDWARE TEST - REP8MP -

```

8343 ;*****
8344 ;
8345 ;           VDHB.HWQ
8346 ;
8347 ;*****
8349

```

```

8350 .SBTTL  HARDWARE PARAMETER CODING SECTION
8351
8352
8353
8354
8355

```

```

8356 ;**
8357 ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
8358 ; THAT ARE USED BY THE SUPERVISOR TO BUILD P TABLES. THE
8359 ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
8360 ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
8361 ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
8362 ; WITH THE OPERATOR.
8363 ;--

```

```

8364 035722          BGNHRD
      035722 000027
      035724
                                     .WORD L10055-L$HARD/2
                                     L$HARD::

```

```

8365
8375 ;DEVICE CSR ADDRESS QUESTION:
8376 035724          GPRMA  HWPTQ1,0,0,160000,177776,YES
      035724 000031
      035726 036002
      035730 160000
      035732 177776
                                     .WORD  T$CODE
                                     .WORD  HWPTQ1
                                     .WORD  T$LLOLIM
                                     .WORD  T$HILIM

```

```

8377 ;DEVICE INTERRUPT VECTOR QUESTION:
8378 035734          GPRMA  HWPTQ2,2,0,40,776,YES
      035734 001031
      035736 036020
      035740 000040
      035742 000776
                                     .WORD  T$CODE
                                     .WORD  HWPTQ2
                                     .WORD  T$LLOLIM
                                     .WORD  T$HILIM

```

```

8379 ;ACTIVE LINES BIT MAP QUESTION:
8380 035744          GPRMD  HWPTQ3,4,0,MAPLNS,0,177777,YES
      035744 002032
      035746 036053
      035750 000377
      035752 000000
      035754 177777
                                     .WORD  T$CODE
                                     .WORD  HWPTQ3
                                     .WORD  MAPLNS
                                     .WORD  T$LLOLIM
                                     .WORD  T$HILIM

```

```

8381 ;TYPE OF LOOPBACK QUESTION:
8382 035756          GPRMD  HWPTQ4,6,0,377,1,3,YES
      035756 003032
      035760 036101
      035762 000377
      035764 000001
      035766 000003
                                     .WORD  T$CODE
                                     .WORD  HWPTQ4
                                     .WORD  377
                                     .WORD  T$LLOLIM
                                     .WORD  T$HILIM

```

```

8383 ;INTERRUPT BR LEVEL QUESTION:
8384 035770          GPRMD  HWPTQ5,6,0,177400,0,6,YES
      035770 003032
      035772 036157
      035774 177400
      035776 000000
      036000 000006
                                     .WORD  T$CODE
                                     .WORD  HWPTQ5
                                     .WORD  177400
                                     .WORD  T$LLOLIM
                                     .WORD  T$HILIM

```

HARDWARE PARAMETER CODING SECTION

```

8385
8386
8387 036002          ENDHRD
                        .EVEN
                        L10055:
036002
8388
8395
8396 036002      103    123    122      HWPTQ1: .ASCIZ /CSR ADDRESS: /
036005      040    101    104
036010      104    122    105
036013      123    123    072
036016      040    000
8397 036020      111    116    124      HWPTQ2: .ASCIZ /INTERRUPT VECTOR ADDRESS: /
036023      105    122    122
036026      125    120    124
036031      040    126    105
036034      103    124    117
036037      122    040    101
036042      104    104    122
036045      105    123    123
036050      072    040    000
8398 036053      101    103    124      HWPTQ3: .ASCIZ /ACTIVE LINE BIT MAP: /
036056      111    126    105
036061      040    114    111
036064      116    105    040
036067      102    111    124
036072      040    115    101
036075      120    072    040
036100      000
8399 036101      124    131    120      HWPTQ4: .ASCIZ /TYPE OF LOOPBACK (1=INTERNAL,2=H3277,3=H325):/
036104      105    040    117
036107      106    040    114
036112      117    117    120
036115      102    101    103
036120      113    040    050
036123      061    075    111
036126      116    124    105
036131      122    116    101
036134      114    054    062
036137      075    110    063
036142      062    067    067
036145      054    063    075
036150      110    063    062
036153      065    051    072
036156      000
8400 036157      111    116    124      HWPTQ5: .ASCIZ /INTERRUPT BR LEVEL: /
036162      105    122    122
036165      125    120    124
036170      040    102    122
036173      040    114    105
036176      126    105    114
036201      072    040    000
8401
8402          .EVEN
    
```

HARDWARE PARAMETER CODING SECTION

```

8405 ;*****
8406 ;
8407 ;           VDHB.SWQ
8408 ;
8409 ;*****

8411
8412
8413 .SBTTL  SOFTWARE PARAMETER CODING SECTION
8414
8415 ;**
8416 ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
8417 ; THAT ARE USED BY THE SUPERVISOR TO BUILD P TABLES.  THE
8418 ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
8419 ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
8420 ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
8421 ; WITH THE OPERATOR.
8422 ;--
8423
8424 036204          BCNSFT
      036204 000010
      036206                                .WORD L10056-L$SOFT/2
                                           L$SOFT::

8425
8434 ;UNIT NUMBER PRINTOUT QUESTION:
8435 036206          GPRML  SWPTQ1,0,20,YES
      036206 000130                                .WORD  T$CODE
      036210 036226                                .WORD  SWPTQ1
      036212 000020                                .WORD  20

8436 ;NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE QUESTION:
8437 036214          GPRMD  SWPTQ2,2,D,177777,0,177777,YES
      036214 001052                                .WORD  T$CODE
      036216 036302                                .WORD  SWPTQ2
      036220 177777                                .WORD  177777
      036222 000000                                .WORD  T$LOLIM
      036224 177777                                .WORD  T$HILIM

8438
8439          .EVEN
8440
8441 036226          ENDSFT
                                           .EVEN
                                           L10056:

8442
8443
8450 036226          122      105      120      SWPTQ1: .ASCIZ  /REPORT UNIT NUMBER AS EACH UNIT IS TESTED: /
      036231          117      122      124
      036234          040      125      116
      036237          111      124      040
      036242          116      125      115
      036245          102      105      122
      036250          040      101      123
      036253          040      105      101
      036256          103      110      040
      036261          125      116      111
      036264          124      040      111
      036267          123      040      124
      036272          105      123      124
      036275          105      104      072
      036300          040      000

```



SOFTWARE PARAMETER CODING SECTION

8451	036302	116	125	115
	036305	102	105	122
	036310	040	117	106
	036313	040	111	116
	036316	104	111	126
	036321	111	104	125
	036324	101	114	040
	036327	104	101	124
	036332	101	040	105
	036335	122	122	117
	036340	122	123	040
	036343	124	117	040
	036346	122	105	120
	036351	117	122	124
	036354	040	117	116
	036357	040	101	040
	036362	114	111	116
	036365	105	072	040
	036370	000		

SWPTQ2: .ASCIZ /NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE: /

8452

.EVEN

SOFTWARE PARAMETER CODING SECTION

```

8454
8455
8456 :*****
8457 :
8458 :           SKL6.P11
8459 :*****
8460
8461
8462
8463 036372 $PATCH:
8464 036372 .BLKW 24
8465
8472
8473
8474
8475
8476 036442 LASTAD
                                .EVEN
                                .WORD 0
                                .WORD 0
036442 000000
036444 000000
8477 036446 L$LAST:
                                ENDMOD
8478
8479
8480
8481
8482
8483
8484
8485 000001 .END

```





Symbol table

T\$\$CLE= 010020	T1	020126 G	T19	032250 G	T4	021424 G	WAIBIS	016576 G
T\$\$DU = 010021	T10	024534 G	T2	020416 G	T5	022164 G	WAITTX	016652 G
T\$\$HAR= 010055	T11	025226 G	T20	032664 G	T6	022724 G	WORD1	002314 G
T\$\$HW = 010000	T12	025716 G	T21	033300 G	T7	023320 G	WTWLNC	016712 G
T\$\$INI= 010016	T13	026362 G	T22	033714 G	T8	023742 G	WTWLPR	016742 G
T\$\$MSG= 010013	T14	027006 G	T23	034310 G	T9	024232 G	X\$ALWA=	000000
T\$\$PRO= 010015	T15	027506 G	T24	034704 G	UAM	= 000200 G	X\$FALS=	000040
T\$\$RPT= 010014	T16	030434 G	T25	035222 G	UNITN	002240 G	X\$OFFS=	000400
T\$\$SOF= 010056	T17	031134 G	T26	035642 G	UNSDIV	016366 G	X\$TRUE=	000020
T\$\$SW = 010001	T18	031634 G	T3	021004 G	WAIBIC	016522 G	\$PATCH	036372 G
T\$\$TES= 010054								

. ABS. 036446 000 (RW,I,GBL,ABS,OVR)  
 000000 001 (RW,I,LCL,REL,CON)

Errors detected: 0

\*\*\* Assembler statistics

Work file reads: 258  
 Work file writes: 272  
 Size of work file: 35960 Words ( 141 Pages)  
 Size of core pool: 19714 Words ( 75 Pages)  
 Operating system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:05:41.88  
 CVDHBD.BIN,CVDHBD.LST/-SP=SVC40/ML,CVDHBD.P11