IDENTIFICATION

PRODUCT CODE: AC T7968 MC
PRODUCT NAME: CZ0HVBO DHU 11 FUNC T51 PART2
PRODUCT DATE: 3 MARCH 1984
MAINTAINER: ENE - DIAGNOSTICS GROUP
AUTHOR: ANTHONY HART
MODIFIED BY: ANTHONY HART

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) 1984 BY DIGITAL EQUIPMENT CORPORATION
THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL PDP UNIBUS MASSBUS
DEC DECUS DECTAPE
***** MODIFICATION HISTORY *****

ORIGINAL RELEASE: 15 DEC 83  ANTHONY HART
VERSION 80 3 MAR 84  ANTHONY HART

THE FOLLOWING MODIFICATIONS HAVE BEEN MADE TO THE OLD CZDHVA:

THE HARDWARE QUESTION "TYPE OF LOOPBACK" HAS BEEN ALTERED TO INCLUDE THE
STAGGERED LOOPBACK CONNECTORS ON THE DMU11 DISTRIBUTION PANEL (H3029).

THE HARDWARE QUESTION "BR LEVEL" HAS BEEN REMOVED.

THE MODEM SIGNAL TESTS THAT WERE IN THE OLD VERSION (CZDHVAO) HAVE NOW
BEEN REMOVED TO PART CZDHW. THEY WERE TESTS: 16 THRU 23. IN CZDHVAO.

THE FOLLOWING NEW TESTS HAVE BEEN ADDED TO THIS PART:

TEST 13 - RX TIMER REGISTER TEST
TEST 14 - TX ACTION FIFO TEST
TEST 15 - TX_FIFO TEST
## TABLE OF CONTENTS

1.0 GENERAL PROGRAM CONSIDERATIONS  
1.1 PROGRAM ABSTRACT  
1.2 SYSTEM REQUIREMENTS  
1.3 RELATED DOCUMENTS AND STANDARDS  
1.4 DIAGNOSTIC HIERARCHY PREREQUISITES  
2.0 OPERATING INSTRUCTIONS  
2.1 COMMANDS  
2.2 SWITCHES  
2.3 FLAGS  
2.4 EXTENDED COMMAND SYNTAX  
2.4.1 START COMMAND  
2.4.1.1 TESTS SWITCH (/TESTS:<TEST-LIST>)  
2.4.1.2 PASS SWITCH (/PASS:<PASS-CNT>)  
2.4.1.3 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)  
2.4.1.4 END OF PASS SWITCH (/EOP:<INCR>)  
2.4.1.5 EFFECT OF START COMMAND  
2.4.2 RESTART COMMAND  
2.4.2.1 TESTS, PASS, AND FLAGS SWITCHES  
2.4.2.2 UNITS SWITCH (/UNITS:<UNIT-LIST>)  
2.4.2.3 EFFECT OF RESTART COMMAND  
2.4.3 CONTINUE COMMAND  
2.4.3.1 FLAG SWITCH (/FLAGS:<FLAG-LIST>)  
2.4.3.2 EFFECT OF CONTINUE COMMAND  
2.4.4 PROCEED COMMAND  
2.4.4.1 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)  
2.4.4.2 EFFECT OF PROCEED COMMAND  
2.4.5 ADD COMMAND  
2.4.6 EFFECT OF ADD COMMAND  
2.4.7 DROP COMMAND  
2.4.8 EFFECT OF DROP COMMAND  
2.4.9 PRINT COMMAND  
2.4.9.1 EFFECT OF PRINT COMMAND  
2.4.10 DISPLAY COMMAND  
2.4.10.1 EFFECT OF DISPLAY COMMAND  
2.4.11 FLAGS COMMAND  
2.4.11.1 EFFECT OF FLAGS COMMAND  
2.4.12 IFLAGS COMMAND  
2.4.13 ZFLAGS COMMAND  
2.4.14 CONTROL CHARACTERS  
2.5 HARDWARE QUESTIONS  
2.6 SOFTWARE QUESTIONS  
2.7 EXTENDED P-TABLE DIALOGUE  
2.8 QUICK START-UP PROCEDURE (XXDP)  
3.0 ERROR INFORMATION  
3.1 TYPES OF ERROR MESSAGES  
3.2 SPECIFIC ERROR MESSAGES  
4.0 PERFORMANCE AND PROGRESS REPORTS  
5.0 TEST SUMMARIES  
6.0 EXAMPLE ERROR FREE PASS
1.0 GENERAL PROGRAM CONSIDERATIONS

1.1 PROGRAM ABSTRACT

CDMVBO is part of the DHU-11 functional verification test. This part of the test verifies that the major communications 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 XXP*, ACT, APT, SLIDE and PAPER TAPE. For a complete description of the runtime services, refer to the XXP* 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 DHU FVT:

0 UNIBUS processor with at least 32K bytes of memory.
0 DHU boards installed on the UNIBUS.
0 Appropriate program load device supporting XXP*, media or a down line loading system.

1.3 RELATED DOCUMENTS AND STANDARDS

0 XXP* USER'S MANUAL - DESCRIBES THE RUNNING OF DIAGNOSTICS UNDER THE XXP* MONITOR.

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

The processor, the UNIBUS, the system memory, the console terminal and the load media are assumed to have been tested and found working before this program is run.
2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES.
FOR DETAILED INFORMATION, REFER TO THE XDP* USER'S MANUAL (CMQ54).

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 XDP* USER'S MANUAL HAS MORE DETAILS.

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

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.
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'.

<table>
<thead>
<tr>
<th>SWITCH</th>
<th>EFFECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>/TESTS:LIST</td>
<td>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.</td>
</tr>
<tr>
<td>/PASS:DDDDD</td>
<td>Execute DDDD passes (DDDDD = 1 to 64000)</td>
</tr>
<tr>
<td>/FLAGS:FLGS</td>
<td>Set specified flags. See the flags section of this document.</td>
</tr>
<tr>
<td>/EOP:DDDDD</td>
<td>Report End of Pass Message after every DDDD passes only. (DDDDD = 1 to 64000)</td>
</tr>
<tr>
<td>/UNIT:LIST</td>
<td>Test/ADD/DROP only those units specified in the list. List example: /UNIT:0:5:10:12 use units 0, 5, 10, 11, 12 (unit numbers = 0-63)</td>
</tr>
</tbody>
</table>

Example of switch usage:

START/TXTS: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 '/TEST:1-5' instead of '/TESTS:1-5'.

Below is a table that specifies which switches can be used by each command.

<table>
<thead>
<tr>
<th>TESTS</th>
<th>PASS</th>
<th>FLAGS</th>
<th>EOP</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RESTART</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CONTINUE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCEED</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DROP</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRINT</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISPLAY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLAGS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZFLAGS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>FLAG</th>
<th>EFFECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOE</td>
<td>Halt on error - control is returned to runtime services command mode</td>
</tr>
<tr>
<td>LOE</td>
<td>Loop on error</td>
</tr>
<tr>
<td>IER*</td>
<td>Inhibit all error reports</td>
</tr>
<tr>
<td>IBR*</td>
<td>Inhibit all error reports except first level (first level contains error type, number, pc, test and unit)</td>
</tr>
<tr>
<td>IXR*</td>
<td>Inhibit extended error reports (those called by printx macro's)</td>
</tr>
<tr>
<td>PRI</td>
<td>Direct messages to line printer</td>
</tr>
<tr>
<td>PNT</td>
<td>Print test number as test executes</td>
</tr>
<tr>
<td>BOE</td>
<td>&quot;Bell&quot; on error</td>
</tr>
<tr>
<td>UAM</td>
<td>Unattended mode (no manual intervention)</td>
</tr>
<tr>
<td>ISR</td>
<td>Inhibit statistical reports (does not apply to diagnostics which do not support statistical reporting)</td>
</tr>
<tr>
<td>IDR</td>
<td>Inhibit program dropping of units</td>
</tr>
<tr>
<td>ADR</td>
<td>Execute automap code</td>
</tr>
<tr>
<td>LOT</td>
<td>Loop on test</td>
</tr>
<tr>
<td>EVL</td>
<td>Execute evaluation (on diagnostics which have evaluation support)</td>
</tr>
</tbody>
</table>

*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
2.4 EXTENDED COMMAND SYNTAX

2.4.1 START COMMAND -

START/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:
  <FLAG-LIST>/END:<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 (1:5:10 ETC.), SEPARATED 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>>) -

<FALG-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.
  Beo Bell On Error (Not Related To Bell Prompting).
  Uam Run In Unattended Mode, Bypassing Manual
    Intervention (Illegal For This Diagnostic).
  Isr Inhibit Statistical Reports.
IDU INHIBIT DROPPING OF UNITS BY DIAGNOSTIC.
(HAS NO EFFECT IN THIS 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:<INCR>)

<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:MOE:1

THIS COMMAND WILL CAUSE THREE PASSES TO BE MADE, WITH EACH PASS
CONSIDERING 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.
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 dialogue. 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 units switch should not be used with this program. 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 end error flag set, or c) a control/c was entered by the operator.

2.4.3 CONTINUE COMMAND -

CONTINUE/PASS:<PASS-CNT>/FLAGS:<FLAG-LIST>

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

******************************************************************************
PROCEED)/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.

2.4.7 DROP COMMAND

******************************************************************************
DROP(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 ENTERED. THIS COMMAND
MUST BE FOLLOWED BY A RESTART OR A CONTINUE COMMAND.

2.4.9 PRINT COMMAND

2.4.9.1 EFFECT OF PRINT COMMAND -
THE TOTAL NUMBER OF ERRORS FOR EACH UNIT SINCE THE LAST
START OR RESTART COMMAND ARE PRINTED. THE ISR (INHIBIT
STATISTICAL REPORTING) FLAG IS CLEARED.

2.4.10 DISPLAY COMMAND -

2.4.10.1 EFFECT OF DISPLAY COMMAND -
THE HARDWARE P-TABLE FOR THE TEST STATION IS PRINTED IN THE
FORMAT IN WHICH IT WAS ENTERED.

2.4.11 FLAGS COMMAND -

2.4.11.1 EFFECT OF FLAGS COMMAND -
THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.
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 TELETYPING OUTPUT TO BE
SUPPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR
UNTIL ANOTHER CONTROL/O IS TYPED, WHICH RESTORES
NORMAL TELETYPING 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 XDP+ 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 DMU-11. THE DEFAULT ANSWER FOR THIS QUESTION IS ADDRESS 160460 (OCTAL).


3. ACTIVE LINES BIT MAP - THIS QUESTION REQUESTS AN OCTAL BIT MAP OF THE SERIAL COMMUNICATION LINES ON THE DMU-11 WHICH ARE BEING SELECTED FOR TESTING. IF THE BIT IN THE BIT MAP IS SET WHICH CORRESPONDS TO A PARTICULAR LINE (I.E. BIT 5 FOR LINE 5) THAT LINE WILL BE TESTED BY THE FVT.

4. TYPE OF LOOPBACK (1-INTERNAL, 2-H3029 OR H3277) - THIS QUESTION REQUESTS THE TYPE OF LOOPBACK TO BE USED WHEN TESTING THE DMU-11. THE FOLLOWING TYPES ARE SUPPORTED:
   a. INTERNAL - ONLY INTERNAL UART LOOPBACK IS TO BE USED IN TESTING THE DMU-11.
   b. H3029 OR H3277 - STAGGERED LOOPBACK CONNECTORS ARE PROVIDED ON THE DMU11 DISTRIBUTION PANEL (H3029). IF THIS DISTRIBUTION PANEL IS NOT PRESENT THEN H3277 STAGGERED BERG CONNECTOR(S) MUST BE INSTALLED ON THE BERG CONNECTOR SOCKETS OF THE DMU11.
   c. H325 - SINGLE LINE, 25 PIN LOOPBACK CONNECTORS (TYPE H325) ARE INSTALLED ON THE LINES TO BE TESTED.
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 software questions and the default values are described in the next paragraph(s).

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 THAT UNIT.

2. EXTENDED ERROR REPORTING - THIS QUESTION ASKS WHETHER EXTENDED ERROR INFORMATION IS REQUIRED OTHER THAN THE “TEST FAILED” MESSAGE, ON EACH ERROR REPORTED. THE DEFAULT IS “NO” I.E. ONLY A MESSAGE REPORTING THE FACT THAT THE TEST FAILED WILL BE PRINTED.

3. NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE - THIS QUESTION IS ASKED ONLY IF THE PREVIOUS QUESTION WAS ANSWERED “YES”. THE 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.
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 REPEITIONOUS.

TO ILLUSTRATE A MORE EFFICIENT METHOD, SUPPOSE YOU ARE TESTING A FICTICAL DEVICE, THE X/11. 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 X/11 WITH EIGHT UNITS.

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) 2 = <CR>

UNIT 4
CSR ADDRESS (0) = 160000<CR>
SUB-DEVICE # (0) = 3<CR>
Q-FACTOR (0) 3 = <CR>

UNIT 5
CSR ADDRESS (0) = 160000<CR>
SUB-DEVICE # (0) = 4<CR>
Q-FACTOR (0) 4 = <CR>

UNIT 6
CSR ADDRESS (0) = 160000<CR>
SUB-DEVICE # (0) = 5<CR>
Q-FACTOR (0) 5 = <CR>

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

UNIT 8
CSR ADDRESS (0) = 160000<CR>
SUB-DEVICE # (0) = 7<CR>
Q-FACTOR (0) 7 = <CR>
UNIT 8
CSR ADDRESS (O)  160000<CR>
SUB-DEVICE @ (O) ? 7<CR>
Q-FACTOR (O) 1 ? <CR>

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 (O) ? 8<CR>

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

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

UNIT 7
CSR ADDRESS (O)  160000<CR>
SUB-DEVICE @ (O) ? 6,7<CR>
Q-FACTOR (O) 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 (O) ? 8<CR>

UNIT 1
2.8 QUICK START-UP PROCEDURE (XXDP+)

TO START UP THIS PROGRAM:

1. BOOT XXDP+

2. GIVE THE DATE AND ANSWER THE LSI/UNIBUS AND 50HZ (IF THERE IS A CLOCK) QUESTIONS. NOTE, NOT ALL VERSIONS OF XXDP+ ASK FOR THE CLOCK FREQUENCY.

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 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 "IBR2" 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 SPECIFIC ERROR MESSAGES

THIS PROGRAM IS INTENDED TO PROVIDE A GO/NOGO INDICATION OF THE FUNCTIONALITY OF THE DHU-11 BOARDS. TO EXECUTE THE PROGRAM IN THIS MODE THE OPERATOR NEEDS ONLY ANSWER THE "EXTENDED ERROR REPORTING" SOFTWARE QUESTION WITH "NO". THE PROGRAM WILL THEN ONLY PRINT THE NAME OF THE FAILING TEST, THE TEST AND ERROR NUMBERS. FOR A LIST OF THE TEST NAMES IN THIS PROGRAM SEE THE TEST SUMMARY SECTION OF THIS DOCUMENT. AN EXAMPLE OF SUCH AN ERROR MESSAGE IS THE FOLLOWING:

CZDHV DVC FTL ERR 04106 ON UNIT 00 TST 003 SUB 000 PC: XXXXXX
DMA_ABORT BIT TEST FAILED

THIS ERROR INDICATES THAT A FATAL ERROR WAS ENCOUNTERED DURING THE TEST WHICH TESTS THE DMA_ABORT BIT.

IF THE OPERATOR HAD REQUESTED EXTENDED ERROR REPORTING THE SAME ERROR WOULD BE REPORTED AS FOLLOWS:

CZDHV DVC FTL ERR 04106 ON UNIT 00 TST 003 SUB 000 PC: XXXXXX
DMA_ABORT BIT TEST FAILED
DMA_START BIT FOUND SET AFTER DMA ABORTED ON LINE: 8

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.
5.0 TEST SUMMARIES

THE FOLLOWING ARE INCLUDED WITHIN CZDMV8:

1. DEVICE REGISTER ACCESS TEST - VERIFIES THAT THE UUT REGISTERS
   WILL RESPOND WITH THE CORRECT UNIBUS HANDSHAKING SIGNALS.
   VERIFIES THAT THE UUT IS AT THE CORRECT ADDRESS.

2. DMA START TEST - VERIFIES THAT EACH DMA START BIT WILL
   INITIATE A DMA TRANSMISSION ON A LINE

3. DMA ABORT TEST - VERIFIES THAT EACH DMA ABORT BIT WILL
   STOP A DMA TRANSMISSION, RETURN A TX_ACTION AND SUCCESSFULLY
   RESTART THE DMA.

4. DMA ERROR TEST - VERIFIES THAT THE DMA ERROR BIT IN THE CSR
   REPORTS DMA ERRORS CORRECTLY WHEN THEY OCCUR.

5. 0.AUTO INACTIVE TEST - VERIFIES THAT THE DUT WILL NOT RESPOND
   TO INCOMING XON AND XOFF CHARACTERS WHEN 0.AUTO IS CLEAR.

6. 0.AUTO ACTIVE TEST - VERIFIES THAT THE DUT RESPONDS CORRECTLY
   TO INCOMING FLOW CONTROL CHARACTERS WHEN ACTIVE

7. I.AUTO INACTIVE TEST - VERIFIES THAT THE DUT WILL NOT GENERATE
   XON AND XOFF CHARACTERS IN RESPONSE TO THE APPROPRIATE FIFO
   CONDITIONS WHEN I.AUTO IS INACTIVE.

8. I.AUTO ACTIVE TEST - VERIFIES THAT THE DUT WILL GENERATE
   XON AND XOFF CHARACTERS IN RESPONSE TO THE APPROPRIATE FIFO
   CONDITIONS WHEN I.AUTO IS ACTIVE.

9. FIFO DATA TEST - VERIFIES THAT THE FIFO WILL HOLD 256
   CHARACTERS WITHOUT CORRUPTING DATA.

10. FIFO 3/4 LEVEL INACTIVE TEST - VERIFIES THAT THE 3/4 LEVEL
    ALARM DOES NOT BECOME ACTIVE BELOW THE 3/4 LEVEL.

11. FIFO 3/4 LEVEL ACTIVE TEST - VERIFIES THAT THE 3/4 LEVEL
    ALARM BECOMES ACTIVE WHEN THE FIFO IS 3/4 FULL.

12. FIFO 3/4 LEVEL ACTIVE/INACTIVE TEST - VERIFIES THAT THE 3/4
    LEVEL ALARM, ONCE ACTIVATED, REMAINS ACTIVE UNTIL THE FIFO
    IS REDUCED BELOW THE 1/2 LEVEL.

13. FIFO 1/2 LEVEL TEST - VERIFIES THAT THE FIFO 1/2 LEVEL
    ALARM SYSTEM BECOMES ACTIVE AND INACTIVE AT THE CORRECT
    LEVELS.

14. RX TIMER TEST - VERIFIES THAT THE HOLD OFF TIMER FOR
    RX INTERRUPTS IS OPERATING CORRECTLY, AND THAT THE 3/4 FULL
    LEVEL OVERIDES THE TIMER.
15. **TX ACTION FIFO TEST** - VERIFIES THAT THE TX ACTION FIFO CAN HOLD 16 UNIQUE TX-ACTIONS, AND THAT ONLY ONE TX INTERRUPT OCCURS FOR ALL 16 TX-ACTIONS.

16. **TX FIFO TEST** - VERIFIES THAT THE FIFO WILL 64 UNIQUE CHARACTERS AND ALSO THAT ONLY ONE INTERRUPT OCCURS FOR ALL 64 CHARACTERS.

17. **BREAK GENERATION TEST** - VERIFIES THAT ALL SERIAL TRANSMIT LINES CAN GENERATE A BREAK BY SETTING THE BRK BIT IN THE ASSOCIATED LINCTRL REGISTER.

18. **NO OVERRUN ERROR TEST** - VERIFIES THAT THE DUT WILL NOT REPORT DATA OVERRUN ERRORS WHEN THEY DO NOT OCCUR.

19. **OVERRUN ERROR TEST** - VERIFIES THAT THE DUT WILL REPORT DATA OVERRUN ERRORS WHEN THEY OCCUR.

20. **REPORT BMP CODES TEST** - THIS PSEUDO TEST REPORTS THE FIRST 32 CHARACTERS WHICH WERE DISCOVERED IN THE FIFO DURING THE EXECUTION OF THE OTHER TESTS. THIS AVOIDS INTERRUPTION OF THE OTHER TESTS BY THESE CODES IF THEY ARE NOT CRITICAL TO THE PERFORMANCE OF THE TESTS.
6.0 EXAMPLE ERROR FREE PASS

THE FOLLOWING IS AN EXAMPLE OF AN ERROR FREE PASS DIALOGUE:

.R CZDMVBO
CZDMVBO.BIN
DRS
CZOHV-A-0
DMU-11 FUNC TSI PART2
UNIT IS DMU-11
RESTRT ADDR: 147670
DR> STA/PAS: 1

CHANGE HW (L) ? Y

0 UNITS (D) ? 2

UNIT 0
CSR ADDRESS: (O) 160460 ? T2

UNIT 1
CSR ADDRESS: (O) 160460 ? 160500
INTERUPT VECTOR ADDRESS: (O) 310 ? 520
ACTIVE LINE BIT MAP: (O) 177777 ? <CR>
TYPE OF LOOPBACK (1=INTERNAL, 2=H3029 OR H3277): (O) 2 ? 1

CHANGE SW (L) ? Y

REPORT UNIT NUMBER AS EACH UNIT IS TESTED: (L) Y ? <CR>

TESTING UNIT: 0
TESTING UNIT: 1
CZOHV EOP 1
0 TOTAL ERRS

DR>

&
LIST SEQ, LOC, BIN, MEB
NLIST CND

SBTL PROGRAM HEADER

SCALL SVC ; INITIALIZE SUPERVISOR MACROS

; IF STRUCTURED MACROS ARE TO BE USED, ADD "SCALL STRUCT" AND "STRUCT"
; TO INITIALIZE THE STRUCTURED MACROS.

SVCINS = 1 ; LIST INSTRUCTIONS, SHIFTED RIGHT
SVCST = 1 ; LIST TEST TAGS, SHIFTED RIGHT
SVCSTB = 1 ; LIST SUBTEST TAGS, SHIFTED RIGHT
SVCGLBL = 1 ; LIST GLOBAL TAGS, SHIFTED RIGHT
SVCSTG = 1 ; LIST OTHER TAGS, SHIFTED RIGHT

; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.

ENABL ABS
ENABL AMA
. BGNMOD
. = 2000

;* THE PROGRAM HEADER IS THE INTERFACE BETWEEN
; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
;*

POINTER BGNRPT, BGNSW, BGNSFT, BGNDU, ERRTBL

HEADER CZOHV, B0, 22, 0, PRI07

NAME: .ASCII /C/
NAME: .ASCII /Z/
NAME: .ASCII /D/
NAME: .ASCII /H/
NAME: .ASCII /V/
. BYTE 0
. BYTE 0
. BYTE 0

REV: .ASCII /B/
REPO: .ASCII /O/
UNIT: .WORD 0
TIMEI: .
**DISPATCH TABLE**

```
;**
; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
;
; DISPATCH 20

002122 000024
002124 021022
002126 021304
002130 021710
002132 022346
002134 022674
002136 023474
002140 024274
002142 024716
002144 025410
002146 025716
002150 026226
002152 026726
002154 027424
002156 030102
002160 031152
002162 031740
002164 032766
002166 033412
002170 034130
002172 035074

```

.SBTTL DEFAULT HARDWARE P TABLE

**
: THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
: THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
: IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P TABLES.
: AND IS USED AS A "TEMPLATE" FOR BUILDING THE P-TABLES.
:

BGNHW DFPTBL

.LIMW:: .WORD L10000 L#Mw/2

DFPTBL::

.LIMW::

1159
1160 002174 002177 000004
1161
1162 002176 160460
1163 002200 000310
1164 002202 177777
1165 002204 002
1166
1167 002206 002206

.ENDHW

.L10000:
SOFTWARE P TABLE

; THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
; PROGRAM AS OPERATIONAL PARAMETERS. THESE PARAMETERS ARE
; SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
; AT RUN TIME.

BGNSW SFPTBL

L1SW:: .WORD L10001-L1SW/2
SFPTBL::

OPTION:: .WORD 20 ; BIT MAP OF PROGRAM CONTROL FLAGS
NDERPT:: .WORD 0 ; DEFAULT NUMBER OF INDIVIDUAL DATA ERRORS TO RPT.
.SBTTL GLOBAL EQUATES SECTION

!**
! THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
! ARE USED IN MORE THAN ONE TEST.
!**

000020 NUMLNS=20 ; NUMBER OF LINES ON DMU11 IS 8.
177777 MAPLNS=177777 ; BIT MAP OF LINES ON DMU11.

;****** DEVICE REGISTER OFFSETS FROM THE CSR'S ADDRESS *******

000000 CSR0=0 ; CSR REGISTER OFFSET FROM THE CSR ADDRESS
000002 ABUF0=2 ; RECEIVE REGISTER OFFSET FROM THE CSR ADDRESS
000002 RXTIM0=2 ; RECEIVE TIMER REGISTER OFFSET FROM THE CSR ADDRESS
000004 LPRO=4 ; LINE PARAMETER REGISTER OFFSET FROM THE CSR ADDRESS
000006 FSL0=6 ; FIFO SIZE/STATUS REGISTER OFFSET FROM THE CSR ADDRESS
000006 FDAT0=6 ; FIFO DATA REGISTER OFFSET FROM THE CSR ADDRESS
000010 LNCST0=10 ; LINE CONTROL REGISTER OFFSET FROM THE CSR ADDRESS
000012 TXAD10=12 ; TRANSMIT ADDRESS 1 REGISTER OFFSET FROM THE CSR ADDRESS
000014 TXADD2=14 ; TRANSMIT ADDRESS 2 REGISTER OFFSET FROM THE CSR ADDRESS
000016 TXBCD=16 ; TRANSMIT COUNT REGISTER OFFSET FROM THE CSR ADDRESS

;****** EQUATES USED WITH RESPECT TO THE RX BUFFER *******

000020 RXBETX=16 ; LEVEL OF RX BUFFER AT WHICH TO RE-ENABLE TRANSMISSION.
000030 RXBDTX=24 ; LEVEL OF RX BUFFER AT WHICH TO DISABLE TRANSMISSION.
000100 RXBFUL=64 ; TOTAL CHARACTER CAPACITY OF THE RX BUFFER.

100000 BIT15= 100000
040000 BIT14= 400000
020000 BIT13= 200000
010000 BIT12= 100000
004000 BIT11= 400000
002000 BIT10= 200000
001000 BIT9= 100000
000400 BIT8= 400000
000200 BIT7= 200000
000100 BIT6= 100000

001000 BIT5= BIT09
000400 BIT4= BIT08
000200 BIT3= BIT07
000100 BIT2= BIT06
EVENT FLAG DEFINITIONS

EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

START COMMAND WAS ISSUED
RESTART COMMAND WAS ISSUED
CONTINUE COMMAND WAS ISSUED
A NEW PASS HAS BEEN STARTED
A POWER FAIL/POWER-UP OCCURRED

PRIORITY LEVEL DEFINITIONS

OPERATOR FLAG BITS

1257
.SBTTL GLOBAL DATA SECTION

***
THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
IN MORE THAN ONE TEST.
***

******************************************************************************
UNIT VARIABLE AREA
******************************************************************************

RXVECA:: .WORD 200  RX VECTOR ADDRESS.
TXVECA:: .WORD 204  TX VECTOR ADDRESS.
ACTLNS:: .WORD 177777  ACTIVE LINE BIT MAP.
LOPBACK:: .BYTE 0  LOOPTBACK MODE
BRLEVEL:: .BYTE 4  INTERRUPT BUS REQUEST LEVEL
UNITN:: .WORD 0  UNIT NUMBER.

******************************************************************************
DEVICE REGISTER ADDRESS TABLE
******************************************************************************

CSRA:: .WORD 160020  DMU-11 CSR ADDRESS.
RXMA:: .WORD 160022  DMU-11 RECEIVE BUFFER/TIMER ADDRESS.
LPRA:: .WORD 160024  DMU-11 LINE PARAMETER REGISTER ADDRESS.
FDATA:: .WORD 160026  DMU-11 FIFO SIZE/LINE STATUS REGISTER ADDRESS.
                   AND FIFO DATA REGISTER ADDRESS.
LCRTRA:: .WORD 160030  DMU-11 LINE CONTROL REGISTER ADDRESS.
TRANDA:: .WORD 160032  DMU-11 TRANSMIT BUFFER 1 REGISTER ADDRESS
TRABA:: .WORD 160034  DMU-11 TRANSMIT BUFFER 2 REGISTER ADDRESS
TXBCRA:: .WORD 160036  DMU-11 TRANSMIT BUFFER COUNT REGISTER ADDRESS

******************************************************************************
ASSORTED GLOBAL VARIABLES
******************************************************************************

BUFFPTR:: .WORD 0  STORAGE FOR RECEIVE CHARACTER BUFFER POINTER.
CTRLCF:: .WORD 0  STORAGE FOR THE CONTROL-C FLAG.
EXQERR:: .WORD 0  "EXIT ON ERROR" FLAG.
TEST:: .WORD 1  STORAGE FOR THE TEST NUMBER.
IESTAT:: .WORD 0  STORAGE FOR STATES OF THE OUT INT ENABLE BITS.
LGPR1:: .WORD 31463  BIT MAP OF LINES IN LINE GROUP I
LGPR2:: .WORD 146314  BIT MAP OF LINES IN LINE GROUP II
PASOMET:: .WORD 0  STO'G FOR PASS COUNT USED IN ROM VERSION TST.
RXINT:: .WORD 0  STORAGE FOR RECEIVER INTERRUPT FLAGS.
TXINT:: .WORD 0  STORAGE FOR TRANSMITTER INTERRUPT FLAGS.
INT:: .WORD 0  STORAGE FOR TRANSMITTER INTERRUPT COUNT.
TPAFLG:: .WORD 0  FLAGS SET WHEN AN EXPECTED O04 TRAP OCCURS.
WORD1:: .WORD 0  LOCATION FOR PASS INDIRECT PARAMETERS.

******************************************************************************
LINE VARIABLES AND STORAGE
******************************************************************************

CLKCSR:: .WORD 177546  CSR ADDRESS OF THE LTC.
CLKBR:: .WORD PRIO6  INTERRUPT PRIORITY LEVEL OF THE LTC.
CLKVEC: .WORD 100 ; INTERRUPT VECTOR ADDRESS OF THE LTC.
CLKFREQ: .WORD 60 ; INTERRUPT FREQUENCY OF THE LTC.
TIMERR: .WORD 0 ; HARDWARE CLOCK COUNTER 01.
HARDWARE CLOCK COUNTER 02.
TIMERSL: .WORD 0 ; HARDWARE BREAK COUNTER LOCATION.
BREAK COUNT VALUE IN CLOCK TICS.
MS: .WORD 0 ; NUMBER OF MILLI-SECONDS PER LTC TICK.
MSCNT: .WORD 62 ; LOOP COUNT (USED BY MSLOOP) TO DELAY 1 MS.

**MEMORY MANAGEMENT VARIABLES AND FLAGS.**

MMRS: .WORD 177572 ; ADDRESS OF MEM MGT STATUS REGISTER 00.
MMRPE: .WORD 0 ; MEM MGT PRESENT FLAG (0 IF MM NOT PRESENT).
MMRENABLE: .WORD 0 ; MEM MGT ENABLED FLAG (0 IF MM NOT ENABLED).

**TABLE OF WORDS WITH CORRESPONDING BIT SET FOR GENERATION OF BIT MAPS.**

BITBL: .WORD 1 ; BIT 0 SET.
.WORD 2 ; BIT 1 SET.
.WORD 4 ; BIT 2 SET.
.WORD 10 ; BIT 3 SET.
.WORD 20 ; BIT 4 SET.
.WORD 40 ; BIT 5 SET.
.WORD 100 ; BIT 6 SET.
.WORD 200 ; BIT 7 SET.
.WORD 400 ; BIT 8 SET.
.WORD 1000 ; BIT 9 SET.
.WORD 2000 ; BIT 10 SET.
.WORD 4000 ; BIT 11 SET.
.WORD 10000 ; BIT 12 SET.
.WORD 20000 ; BIT 13 SET.
.WORD 40000 ; BIT 14 SET.
.WORD 100000 ; BIT 15 SET.

**GPR SAVE AREAS ZERO AND ONE.**

GPRSIZERO: .WORD 0 ; BASE OF GPR SAVE AREA NUMBER ZERO.
.WORD 1 ; STORAGE FOR R1.
.WORD 2 ; STORAGE FOR R2.
.WORD 3 ; STORAGE FOR R3.
.WORD 4 ; STORAGE FOR R4.
.WORD 5 ; STORAGE FOR R5.

**STORAGE AREA FOR THE BMP CODE QUEUE.**

BMPQUEUE: .WORD 0 ; POINTER USED TO ACCESS THE NEXT CELL IN QUEUE.
.BLK 64 ; STORAGE FOR 32 CELLS. TEST PLUS BMP CODE.
.BLK 64 ; LAST ADDRESS PLUS 2 OF THE BMP CODE QUEUE.

**STORAGE AREA FOR ERROR SUMMARY TABLE AND FLAGS.**

ERRORSMP: .WORD 0 ; ERROR SUMMARY FLAGS.
.ERRCNTB: .BLK 16 ; TABLE OF ERROR COUNTS.
GLOBAL DATA SECTION

BUFFAS: BASE OF MEMORY BUFFER.
ERLTLB: FIRST HALF OF GENERAL TABLE OR BUFFER.
DUFND: SECOND HALF OF GENERAL TABLE OR BUFFER.
BUFFQI: LAST QUARTER OF THE BUFFER AREA.
BUNDEND: END OF GENERAL PURPOSE MEMORY BUFFER.
ENDTABLE: BUFFER OVERFLOW SPACE.

RECEPTION TABLE OF COUNTERS
RXCNTR: RECEPTION CHARACTER COUNTERS TABLE.

* TABLE FOR STORAGE OF RX/TX LINE NUMBERS ASSOCIATIONS.
* THE ASSOCIATIONS ARE STORED AS LINE NUMBER TIMES 2 FOR USE AS OFFSETS.
* WHEN ACCESSING A TABLE OF WORDS.
* NOTE: DO NOT WRITE A NON-ZERO VALUE INTO THE UPPER BYTE OF ANY ENTRY.

TXRXLB: BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.

WORD 0: TX/RX LINE OFFSET FOR RX/TX LINE 0.
WORD 1: TX/RX LINE OFFSET FOR RX/TX LINE 1.
WORD 2: TX/RX LINE OFFSET FOR RX/TX LINE 2.
WORD 3: TX/RX LINE OFFSET FOR RX/TX LINE 3.
WORD 4: TX/RX LINE OFFSET FOR RX/TX LINE 4.
WORD 5: TX/RX LINE OFFSET FOR RX/TX LINE 5.
WORD 6: TX/RX LINE OFFSET FOR RX/TX LINE 6.
WORD 7: TX/RX LINE OFFSET FOR RX/TX LINE 7.
WORD 8: TX/RX LINE OFFSET FOR RX/TX LINE 8.
WORD 9: TX/RX LINE OFFSET FOR RX/TX LINE 9.
WORD 10: TX/RX LINE OFFSET FOR RX/TX LINE 10.
WORD 11: TX/RX LINE OFFSET FOR RX/TX LINE 11.
WORD 12: TX/RX LINE OFFSET FOR RX/TX LINE 12.
WORD 13: TX/RX LINE OFFSET FOR RX/TX LINE 13.
WORD 14: TX/RX LINE OFFSET FOR RX/TX LINE 14.
WORD 15: TX/RX LINE OFFSET FOR RX/TX LINE 15.

END OF TX/RX LINE NUMBER ASSOCIATION TABLE.

TABLE FOR STORAGE OF RX/TX LINE NUMBERS WHICH CAN BE USED AS SUCH OR OFFSETS WHEN ACCESSING A TABLE OF BYTES.

TXRLNB: BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.

BYTE 0: TX/RX LINE FOR RX/TX LINE 0.
BYTE 1: TX/RX LINE FOR RX/TX LINE 1.
BYTE 2: TX/RX LINE FOR RX/TX LINE 2.
BYTE 3: TX/RX LINE FOR RX/TX LINE 3.
BYTE 4: TX/RX LINE FOR RX/TX LINE 4.
BYTE 5: TX/RX LINE FOR RX/TX LINE 5.
BYTE 6: TX/RX LINE FOR RX/TX LINE 6.
BYTE 7: TX/RX LINE FOR RX/TX LINE 7.
BYTE 8: TX/RX LINE FOR RX/TX LINE 8.
BYTE 9: TX/RX LINE FOR RX/TX LINE 9.
GLOBAL DATA SECTION

1437 004022 012 .BYTE 10.
1438 004023 013 ;TX/RX LINE FOR RX/TX LINE 10.
1439 004024 014 .BYTE 11.
1440 004025 015 ;TX/RX LINE FOR RX/TX LINE 11.
1441 004026 016 .BYTE 12.
1442 004027 017 ;TX/RX LINE FOR RX/TX LINE 12.
1443 004028 018 .BYTE 13.
1444 004029 019 ;TX/RX LINE FOR RX/TX LINE 13.
1445 00402a 01a .BYTE 14.
1446 00402b 01b ;TX/RX LINE FOR RX/TX LINE 14.
1447 00402c 01c .BYTE 15.
1448 00402d 01d ;TX/RX LINE FOR RX/TX LINE 15.
1449 00402e 01e TXRLINE::
1450 00402f 01f .EVEN
1451 004030 020 ;END OF TX/RX LINE NUMBER ASSOCIATION TABLE.
1452 004031 021 **********************************************
1453 004032 022 ;* TABLE OF TX/RX LINE NUMBER ASSOCIATIONS IN STAGGERED LOOPBACK.
1454 004033 023 ;* THE ASSOCIATIONS ARE STORED AS LINE NUMBER TIMES 2 FOR USE AS OFFSETS
1455 004034 024 ;* WHEN ACCESSING A TABLE OF WORDS.
1456 004035 025 ;* THIS IS A TABLE OF DATA FOR READING ONLY. USE TO LOAD THE ABOVE TABLE.
1457 004036 026 ;* NOTE: MUST CONVERT FROM BYTES TO WORDS WHEN LOADING ABOVE TABLE.
1458 004037 027 **********************************************
1459 004038 028 STGTRB::
1460 004039 029 ;BASE OF STAGGERED TX/RX LINE NUMBER TABLE.
1461 00403a 02a .BYTE 4.
1462 00403b 02b ;TX/RX LINE OFFSET FOR RX/TX LINE 0.
1463 00403c 02c .BYTE 6.
1464 00403d 02d ;TX/RX LINE OFFSET FOR RX/TX LINE 1.
1465 00403e 02e .BYTE 0.
1466 00403f 02f ;TX/RX LINE OFFSET FOR RX/TX LINE 2.
1467 004040 030 .BYTE 2.
1468 004041 031 ;TX/RX LINE OFFSET FOR RX/TX LINE 3.
1469 004042 032 .BYTE 12.
1470 004043 033 ;TX/RX LINE OFFSET FOR RX/TX LINE 4.
1471 004044 034 .BYTE 14.
1472 004045 035 ;TX/RX LINE OFFSET FOR RX/TX LINE 5.
1473 004046 036 .BYTE 8.
1474 004047 037 ;TX/RX LINE OFFSET FOR RX/TX LINE 6.
1475 004048 038 .BYTE 10.
1476 004049 039 ;TX/RX LINE OFFSET FOR RX/TX LINE 7.
1477 00404a 03a .BYTE 20.
1478 00404b 03b ;TX/RX LINE OFFSET FOR RX/TX LINE 8.
1479 00404c 03c .BYTE 22.
1480 00404d 03d ;TX/RX LINE OFFSET FOR RX/TX LINE 9.
1481 00404e 03e .BYTE 16.
1482 00404f 03f ;TX/RX LINE OFFSET FOR RX/TX LINE 10.
1483 004050 040 .BYTE 28.
1484 004051 041 ;TX/RX LINE OFFSET FOR RX/TX LINE 11.
1485 004052 042 .BYTE 26.
1486 004053 043 ;TX/RX LINE OFFSET FOR RX/TX LINE 12.
1487 004054 044 .BYTE 30.
1488 004055 045 ;TX/RX LINE OFFSET FOR RX/TX LINE 13.
1489 004056 046 .BYTE 24.
1490 004057 047 ;TX/RX LINE OFFSET FOR RX/TX LINE 14.
1491 004058 048 .BYTEF 26.
1492 004059 049 ;TX/RX LINE OFFSET FOR RX/TX LINE 15.
1493 00405a 04a .EVEN
1494 00405b 04b ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
1495 00405c 04c LERRTBL::
1496 00405d 04d ERRATYP::
1497 00405e 04e .WORD 0
1498 00405f 04f 000000
1499 004060 050 .WORD 1
1500 004061 051 000000
1501 004062 052 .WORD 0
1502 004063 053 000000
1503 004064 054 .WORD 0
1504 004065 055 000000
1505 004066 056 .WORD 0
1506 004067 057 .EVEN
SBTTL  GPR HANDLING ROUTINES FOR SUBROUTINE CALLS.

1487*****************************************************************
1488* THERE ARE 4 ROUTINES AND MACRO DEFINITIONS USED FOR THE HANDLING OF  *
1489* GPR VALUES DURING SUBROUTINE CALLS WITHIN THIS PROGRAM. THE FOUR  *
1490* ROUTINES/MACRO CALLS HAVE THE FOLLOWING NAMES:                *
1491* * SAVE - MACRO DEFINITION USED AT THE BEGINNING OF A SUBROUTINE TO  *
1492* * SAVE THE GPR CONTENTS FOR LATER RESTORATION.              *
1493* * PASS - MACRO DEFINITION USED AT THE END OF A SUBROUTINE TO RESTORE  *
1494* * THE PREVIOUSLY SAVED GPR CONTENTS AND TO LEAVE THE CONTENTS  *
1495* * OF THE SPECIFIED GPR(S) INTACT (NOT RESTORED).             *
1496* * PREGOS - SUBROUTINE WHICH IS CALLED FROM THE SAVE AND PASS MACRO  *
1497* * EXPANSIONS WHICH ACTUALLY PERFORMS THE ACTIONS ON THE GPRS.   *
1498* * * DURING A SUBROUTINE WHICH USES THESE GPR SAVE ROUTINES THE VALUES  *
1499* * OF THE GPRS ARE STORED ON THE STACK IN THE FOLLOWING STACK FRAME:  *
1500* * SP  -> RET PC INTO PREGOS ROUTINE.                        *
1501* * SP+2 -> GPR R0 CONTENTS.                                  *
1502* * SP+4 -> GPR R1 CONTENTS.                                  *
1503* * SP+6 -> GPR R2 CONTENTS.                                  *
1504* * SP+8 -> GPR R3 CONTENTS.                                  *
1505* * SP+10 -> GPR R4 CONTENTS.                                 *
1506* * SP+12 -> GPR R5 CONTENTS.                                 *
1507* * SP+14 -> RET PC INTO CALLER OF SUB'THE WHICH CALLED PREGOS.  *
1508* * * EACH LEVEL OF SUB'THE CALLING USES 8 WORDS OF STACK OVERHEAD.  *
1509* * THE SAVE AND PASS MACROS CAN ALSO BE USED IN "STRAIGHT LINE CODE"  *
1510* * TO SAVE AND RESTORE THE GPR VALUES. IN ANY CASE, AFTER THE  *
1511* * ISSUING OF A PASS CALL THE GPRS WILL BE RESTORED TO THE VALUES  *
1512* * THEY HAD PRIOR TO THE LAST SAVE CALL (EXCEPT FOR THE EXCEPTED,  *
1513* * OR PASSED INTACT, GPRS SPECIFIED AS PARAMETERS TO THE PASS CALL)  *
1514* * AND THE SP WILL ALSO BE RESTORED TO ITS CONDITION BEFORE THE LAST  *
1515* * SAVE CALL. THE PROGRAMMERR MUST BE SURE THAT THE SP WAS THE SAME  *
1516* * VALUE WHEN THE PASS MACRO IS CALLED AS IT HAD IMMEDIATELY AFTER  *
1517* * THE SAVE MACRO WAS CALLED.                                *
1518*****************************************************************
<table>
<thead>
<tr>
<th>Address</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000036</td>
<td>36</td>
<td>OFFSET FOR LAST RETURN PC.</td>
</tr>
<tr>
<td>000016</td>
<td>16</td>
<td>OFFSET FOR RETURN PC.</td>
</tr>
<tr>
<td>000014</td>
<td>14</td>
<td>OFFSET FOR R5.</td>
</tr>
<tr>
<td>000012</td>
<td>12</td>
<td>OFFSET FOR R4.</td>
</tr>
<tr>
<td>000010</td>
<td>10</td>
<td>OFFSET FOR R3.</td>
</tr>
<tr>
<td>000006</td>
<td>6</td>
<td>OFFSET FOR R2.</td>
</tr>
<tr>
<td>000004</td>
<td>4</td>
<td>OFFSET FOR R1.</td>
</tr>
<tr>
<td>000002</td>
<td>2</td>
<td>OFFSET FOR R0.</td>
</tr>
</tbody>
</table>
**SBTL. GLOBAL MACRO DEFINITION**  
**SAVE**

```
*** THIS MACRO IS USED AT THE BEGINNING OF A SUBROUTINE TO SAVE THE ***
*** CONTENTS OF THE GPRS R0 THRU RS. ***
*** INPUTS: SP - UNCHANGED SINCE SUBROUTINE WAS ENTERED ***
*** RS SLOT - OFFSET TO STACK SLOT FOR RS (EQUATED TO 14 OCTAL) ***
*** OUTPUTS: GPR SAVE AREA ON THE STACK IS LOADED WITH THE CONTENTS OF GPRS ***
*** TOP OF STACK LOADED WITH THE RETURN ADDRESS INTO PREG05 ***
*** CALLING SEQUENCE: SAVE ***
*** COMMENTS: NO ARGUMENTS ARE ALLOWED. ***
*** THE PASS MACRO SHOULD BE CALLED TO RESTORE THE GPR VALUES. ***
*** SUBORDINATE ROUTINES CALLED: PREG05. ***
```
**.GLOBAL MACRO DEFINITION - PASS -**

**THIS MACRO IS USED IN CONJUNCTION WITH THE SAVE MACRO. IT IS**

**CALLED AT END OF A SUBROUTINE TO PASS PARAMETERS IN GPRS BACK TO THE**

**CALLING ROUTINE BY ALTERING THE GPR SAVE AREA ON THE STACK AND THEN**

**RETURNING TO PREGOS TO RESTORE THE GPRS TO THEIR SAVED VALUES.**

**INPUTS:**

**ONLY ALLOWED ARGUMENTS ARE "RO" THRU "RS5".**

**ROSLAT THRU RSSLAT MUST BE EQUATED TO THEIR RESPECTIVE GPR SAVE**

**SLOT OFFSETS BEFORE CALLING THIS MACRO.**

**OUTPUTS:**

**THE GPR VALUES ARE PUT IN THEIR RESPECTIVE SLOTS ON THE STACK.**

**CALLING SEQUENCE:**

**PASS RO,R1,**

**...**

**COMMENTS:**

**ANY COMBINATION OF GPR ARGUMENTS MAY BE LISTED IN ANY ORDER.**

**FOR EXAMPLE, THE FOLLOWING ARE LEGAL:**

**PASS R1**

**PASS R4,RO,R2**

**THE GPRS LISTED AS ARGUMENTS WILL BE PASSED INTACT TO THE**

**CALLING ROUTINE, ALL OTHER GPRS WILL BE RESTORED.**

**THE SP MUST BE AT ITS ORIGINAL VALUE WHEN PASS IS CALLED.**

**THE MACRO CALL**

**PASS RO,R3**

**EXPANDS INTO THE FOLLOWING ASSEMBLY CODE:**

**MOV RO,ROSLAT(SP) ;PUT RO IN STACK SLOT.**

**MOV R3,RSSLAT(SP) ;PUT R3 IN STACK SLOT.**

**JSR PC,(SP) ;RETURN TO PREGOS SUBRT.**

**IN THIS EXAMPLE GPRS R1, R2, R4, AND R5 WILL BE RESTORED TO**

**THEIR VALUES CONTAINED IN THE STACK FRAME AND RO AND R3**

**WILL BE LEFT AT THEIR VALUES PRIOR TO THIS PASS CALL.**

**SUBORDINATE ROUTINES CALLED: (PREGRT - LABEL WITHIN PREGOS, VALUE ON STACK.)**

```assembly
.MACRO PASS A,B,C,D,E,F
.IRP X,<A,B,C,D,E,F>
.IF NB,X
.LIST MOV X,X'SLOT(SP) ;PUT X IN STACK SLOT.
.NLIST ENDC
.ENDM
.LIST JSR PC,(SP) ;RETURN TO PREGOS SUBRT.
.NLIST ENDM PASS
```
**GLOBAL SUBROUTINE**

1614 .SBTTL GLOBAL SUBROUTINE
1616 PREG05
1618
1619 * PRESERVE REGISTERS R0 THROUGH R5 FOR SUBROUTINE CALLS.
1620 *
1621 * INPUTS: THE RETURN ADDRESS BACK INTO THE CALLING ROUTINE MUST BE IN
1622 * GPR R5, (I.E.: MACROS USE "JSR R5,PREG05:"
1623 *
1624 * OUTPUTS: REGISTERS R0 THROUGH R5 ARE SAVED ON THE STACK.
1625 *
1626 * CALLING SEQUENCE: [SUBROUTINE CODE]...
1627 * MACRO EXPANSION CALLS PREG05.
1628 * PASS MACRO EXPANSION RECALLS PREG05.
1629 *
1630 * COMMENTS: THIS ROUTINE IS RE-ENTRANT.
1631 *
1632 * PARAMETERS MAY BE PASSED OUT OF A SUBROUTINE BY MODIFYING THE
1633 * REGISTER SAVE AREA ON THE STACK. USE THE PASS GPR MACRO
1634 * TO RETURN GPR VALUES INTACT.
1635 *
1636 * USE THE RNSLOT OFFSETS FROM THE SP TO PASS OTHER PARAMETERS.
1637 * [EXAMPLE: MOV VALUE,RNSLOT(SP)]
1638 * MAKE SURE THE SP IS AT ITS ORIGINAL VALUE WHEN YOU DO THIS.
1639 *
1640 * SUBORDINATE ROUTINES CALLED: NONE.
1641 *

**PREG05:**

1642 MOV R4, (SP) ; SAVE R4
1643 MOV R3, (SP) ; SAVE R3
1644 MOV R2, (SP) ; SAVE R2
1645 MOV R1, (SP) ; SAVE R1
1646 MOV R0, (SP) ; SAVE R0
1647 MOV R5, (SP) ; PUSH RETURN PC ON TOP OF STACK
1648 JSR PC,(SP) ; CALL THE SUBROUTINE AT THE RETURN ADDRESS;
1649 ; FROM THE PREG05 CALL, PUTTING THE PRESENT
1650 ; PC ON THE STACK AS A RETURN ADDRESS INTO
1651 ; THIS (PREG05) ROUTINE.
1652 *
1653 ; THE FOLLOWING CODE IS EXECUTED WHEN THE CALLING ROUTINE DOES A
1654 ; "RETURN" [JSR PC,(SP)] USING THE PC DEPOSITED ON THE STACK ABOVE.
1655 *
1656 *
1657 PREG05: MOV (SP),R5 ; PUT RETURN PC IN R5.
1658 MOV (SP),R0 ; RESTORE R0.
1659 MOV (SP),R1 ; RESTORE R1.
1660 MOV (SP),R2 ; RESTORE R2.
1661 MOV (SP),R3 ; RESTORE R3.
1662 MOV (SP),R4 ; RESTORE R4.
1663 MOV (SP),R5 ; RETURN TO THE SUBROUTINE WHICH CALLED PREG05.
1664 RESTORING R5 IN THE PROCESS.
**GLOBAL TEXT SECTION**

```assembly
; SBTLGLOBAL TEXT SECTION

; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
; MORE THAN ONE TEST.

; NAMES OF DEVICES SUPPORTED BY PROGRAM
; DESCRIPT <DMU-11>

; DESCRIPT <DMU 11 FUNC TST PART2>

; L4DESC::
; ASCIZ /DMU-11 FUNC TST PART2/>

; L4DVTYP::
; ASCIZ /DMU-11/;

; .EVEN
```

---

1687 004120
004120 004120 104 110 125
004123 055 061 061
004126 000

1697 004130
004130 004130 104 110 125
004133 055 061 061
004136 040 106 125
004141 116 103 040
004144 124 123 124
004147 040 120 101
004152 122 124 062
004155 000

1698
1699
1706

; .EVEN
1715 .NLIST BIN

1716

1717

1718

1719

1720

1721

1722 004156 EF0503:: .ASCIZ */#N/
1723 004163 EF1601:: .ASCIZ */A #FAIL, TEST ABORTED #N/
1724 004215 EF5801:: .ASCIZ */A RXRIM VALUE USED WAS #:D3A (D)#N/
1725 004270 EF5901:: .ASCIZ */A EXPECTED #:D3A(D)#N/
1726 004320 EF5902:: .ASCIZ */A ACTUAL #:D3A(D)#N/
1727 004350 EF6401:: .ASCIZ */A #D2#N/
1728 004417 EF7501:: .ASCIZ */A ON LINE #D2#A DECIMAL #N/
1729 004455 EF9001:: .ASCIZ */A UNEXPECTED #FAIL FOUND IN RECEIVE CHAR FIFO:#N/
1730 004537 EF9002:: .ASCIZ */A CODE IS ASSOCIATED WITH LINE: #D2#N/
1731 004611 EF9003:: .ASCIZ */A CODE IS: #:D3#N/
1732 004640 EF9004:: .ASCIZ */A #FAIL VALUE: #:D3#N/
1733 004670 EF9005:: .ASCIZ */A #FAIL VALUE: NONE#N/
1734 004721 EF9006:: .ASCIZ */A #FAIL #:D2A(D)#N/
1735 004745 EF9010:: .ASCIZ */A NUMBER OF ERRORS DETECTED ON LINE #:D2#A IS #:D5#N/
1736 005034 EF9019:: .ASCIZ */A #FAIL #:D2A(D), BMP CODE REPORTED #:D3A(D)#N/
1737 005131 EF9301:: .ASCIZ */A OVERFLOW OCCURRED (MORE THAN 31 BMP CODES FOUND IN QUEUE)#N/
1738 005251 MFUNIT:: .ASCIZ */A TESTING UNIT #:D4#N/
1740 .EVEN

1741 .LIST BIN
**LIST BIN**

;********* GLOBAL ERROR MESSAGES **********

005262 EM0103:: .ASCIZ /DEVICE REGISTER ACCESS ERRORS/
005320 EM1601:: .ASCIZ /TIME OUT OCCURED WAITING FOR MASTER RESET TO CLEAR/
005403 EM0001:: .ASCIZ /DMA START BIT TEST FAILED/
005435 EM4002:: .ASCIZ /DMA START BIT BAD ON LINE: /
005471 EM4010:: .ASCIZ /DMA ABOERT BIT TEST FAILED/
005523 EM4102:: .ASCIZ /DMA ABOERT BIT BAD ON LINE: /
005557 EM4103:: .ASCIZ /DMA START BIT FOUND SET AFTER DMA ABOERT ON LINE: /
005643 EM4201:: .ASCIZ /DMA ERROR BIT TEST FAILED/
005675 EM4302:: .ASCIZ /DMA ERROR BIT BAD/
005717 EM4901:: .ASCIZ /DMAUTO (INACTIVE) BIT TEST FAILED/
005760 EM4902:: .ASCIZ /DMAUTO BIT BAD ON LINE: /
006012 EM5001:: .ASCIZ /DMAUTO (ACTIVE) BIT TEST FAILED/
006051 EM5101:: .ASCIZ /INAUTO (INACTIVE) TEST FAILED/
006106 EM5102:: .ASCIZ /INAUTO BIT FOUND SET ON LINE: /
006144 EM5103:: .ASCIZ /INAUTO BIT BAD ON LINE: /
006174 EM5201:: .ASCIZ /INAUTO (ACTIVE) TEST FAILED/
006227 EM5202:: .ASCIZ /INAUTO BIT FOUND CLR ON LINE: /
006265 EM5301:: .ASCIZ /FIFO VALID DATA TEST FAILED/
006321 EM5302:: .ASCIZ /FIFO BAD, DATA FIELD CORRUPTED, TEST USED LINE: /
006401 EM5303:: .ASCIZ /BMP CODE FOUND IN FIFO, TEST INVALIDATED/
006452 EM5401:: .ASCIZ /FIFO 3/4 ALARM (INACTIVE) TEST FAILED\n006520 EM5402:: .ASCIZ /FIFO BAD, ALARM SIGNAL DEFECTIVE/
006561 EM5501:: .ASCIZ /FIFO 3/4 ALARM (ACTIVE) TEST FAILED\n006625 EM5601:: .ASCIZ /FIFO 3/4 ALARM (ACTIVE/INACTIVE) TEST FAILED\n006702 EM5701:: .ASCIZ /FIFO 1/2 LEVEL (ACTIVE/INACTIVE) TEST FAILED\n006757 EM5801:: .ASCIZ /RXTIMER TEST FAILED\n007003 EM5802:: .ASCIZ /RXTIMER BAD, RX-INT DELAYED BY WRONG NUMBER OF MILLISECCONDS\n007077 EM5803:: .ASCIZ /RXTIMER BAD, RX-INT DIDN'T OCCUR IMMEDIATELY WITH RXFIFO 3/4 FULL\n007201 EM5804:: .ASCIZ /RXTIMER BAD, RX-INT OCCURED WITH RXTIMER VALUE ZERO\n007265 EM5805:: .ASCIZ /RXTIMER BAD, TIME-OUT OCCURED WAITING FOR RX-INT\n007346 EM5901:: .ASCIZ /TX-ACTION FIFO TEST FAILED\n007402 EM5902:: .ASCIZ /TX-ACTION FIFO BAD, TX-ACTION RECIIVED FROM THE WRONG LINE\n007476 EM5903:: .ASCIZ /TX-ACTION FIFO BAD, INCORRECT NUMBER OF TX-ACTIONS FOUND\n007570 EM5904:: .ASCIZ /TX-ACTION FIFO BAD, TX-ACTION FIFO WOULD NOT EMPTY\n007654 EM5905:: .ASCIZ /TX INTERRUPT OCCURED AFTER THE TX-ACTION FIFO HAD BEEN EMTIED\n007754 EM6001:: .ASCIZ /TX FIFO TEST FAILED\n008000 EM6002:: .ASCIZ /INCORRECT VALUE IN FIFO SIZE REG\n008040 EM6003:: .ASCIZ /MORE THAN ONE TX-INT OCCURED FROM A FULL TXFIFO\n008120 EM6004:: .ASCIZ /TX FIFO BAD, RECEIVED CHAR INCORRECT\n008165 EM6005:: .ASCIZ /TX FIFO BAD, CHARACTER RECEIVED ON WRONG LINE\n008243 EM6006:: .ASCIZ /TX FIFO BAD, TOO FEW CHARs RECEIVED\n008507 EM6401:: .ASCIZ /BREAK GENERATION TEST FAILED\n008642 EM6402:: .ASCIZ /BREAK NOT RECEIVED ON LINE(S): /
008744 EM6601:: .ASCIZ /NO OVERRUN ERROR TEST FAILED\n008842 EM6602:: .ASCIZ /OVERRUN ERROR REPORTED WHEN NONE FORCED\n008954 EM6701:: .ASCIZ /OVERRUN ERROR TEST FAILED\n008956 EM6702:: .ASCIZ /NO OVERRUN ERROR REPORTED, OVERRUN FORCED\n008963 EM6909:: .ASCIZ /EXPECTED OR CORRECT\n008964 EM7010:: .ASCIZ /ACTUAL OR MEASURED\n008965 EM9014:: .ASCIZ /SUMMARY REPORTS FOR LINES WITH EXCESSIVE NUMBERS OF ERRORS:\n008976 EM9017:: .ASCIZ /FIFO WILL NOT PURGE (DATA,VALID STUCK SET),/
1807 011044  .ASCIZ / REMAINDER OF TEST SKIPPED. /
1808 011100  EM9026:: .ASCIZ / LPR CONTENTS: /
1809 011124  EM9104:: .ASCIZ / UNEXPECTED DATA FOUND IN FIFO FROM LINE: /
1810 011200  EM9301:: .ASCIZ / BMP CODES WERE REPORTED DURING THIS DIAGNOSTIC /
1811 011257  EM9302:: .ASCIZ / BMP CODE FOUND IN TEST /
1812 011307  EM9303:: .ASCIZ / THE LAST BMP CODE WAS FOUND IN TEST /
1813 011354  EM9304:: .ASCIZ / UNEXPECTED BMP CODES FOUND DURING THIS PASS /
1814
1815      .EVEN
1816      .LIST BIN
.SBTTL GLOBAL ERROR REPORT SECTION

"THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
USED BY MORE THAN ONE TEST TO OUTPUT ADDITIONAL ERROR INFORMATION. PRINTB
(BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
"
GLOBAL ERROR REPORTING ROUTINE

**SUBTL:** GLOBAL ERROR REPORTING ROUTINE - ERO101

* THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS ADDITIONAL ERROR
* INFORMATION IF AN ERROR IS DETECTED IN TEST 1 (REGISTER ADDRESS
* ACCESS TEST). IF THE "EXTENDED ERROR INFO" OPTION HAS BEEN SELECTED
* THEN THIS SUBROUTINE WILL REPORT THE TYPE OF ACCESS (READ OR WRITE OR
* BOTH) WHICH CAUSED A BUS TIME-OUT TRAP (004 TRAP), A MESSAGE INDICATING
* THAT THE DMU MAY BE AT THE WRONG UNIBUS ADDRESS IS ALSO PRINTED.

* INPUTS: R5 - ERROR FLAG WORD.
  * IF BIT 0 IS SET, A READ ERROR OCCURRED.
  * IF BIT 1 IS SET, A WRITE ERROR OCCURRED.

* OUTPUTS: MESSAGES ARE PRINTED AT THE OPERATOR CONSOLE.

* CALLING SEQUENCE: INCLUDE THE LABEL "ERO101" AS THE MESSAGE POINTER
  * PARAMETER IN THE ONS ERROR REPORT MACRO CALL.

* COMMENTS:

* SUBORDINATE ROUTINES USED: NONE.

```
BGHNSG ERO101
SAVE ERO101::
JSR R5,PRECS ;CALL REGISTER SAVE SUBRT.

MOV #BIT0,RO ;SET-UP THE BIT MAP FOR 'REPORT EXT'D ERROR INFO'
BIC OPTION,RO ;TRY AND CLEAR THE FLAG.
BNE 6# ;EXIT IF OPTION NOT SELECTED.

; REPORT EXTENDED ERROR INFORMATION

BIT #BIT0,R5 ;TEST FOR READ ERROR.
BEQ 2# ;SKIP READ ERROR MSG IF NO READ ERROR.
PRINTB #MSG1 ;PRINT READ ERROR MESSAGE.

MOV #MSG1,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C1PNTB
ADD #4,SP

2#: BIT #BIT1,R5 ;TEST FOR WRITE ERROR.
BEQ 4# ;SKIP WRITE ERROR MSG IF NO WRITE ERROR.
PRINTB #MSG2 ;PRINT WRITE ERROR MESSAGE.

MOV #MSG2,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C1PNTB
ADD #4,SP

4#: PRINTX #MSG3 ;SUGGEST THAT DMU MAY BE AT WRONG ADDRESS.

MOV #MSG3,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C1PNTX
ADD #4,SP
```
<table>
<thead>
<tr>
<th>Line</th>
<th>Address</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1874</td>
<td>101542</td>
<td>00436</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1875</td>
<td>101544</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1876</td>
<td>101544</td>
<td>104423</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1877</td>
<td>101546</td>
<td>045101</td>
<td>102</td>
<td>MSG1:</td>
</tr>
<tr>
<td></td>
<td>011551</td>
<td>125123</td>
<td>040</td>
<td>.ASCIZ</td>
</tr>
<tr>
<td></td>
<td>011554</td>
<td>124111</td>
<td></td>
<td>/#ABUS TIME-OUT TRAP CAUSED BY READ ATTEMPT.\n/</td>
</tr>
<tr>
<td></td>
<td>011557</td>
<td>105055</td>
<td>117</td>
<td>.\n/</td>
</tr>
<tr>
<td></td>
<td>011562</td>
<td>125124</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011565</td>
<td>124122</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011570</td>
<td>120040</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011573</td>
<td>101125</td>
<td>123</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011576</td>
<td>105104</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011601</td>
<td>102131</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011604</td>
<td>122105</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011607</td>
<td>104040</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011612</td>
<td>124120</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011615</td>
<td>115120</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011620</td>
<td>056045</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011623</td>
<td>000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1878</td>
<td>011624</td>
<td>045101</td>
<td>102</td>
<td>MSG2:</td>
</tr>
<tr>
<td></td>
<td>011627</td>
<td>125123</td>
<td>040</td>
<td>.ASCIZ</td>
</tr>
<tr>
<td></td>
<td>011632</td>
<td>124111</td>
<td></td>
<td>/#ABUS TIME-OUT TRAP CAUSED BY WRITE ATTEMPT.\n/</td>
</tr>
<tr>
<td></td>
<td>011635</td>
<td>105055</td>
<td>117</td>
<td>.\n/</td>
</tr>
<tr>
<td></td>
<td>011640</td>
<td>125124</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011643</td>
<td>124122</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011646</td>
<td>120120</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011651</td>
<td>101125</td>
<td>123</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011654</td>
<td>005104</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011657</td>
<td>102131</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011662</td>
<td>127122</td>
<td>111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011665</td>
<td>124105</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011670</td>
<td>101124</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011673</td>
<td>105115</td>
<td>120</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011676</td>
<td>124124</td>
<td>056</td>
<td>045</td>
</tr>
<tr>
<td></td>
<td>011701</td>
<td>116</td>
<td>000</td>
<td></td>
</tr>
<tr>
<td>1879</td>
<td>011703</td>
<td>045101</td>
<td>104</td>
<td>MSG3:</td>
</tr>
<tr>
<td></td>
<td>011706</td>
<td>110125</td>
<td>040</td>
<td>.ASCIZ</td>
</tr>
<tr>
<td></td>
<td>011711</td>
<td>115101</td>
<td>131</td>
<td>/#ADMU MAY BE AT THE WRONG UNIBUS ADDRESS.\n/</td>
</tr>
<tr>
<td></td>
<td>011714</td>
<td>040102</td>
<td>105</td>
<td>.\n/</td>
</tr>
<tr>
<td></td>
<td>011717</td>
<td>040101</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011722</td>
<td>040124</td>
<td>110</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011725</td>
<td>105040</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011730</td>
<td>122117</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011733</td>
<td>107040</td>
<td>125</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011736</td>
<td>116111</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011741</td>
<td>125123</td>
<td>040</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011744</td>
<td>101104</td>
<td>104</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011747</td>
<td>122105</td>
<td>123</td>
<td></td>
</tr>
<tr>
<td></td>
<td>011752</td>
<td>123124</td>
<td>056</td>
<td>045</td>
</tr>
<tr>
<td></td>
<td>011755</td>
<td>116124</td>
<td>045</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td>011760</td>
<td>000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1980

1981

.EVEN
GLOBAL ERROR REPORTING ROUTINE

* THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS AN ADDITIONAL ERROR
* MESSAGE WHOSE ADDRESS IS PASSED AS AN INPUT PARAMETER, PROVIDED
* EXTENDED ERROR REPORTING HAS BEEN REQUESTED.

* INPUTS: R1 - ADDRESS OF THE MESSAGE TO PRINT.
* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
* INCLUDE THE LABEL "ER0503:" AS THE MESSAGE POINTER
* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
* COMMENTS: THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION.
* SUBORDINATE ROUTINES USED: NONE.

BGNMSG ER0503

ER0503::

MOV @BIT06,RO ;TRY TO CLEAR THE
BIC OPTION,RO ;EXT.'D ERROR REPORTING FLAG
BNE 2$ ;EXIT IF FLAG NOT SET.

PRINTB @EF0503,R1 ;PRINT THE MESSAGE.

MOV R1,(SP)
MOV @EF0503,(SP)
MOV @2,(SP)
MOV SP,R0
TRAP C1PNTB
ADD $6,SP

2$: ENDMSSG

L10003: TRAP C1MSG
GLOBAL ERROR REPORTING ROUTINE

**SUBTITLE**
GLOBAL ERROR REPORTING ROUTINE

- ER1603 -

**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**

**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, E.G. DMA, BREAK ETC.

**OUTPUTS:**
- MESSAGES ARE PRINTED AT THE OPERATORS CONSOLE.
- "TESTNAME TEST ABORTED"

**CALLING SEQUENCE:**
INCLUDE THE LABEL "ER1603" AS THE MESSAGE POINT.
PARAMETER IN THE DRS ERROR REPORT MACRO CALL.

**COMMENTS:**

**SUBORDINATE ROUTINES CALLED:**
NONE.

**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**
**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**

BGMSG ER1603

```
SAVE
JSR  ;SAVE THE CONTENTS OF THE GPRS.
RS,PREG05  ;CALL REGISTER SAVE SUBRT.

MOV  #BIT06,RO  ;TRY TO CLEAR THE
BIC  OPTION,RO  ;EXT'D ERROR REPORTING FLAG
BNE  2$  ;EXIT IF FLAG NOT SET.

PRINTB  #EF0503,R1  ;PRINT BASIC MESSAGE ON OPERATORS CONSOLE.
MOV  R1, (SP)
MOV  #EF0503,-(SP)
MOV  #2, (SP)
MOV  SP,RO
TRAP  CIPNTB
ADD  #6, SP

MOV  ER6MSG, R2  ;GET THE "TEST MESSAGE".
PRINTB  #EF1601,R2  ;PRINT "TEST ABORTED" MESSAGE.

MOV  R2, (SP)
MOV  #EF1601,-(SP)
MOV  #2, (SP)
MOV  SP,RO
TRAP  CIPNTB
ADD  #6, SP

2$:  PASS  ;RESTORE THE CONTENTS OF THE GPRS.
JSA  PC, @(SP)  ;RETURN TO PREG05 SUBRT.
ENDMSG
```

L10004: TRAP CIMS.
GLOBAL ERROR REPORTING ROUTINE

.SBTTL GLOBAL ERROR REPORTING ROUTINE

THESE ARE THE ERROR REPORTING ROUTINE WHICH PRINTS THE MESSAGE PASSED
AS A PARAMETER IN R1, AND THE RXTIMER VALUE IN R2, PROVIDED
EXTENDED ERROR REPORTING HAS BEEN REQUESTED.
THIS ROUTINE IS USED BY THE RXTIMER TEST.

* INPUTS:  R1 - ADDRESS OF THE MESSAGE TO PRINT.
*         R2 - RXTIMER VALUE.
* OUTPUTS: THE MESSAGE FOLLOWED BY THE RXTIMER VALUE ARE PRINTED AT
*         THE OPERATOR CONSOLE.
* CALLING SEQUENCE: INCLUDE THE LABEL ER5801 AS THE MESSAGE POINTER
*                  PARAMETER IN THE DAS ERROR REPORT MACRO CALL.
* COMMENTS:  THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION AND THE
*            RXTIMER VALUE IS PRINTED AS A 3 DIGIT DECIMAL NUMBER.
* SUBORDINATE ROUTINES USED: NONE.

BGMSG ER5801

BIT @BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
BEQ 2:
;EXIT WITH "TEST FAILED" MESSAGE IF NOT.

PRINTB @EF0503,R1
MOV R1, -(SP)
MOV @EF0503, -(SP)
MOV @SP, -(SP)
MOV SP, R0
TRAP CSPNTB
ADD @SP, 6

PRINTB @EF5801,R2
MOV R2, -(SP)
MOV @EF5801, -(SP)
MOV @SP, -(SP)
MOV SP, R0
TRAP CSPNTB
ADD @SP, 6

ENDMSG

L10005:
TRAP CSPMSG
GLOBAL ERROR REPORTING ROUTINE

SBTTL GLOBAL ERROR REPORTING ROUTINE - ER5901 -

***************

* THIS IS AN ERROR REPORTING ROUTINE WHICH PRINTS AN ADDITIONAL
* MESSAGE IN ADDITION TO THE "TEST FAILED" MESSAGE AND ALSO A
* MESSAGE SHOWING THE EXPECTED VALUE OF A PIECE OF DATA AND THE
* ACTUAL VALUE OF THAT DATA. THE DATA IS PRINTED AS A 3 DIGIT
* DECIMAL NUMBER.

* INPUTS :     R1 - EXPECTED VALUE OF DATA,
*              R2 - ACTUAL VALUE OF DATA,
*              R3 - ADDRESS OF THE MESSAGE TO PRINT.

* OUTPUTS :    MESSAGES ARE PRINTED AT THE OPERATORS CONSOLE.

* CALLING SEQUENCE: INCLUDE THE LABEL "ER5901" AS THE MESSAGE POINTER
* PARAMETER IN THE DRS ERROR REPORT MACRO CALL.

* COMMENTS :

* SUBORDINATE ROUTINES USED : NONE.

***************

BGHMSG ER5901

ER5901:

SAVE

JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.

BIT #BIT06,OPTION ;EXIT THE ROUTINE IF EXTENDED
BEQ 601 ;ERROR REPORTING IS NOT REQUESTED.

; REPORT EXTENDED ERROR INFORMATION

PRINTB #EF0503,R3 ;PRINT THE MESSAGE.

MOV R3,-(SP)
MOV #EF0503,(SP)
MOV #R2,-(SP)
MOV SP,R0
TRAP CIPNTB
ADD #6,SP

PRINTX #EF5901,R1 ;PRINT THE "EXPECTED VALUE" MESSAGE.

MOV R1,-(SP)
MOV #EF5901,-(SP)
MOV #R2,-(SP)
MOV SP,R0
TRAP CIPNTX
ADD #6,SP

PRINTX #EF5902,R2 ;PRINT THE "ACTUAL VALUE" MESSAGE.

MOV R2,-(SP)
MOV #EF5902,-(SP)
MOV #R2,-(SP)
MOV SP,R0
TRAP CIPNTX
ADD #6,SP
601: PASS
ENDMG

;RESTORE THE GPR CONTENTS.
JSR PC, @ (SP)

;RETURN TO PREGOS SUBRT.

L10006: TRAP CMSG
GLOBAL ERROR REPORTING ROUTINE

**GLOBAL ERROR REPORTING ROUTINE**

This is an error reporting routine which prints out a message at the console informing the operator of an error on a particular line. The routine also prints out a message informing the operator of what data was "expected" and what "actual" data was found, in the form of a 3 digit decimal number.

If extended error reporting has not been requested then only the "test failed" message will be printed.

**Inputs:**
- R1 - Expected Data
- R2 - Actual Data
- R3 - Address of the message to print
- R4 - Line number on which the error occurred

**Outputs:**
- Messages are printed at the operator console
- ""Expected:" "Actual:" output

**Calling sequence:**
- Include the label "ER6001" as the error routine pointer parameter in the DRS error report macro call.

**Subordinate routines called:**
- None

**Source Code:**

```
012276 012276 032767 000100 167704
012304 001434

BIT  @BIT06, OPTION  ; Exit the routine if extended
BEQ  601  ; Error reporting is not requested.

; Report extended error information

PRINTB  @EF7801, R3, R4  ; Print the message with the line number.
MOV    R4, -(SP)
MOVT   R3, -(SP)
MOV    @EF7801, -(SP)
MOV    @EF5901, -(SP)
MOV    @EF5902, -(SP)
TRAP   C1PNTB
ADD    @10, SP

PRINTX  @EF5901, R1  ; Print the "expected" data message.
MOV    R1, -(SP)
MOV    @EF5901, -(SP)
MOV    @EF5902, -(SP)
TRAP   C1PNTX
ADD    @6, SP

PRINTX  @EF5902, R2  ; Print the "actual" data message.
MOV    R2, -(SP)
MOV    @EF5902, -(SP)
MOV    @EF5901, -(SP)
MOV    SP, RO
TRAP   C1PNTX
ADD    @6, SP
```

SEQ 53
GLOBAL ERROR REPORTING ROUTINE
MACRO ML200 15 MAR 84 09:28 PAGE 45 1

2059 012376
012376
012376 104423

501: ENDMVG

L10007: TRAP CMMSG
GLOBAL ERROR REPORTING ROUTINE

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.

                   INCLUDE THE LABEL "ER6401" AS THE MESSAGE POINTER.
                   PARAMETER IN THE DIAK 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.

BGMSG ER6401

SAVE
JSR R5,PREGOS ;CALL REGISTER SAVE SUBRT.

;EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT @BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
BEQ 601 ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
DURING THE SOFTWARE QUESTIONS.

CLR R2 ;CLEAR LINE NUMBER TO ZERO.
MOV R2,#00000100
PRINTB #EF0503,R1 ;PRINT MESSAGE.

MOV R1,#(SP)
MOV #EF0503,(SP)
MOV #2,(SP)
MOV SP,RO
TRAP CIPRTB
ADD #6,SP

CLC ;CLEAR CARR1.
ASR R5 ;SHIFT FLAG OUT INTO CARRY BIT.
BCC 43 ;SKIP ERROR REPORT IF CLEAR.
PRINTB #EF6401,R2 ;PRINT MESSAGE.
```
012470  062706  000006
2105  012474  005202  41:  INC  R2  ;INCREMENT LINE COUNT.
2106  012476  020302  CMP  R3,R2  ;CHECK IF MAX LINE COUNT EXCEEDED.
2107  012500  001362  BNE  21  ;LOOP IF NOT DONE.
2108  012502  004736  60$:  PASS  ;RESTORE THE SAVED CONTENTS OF THE GPRs.
2109  012504  004736  ENDMNG  JSR  PC,(SP)  ;RETURN TO PREGOS SUBRT.
012504
012504  104423
```

GLOBAL ERROR REPORTING ROUTINE

.SBTL GLOBAL ERROR REPORTING ROUTINE       ER7801

* THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS AN ADDITIONAL ERROR
* MESSAGE WHOSE ADDRESS IS PASSED AS AN INPUT PARAMETER. A LINE NUMBER
* IS INCLUDED AT THE END OF THE MESSAGE. THE MESSAGE IS PRINTED ONLY IF
* EXTENDED ERROR REPORTING IS REQUESTED.

* INPUTS:   R1 - ADDRESS OF THE MESSAGE TO PRINT.
*          R3 - NUMBER OF LINE ON WHICH ERROR OCCURRED.

* OUTPUTS:  A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.

*                    LOAD THE LINE NUMBER INTO R3.
*                    INCLUDE THE LABEL "ER7801" AS THE MESSAGE POINTER
*                    PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.

* COMMENTS:  THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION.

* SUBORDINATE ROUTINES USED: NONE.

BGMSG  ER7801

ER7801:

; EXIT IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT   @B1106,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF

BEG  2$ ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
DURING THE SOFTWARE QUESTIONS.

PRINTB  @EF7801,R1,R3 ;PRINT THE MESSAGE.

MOV    R3,-(SP)
MOV    R1,-(SP)
MOV    @EF7801,-(SP)
MOV    @3,-(SP)
MOV    SP,R0
TRAP   C;PRINTB
ADD    @10,SP

2$:   ENDSMSG

L10011:  TRAP   C;MSG
GLOBAL ERROR REPORTING ROUTINE

BEGIN
* THIS IS AN ERROR REPORTING ROUTINE WHICH REPORTS AN UNEXPECTED
* CODE WHICH HAS BEEN FOUND IN THE DUT CSR. THIS CODE CAN BE A BMP
* CODE, A SELF-TEST CODE, OR A MODEM STATUS CODE.
*
* INPUTS:  
* R1 - ADDRESS OF MESSAGE TO PRINT FIRST.
* R2 - SINGLE BYTE CODE WHICH HAS BEEN READ FROM THE DUT.
* R4 - LINE NUMBER ASSOCIATED WITH THE CODE.
*
* OUTPUTS:  
* A MESSAGE IS PRINTED AT THE OPERATOR CONSOLE.
*
* CALLING SEQUENCE:  
* INCLUDE THE LABEL "ER9001" AS THE MESSAGE POINTER
* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
*
* COMMENTS:  
* THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
*
* SUBORDINATE ROUTINES USED:  

BGNE ERROR ROUTINE ER9001

ER9001:

** EXIT IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED **

BIT #BIT06,OPTION ; EXIT WITH TEST FAILURE MESSAGE IF
BEQ 2:

PRINTB #EF9001,R1 ; REPORT TYPE OF CODE FOUND.

MOV R1, (SP)
MOV #EF9001, (SP)
MOV #2, (SP)
MOV SP, R0
TRAP C:PNX
ADD #6, SP

PRINTX #EF9002,R4 ; REPORT THE LINE NUMBER OF THE CODE.

MOV R4, (SP)
MOV #EF9002, (SP)
MOV #2, (SP)
MOV SP, R0
TRAP C:PNX
ADD #6, SP

PRINTX #EF9003,R2 ; REPORT THE CODE WHICH WAS FOUND

MOV R2, (SP)
MOV #EF9003, (SP)
MOV #2, (SP)
MOV SP, R0
TRAP C:PNX
ADD #6, SP

ENDMSG

L:10012: TRAP C:MSG
.SBTIL GLOBAL ERROR REPORTING ROUTINE - ER9002 -

;******************************************************************************
;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH IS INTENDED FOR USE IN THE
;* TRANSMISSION AND RECESSION TESTS. IT REPORTS THE TYPE OF ERROR WHICH
;* HAS OCCURRED WHEN INCORRECT DATA IS RECEIVED FROM THE DUT. THIS
;* ROUTINE ALSO REPORTS THE READ AND EXPECTED DATA VALUES.
;******************************************************************************

;* INPUTS:  R1 - ADDRESS OF MESSAGE TO PRINT FIRST.
;*          R2 - DATA BYTE READ FROM THE DUT.
;*          R3 - LINE NUMBER MULTIPLIED BY 2.
;*          R4 - EXPECTED DATA BYTE. BIT 15 SET IF "NONE".
;* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9002" AS THE MESSAGE POINTER
;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
;* SUBORDINATE ROUTINES USED: PR1LPR.

BGNSG ER9002

ER9002:

*EXIT IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION
BEQ 62:

NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED;
DURING THE SOFTWARE QUESTIONS.

ASR R3
BIC #177400,R2
;CALCULATE THE LINE NUMBER.
PRINTB #ER9006,R1,R3
;PRINT THE FIRST LINE OF THE MESSAGE.

MOV R3,(SP)
MOV R1,(SP)
MOV #ER9006,(SP)
MOV #3,(SP)
MOV SP,R0
TRAP CIPNTB
ADD #10,SP

PRINTX #ER9004,#EM9010,R2
;PRINT ACTUAL DATA.

MOV R2,(SP)
MOV #EM9010,(SP)
MOV #ER9004,(SP)
MOV #3,(SP)
MOV SP,R0
TRAP CIPNTX
ADD #10,SP

TST R4
;CHECK FOR "NONE" CODE SET IN EXPECTED DATA.
BMI 21:
;BRANCH TO PRINT "NONE" MESSAGE IF FLAG SET.

PRINTX #ER9004,#EM9009,R4
;PRINT EXPECTED DATA.

MOV R4,(SP)
MOV #EM9009,(SP)
MOV #ER9004,(SP)
MOV #3,(SP)
<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>012756</td>
<td>010600</td>
<td>MOV SP, RO</td>
</tr>
<tr>
<td>012760</td>
<td>104415</td>
<td>TRAP C$PNTX</td>
</tr>
<tr>
<td>012762</td>
<td>062706 000010</td>
<td>ADD #10, SP</td>
</tr>
<tr>
<td>2219 012766</td>
<td>000412</td>
<td>BR 604</td>
</tr>
<tr>
<td>2220 012770</td>
<td>012746 010623</td>
<td>PRINTX #EF9005,#EM9009</td>
</tr>
<tr>
<td></td>
<td>012774 012746 004670</td>
<td>PRINT MESSAGE INDICATING NO EXPECTED DATA</td>
</tr>
<tr>
<td></td>
<td>013000 012746 000002</td>
<td></td>
</tr>
<tr>
<td></td>
<td>013004 010600</td>
<td></td>
</tr>
<tr>
<td></td>
<td>013006 104415</td>
<td></td>
</tr>
<tr>
<td></td>
<td>013010 062706 000006</td>
<td></td>
</tr>
<tr>
<td>2221 013014</td>
<td>004767 002266</td>
<td>JSR PC,PRILPR</td>
</tr>
<tr>
<td>2222 013020</td>
<td>624</td>
<td></td>
</tr>
<tr>
<td></td>
<td>013020 104423</td>
<td>L10013: TRAP C$MSG</td>
</tr>
</tbody>
</table>
.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9004 -

* THIS IS AN ERROR REPORTING SUBROUTINE WHICH REPORTS ERROR SUMMARIES
* FOR LINES WHICH HAVE EXCEEDED THE SPECIFIED MAXIMUM NUMBER OF
* INDIVIDUAL RECEPTION ERRORS, PROVIDED EXTENDED ERROR REPORTING HAS
* BEEN REQUESTED BY THE OPERATOR.

* INPUTS:   R1 - ADDRESS OF MESSAGE TO PRINT FIRST.
*           ERCNTB - LABEL AT BASE OF LINE ERROR COUNTERS TABLE.
*           ESRSRF - "REPORT ERROR SUMMARY FOR LINE" FLAGS.

* OUTPUTS:  A MESSAGE IS PRINTED AT THE OPERATOR CONSOLE.

* CALLING SEQUENCE:  INCLUDE THE LABEL "ER9004" AS THE MESSAGE POINTER
*                     PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.

* COMMENTS:  THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
*            THE CONTENTS OF GPR'S R2, R3, R4, AND R5 ARE DESTROYED.

* SUBORDINATE ROUTINES USED: NONE.

BGMSG ER9004

ER9004:

MOV @BIT06, R0 ; TRY TO CLEAR THE
BIC OPTION, R0 ; EXT. ERROR REPORTING FLAG
BNE 61 ; EXIT IF FLAG NOT SET.
PRINTB @EF0503, @EM9014 ; REPORT THE SECONDARY ERROR MESSAGE.

MOV @EM9014, -(SP)
MOV @EF0503, -(SP)
MOV @R2, -(SP)
MOV @SP, R0
TRAP @SP
ADD @SP

CLR R2 ; CLEAR THE LINE COUNTER.
MOV ESRSRF, R3 ; GET THE ERROR SUMMARY FLAGS.
CLR R4 ; CLEAR "LINE COUNTER TIMES 2" OFFSET.
CLC ; CLEAR THE CARRY FOR THE FOLLOWING ROTATE.
ROR R3 ; SHIFT ANOTHER ERROR SUMMARY FLAG INTO CARRY.
BCC 41 ; SKIP PRINTING MESSAGE IF FLAG FOR LINE CLEAR.

PRINTX @EF9010, R2, ERCNTB(R4)

MOV ERCNTB(R4), -(SP)
MOV R2, -(SP)
MOV @EF9010, -(SP)
MOV @R3, -(SP)
MOV @SP, R0
TRAP @SP
ADD @SP

INC (R4)++, R5 ; INCREMENT THE LINE OFFSET BY 2.

INC R2 ; INCREMENT THE LINE COUNTER.
TST R3 ; CHECK THE ERROR SUMMARY FLAGS.
BNE 24 ; IF MORE FLAGS SET, LOOP TO DO OTHER LINES.

ENDMSG

L10014: TRAP CMSG
.SBILG  GLOBAL ERROR REPORTING ROUTINE ER9101

***************
* THIS IS A GENERAL ERROR REPORTING SUBROUTINE WHICH REPORTS A MESSAGE *
* WHICH TAKES A SINGLE, 2 DIGIT DECIMAL ARGUMENT AFTER THE END OF AN *
* ASCII MESSAGE. *
* INPUTS:  R1 - VALUE TO BE PRINTED AFTER MSG AS 2 DECIMAL DIGITS. *
*          R2 - ADDRESS OF MESSAGE TO PRINT FIRST. *
* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE. *
* CALLING SEQUENCE: INCLUDE THE LABEL "ER9101" AS THE MESSAGE POINTER *
*                  PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL. *
* COMMENTS:  THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION. *
* SUBORDINATE ROUTINES USED: NONE. *
***************

BGNMSG ER9101

ER9101:

MOV @BIT06,RO ;TRY TO CLEAR THE
BIC OPTION,RO ;EXIT: ERROR REPORTING FLAG
BNE 2l ;EXIT IF FLAG NOT SET.

PRINTB @EF9006,R2,R1 ;REPORT THE STRING FOLLOWED BY THE NUMBER.

2l: ENDMG

L10015:

TRAP C1MTG
GLOBAL ERROR REPORTING ROUTINE

**Routine Description:**
- This is an error reporting subroutine which prints any BMP codes that are found in the BMP code queue, together with the number of the test that was executing at the time the BMP code was logged.
- Provided extended error reporting has been enabled.

**Inputs:**
- R1 - The address of the first message to be reported.
- R2 - The address of the next empty cell in the queue.

**Outputs:**
- The test number followed by the BMP code are printed at the operator console.

**Calling Sequence:**
- Include the label "ER9301" as the message pointer parameter in the diag super error report macrd call.

**Comments:**
- The message is printed as basic error information.
- Subordinate routines used: None.

**Code Snippet:**
```
BGNMSG ER9301
SAVE
;SAVE THE GP RS ON THE STACK.
JSA R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV @BIT06,RO ;TRY TO CLEAR THE
BIC OPTION,RO ;EXIT'D ERROR REPORTING FLAG
BNE 60$ ;EXIT IF FLAG NOT SET.
PRINTB @EF0503,R1 ;REPORT UNEXPECTED BMP CODES FOUND
MOV R1,(SP) ;MOV @EF0503,(SP)
MOV @EM9302,R5 ;GET THE START ADDRESS OF THE BMP CODE QUEUE.
MOV @EM9302,R5 ;GET THE MESSAGE TO BE REPORTED.
21: MOV (R3),R1 ;GET THE NUMBER OF THE TEST THAT WAS EXECUTING.
MOV (R3),R4 ;GET BMP CODE THAT WAS REPORTED OFF THE QUEUE.
JSR PC,501 ;GO REPORT THE BMP CODE.
CMP R3,R2 ;CHECK IF ALL CODES HAVE BEEN REPORTED.
BLO 21 ;IF IT IS NOT THE LAST BMP CODE THEN LOOP.

;CHECK IF OVERFLOW HAS OCCURRED.
;THE CONDITIONS FOR OVERFLOW ARE: THE POINTER CONTAINS THE ADDRESS OF THE
;LAST CELL IN THE QUEUE, AND A BMP CODE HAS ALREADY BEEN WRITTEN INTO THAT
;CELL.

CMP R2,ABMPCOE.-4 ;CHECK IF THE POINTER IS AT THE LAST LOCATION.
BNE 601 ;EXIT IF NOT AT THE LAST LOCATION.
TST 2(R2) ;CHECK FOR A BMP CODE IN THE LAST CELL
BEQ 601 ;EXIT IF NO OVERFLOW HAS OCCURRED, CELL EMPTY.
MOV (R3),R1 ;GET THE TEST NUMBER OFF THE QUEUE.
MOV (R3),R4 ;GET THE BMP CODE OFF THE QUEUE.
MOV @EM9303,R5 ;SELECT THE MESSAGE TO BE REPORTED.
```
2343 013306
    013306 012746 005131
    013312 012746 000001
    013316 010600
    013320 104415
    013322 062706 000004
2344 013326 004767 000002
JR PS,590
BR 601
;EXIT.
501: PRINTX @EF9301.R5,R1,R4 ;PRINT THE MESSAGE.
    MOV R4,(SP)
    MOV R1,(SP)
    MOV R5,(SP)
    MOV @EF9301,(SP)
    MOV SP,R0
    TRAP CIPNTX
    ADD 012,SP
2346 013334
    013334 010446
    013336 010146
    013340 010546
    013342 012746 005053
    013346 012746 000004
    013352 010600
    013354 104415
    013356 062706 000012
2348 013362 000207
2349 013364
    013364 004736
2350 013366
    104423
ENDMSG
2351 013366
    013366
    013366
L10016: TRAP C1MS
2353
2361
2362
2363
2364
2365
2366

.SBITL GLOBAL SUBROUTINES SECTION

; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
; THAT ARE USED IN MORE THAN ONE TEST.
; --
GLOBAL SUBROUTINE

SDTTL: GLOBAL SUBROUTINE

SUBL: 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 1 AND 2.
CLEAR RX, DMA BITS ON ALL LINES.

INPUTS:
R1 - ADDRESS OF THE REGISTERS TO ALTER.
R2 - BIT FIELDS SEI 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.
ICEST - SAVED STATES OF THE INTERRUPT ENABLE BITS.

OUTPUTS:
DEVICE REGISTERS - SPECIFIED REGISTER FIELDS ALTERED.
CSR IND, ADR, REG FIELD - DESTROYED.

CALLING SEQUENCE:
JSR PC, SUBL

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.

SUBL:: SAVE ;SAVE CONTENTS OF GPRAS $0 Thru $5.
JSR $5, PREGS ;CALL REGISTER SAVE SUBRT.

SET UP TO LOOP FOR EACH LINE:
PREPARE THE WORD TO BE DRED 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 NUMALS).

CLC
;PREPARE FOR ROTATE, "ST R5" DOES THIS BELOW.
28: ROR R3
;GET THE LINE SELECT BIT FOR THIS LINE.
BCC 41
;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.
BIC R2, R0
;OR IN THE NEW STATES OF THE FIELDS.
MOV R0, (R1)
;WRITE THE NEW REGISTER CONTENTS TO THE REG.
44: INC R5
;SET LINE NUMBER TO THE NEXT LINE.
TST R3
;CHECK FOR UNHANDLED LINES, CLEAR CARRY FLAG.
BNE 28
;LOOP IF SELECTED LINE(S) IS NOT HANDLED.
2424
2425 013436
   013436 004736
2426 013440 000207

604:  PASS
      JSR  (CSP)  
      RTS  PC

;RESTORE GPRs.
;RETURN TO PREGOS SUBRT.
;RETURN TO CALLING ROUTE.
<table>
<thead>
<tr>
<th>Address</th>
<th>Value1</th>
<th>Value2</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2484</td>
<td>013544</td>
<td>002772</td>
<td>BLT 10#</td>
<td>Loop if not at end of table yet.</td>
</tr>
<tr>
<td>2485</td>
<td></td>
<td></td>
<td>PASS</td>
<td>Restore GPR.</td>
</tr>
<tr>
<td>2486</td>
<td>013546</td>
<td>004736</td>
<td>JSR PC,(SP)</td>
<td>Return to PREGOS subr.</td>
</tr>
<tr>
<td>2487</td>
<td>013550</td>
<td>000207</td>
<td>RTS PC</td>
<td></td>
</tr>
</tbody>
</table>
SBITL GLOBAL SUBROUTINE

; CALMSL

; THIS SUBROUTINE CALIBRATES THE TIMING LOOP WHICH IS USED IN THE MSLOOP
; ROUTINE. THIS SUBROUTINE CALCULATES A VALUE FOR THE MSLCNT VARIABLE
; WHICH IS THE NUMBER OF SOFTWARE LOOPS WHICH TAKES 1 MS TO EXECUTE IN
; THE MSLOOP ROUTINE. THIS ROUTINE CALIBRATES THE COUNT BY USING THE
; LINE TIME CLOCK (LTC), SO IF NO LTC IS AVAILABLE THE DEFAULT VALUE FOR
; THE DELAY COUNT MUST BE USED.

; INPUTS:  MSLCNT - DEFAULT 1 MS DELAY LOOP COUNT VALUE, OR
; VALUE FROM PREVIOUS CALIBRATION.
; MSITCK - NUMBER OF MS PER LTC CLOCK TICK.
; TIMER1 - TIMER COUNTER CHANGED BY LTC INTERRUPT SERVICE RTN.
; CLKMRZ - NUMBER OF LTC CLICKS PER SECOND (50 OR 60).

; OUTPUTS: CARRY - SET IF LTC IS AVAILABLE, AND NEW CALIBRATION PERFORMED.
; MSLCNT - NEW 1 MS DELAY LOOP COUNT VALUE IF LTC AVAILABLE, OR
; UNCHANGED IF NO LTC IS AVAILABLE.

; CALLING SEQUENCE: JSR PC, CALMSL

; COMMENTS:

; SUBORDINATE ROUTINES CALLED: UNSDIV, OOPS.

CALMSL: SAVE
; SAVE CONTENTS OF GPRs RO THRU RS.

CLR 62h
; CLEAR THE 2ND TIME FLAG.

; SYNCHRONIZE WITH THE LTC.

MOV #1, R5
; SET OUTER LOOP COUNTER TO 1 LOOP.

; INCREASE THE VALUE LOADED INTO THIS COUNTER IF THE FOLLING LOOP FAILS ON FUTURE, FASTER PROCESSORS.

CLR RO
; CLEAR THE WAIT FOR CLOCK INT COUNTER.

MOV #1, TIMER1
; SET UP COUNT OF 1 TO SYNCH WITH LTC.

BEQ 64h
; LOOP OUT OF LOOP IF LTC HAS INTERRUPTED.

INC RO
; COUNT THIS ITERATION OF THE INNER LOOP.

BNE 41h
; LOOP IF COUNTER HAS NOT TURNED OVER.

DEC R5
; DECREMENT THE INNER LOOP COUNTER.

BGT 41h
; LOOP IF OUTER LOOP COUNTER NOT UP.

; IF WE GOT NO LTC INTERRUPT, INDICATE THAT THERE IS NO LTC AVAILABLE.
; LTC MUST BE FLAXY, OR NOT REALLY AN LTC AT ALL.

CLR CLKMRZ
; CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.

CLC
; INDICATE FAILURE FOR RETURN.

BR 60h
; BYPASS THE FOLLOWING CALIBRATION PROCEDURES.

; WE ARE NOW SYNCHRONIZED WITH THE LTC.
; SET UP FOR THE CALIBRATION LOOP.

MOV #1, R4
; WILL TEST TIMER1 IN THE LOOP BELOW.
CLR R1 ; CLEAR THE OUTER LOOP COUNTER
CLR R2 ; INDICATE TO CHECK ALL BITS OF TIMER1.
CLR R3 ; INDICATE TO CHECK FOR TIMER1 CLEAR.
MOV 01,(R4) ; LOAD TIMER1 WITH COUNT OF 1.

2550 013642 016705 166460
81: MOV MSLCNT,R5 ; LOAD MS LOOP COUNT.
101: MOV (R4),R0 ; GET THE TIMER1 VALUE.
MOV R0,64# ; SAVE WORD (LIKE IN THE REAL LOOP).
BIC R2,R0 ; SAVE ALL THE BITS.
CMP R0,R3 ; COMPARE AGAINST ZERO.
SEC ; SET CARRY IN CASE OF SUCCESS.
BEQ 121 ; EXIT LOOP IF TIMER1 HAS CLEARED.
DEC R5 ; COUNT DOWN THE INSIDE MS LOOP COUNT.
BNE 101 ; LOOP IF MS NOT UP.
DEC R1 ; DECREMENT THE MS TIME COUNT.
BNE 81 ; KEEP LOOPS.
JSR PC,00PS ; WE OVERFLOWED, SOMETHING IS WRONG, ABORT.

2562 ; WE HAVE NOW HAVE LOOP COUNT INFORMATION FOR ONE CLOCK TICK.
; WE HAVE NEGATIVE OF NUMBER OF OUTER LOOPS IN R1, EACH IS MSLCNT INNER LOOPS.
; WE HAVE THE PORTION OF THE LAST OUTER LOOP NOT EXECUTED, IN R5.
; NOW WE CALCULATE THE TOTAL NUMBER OF INNER LOOPS EXECUTED.

2567

2568 013700 005401
121: NEG R1 ; GET NUMBER OF OUTER LOOPS.
MOV MSLCNT,R2 ; GET THE NUMBER OF INNER LOOPS PER OUTER LOOP.
MOV R2,R5 ; COPY NUMBER OF LOOP FOR MULTIPLY.
SUB R5,R2 ; CALC # OF INNER LOOPS DONE IN LAST OUTER LOOP
MOV R2,R4 ; AND ADD TO ACCUMULATOR LSWORD.
CLR R5 ; CLEAR ACCUMULATOR MSWORD.
141: DEC R1 ; CHECK R1 FOR 0 CONDITION
BMI 161 ; SKIP MULTIPLICATION IF ZERO
ADD R3,R4 ; MULTIPLY NUMBER OF INNER LOOP
ADC R5 ; LOOPS PER OUTER LOOP BY
BR 141 ; NUMBER OF OUTER LOOPS PERFORMED.

2579 ; DIVIDE THE TOTAL NUMBER OF INNER LOOPS BY THE NUMBER OF MS PER LTC TICK.

2581

2582 013730 016701 166370
161: MOV MSTICK,R1 ; # OF MS PER LTC TICK IS DIVISOR.
MOV R4,R3 ; LSWORD OF LOOP COUNT IS LSWORD OF DIVIDEND.
MOV R5,R2 ; MSWORD OF LOOP COUNT IS MSWORD OF DIVIDEND.
JSR PC,UNS DIV ; DIVIDE NUMBER OF LOOPS BY MS PER LTC TICK.
BCS 181 ; BYPASS OOPS IF WE'RE OK.
JSR PC,00PS ; CLOCK ROUTINES ARE NOT LONG ENOUGH, OR BUG.
181: MOV R1,MSLCNT ; SET NEW VALUE FOR MS LOOP COUNT.
JSR PC,00PS ; SET THE 2ND ITERATION FLAGS IF 1ST ITERATION.
BNE 21 ; BRANCH IF ONLY ONE ITERATION DONE.
SEC ; SET THE SUCCESS FLAG FOR EXIT.

2592

2593 013766
601: PASS JSR PC,00PS ; RESTORE GPR'S.

2594 013770 000207
RTS PC ; CARRY SUCCESS FLAG, SET IF SUCCESS.

2595

2596 013772 000000
621: .WORD 0 2ND CALIBRATION ITERATION FLAGS.

2597 013774 000000
641: .WORD 0 DUMMY WORD FOR STORAGE OF THE READ WORD.
GLOBAL SUBROUTINE

CLE:GLOBAL SUBROUTINE

******************************************************************************

- CHECK IF CHARACTER IS A BMP CODE.
- IF A BMP CODE IS DETECTED, IT WILL BE SAVED ON THE QUEUE TO BE REPORTED LATER. THE CARRY IS USED AS A FLAG TO INDICATE A CODE HAS BEEN FOUND.
- INPUTS: R2 - CONTAINS THE DATA TO BE CHECKED.
- OUTPUTS: R1 - CONTAINS THE MESSAGE TO BE REPORTED.
- ERRBLK - CONTAINS THE ERROR REPORTING ROUTINE.
- CARRY BIT IS USED TO INDICATE A BMP CODE FOUND, CARRY SET.
- CALLING SEQUENCE: JSR PC, CHKMP
- COMMENTS:
- SUBORDINATE ROUTINES CALLED: SAVBMP.

******************************************************************************

CHKMP:

SAVE CONTENTS OF GPRs RO THRU RS.
RS, PREGS, CALL REGISTER SAVE SUBR.

MOV 0170301, R0
SET UP THE FLAGS OF A BMP CODE.

BIC R2, R0
TRY TO CLEAR THE BMP CODE FLAGS.

BNE 24
IF NOT A BMP CODE, EXIT WITH FAILURE.

JSR PC, SAVBMP
SAVE THE BMP CODE ON THE QUEUE.

MOV @EM5303, R1
PASS THE MESSAGE TO BE REPORTED.

MOV @ER1603, ERRBLK
SELECT THE CORRECT ERROR REPORTING ROUTINE.

SEC
PASS FLAG TO INDICATE SUCCESS, BMP CODE FOUND.

BRA 601
EXIT.

24: CLC
PASS FLAG TO INDICATE FAILURE.

601: PASS R1
PUT R1 IN STACK SLOT.

MOV R1, RISLOT(SP)
RESTORE GPRs, EXCEPT

JSR PC, (SP+4)
RETURN TO PREGOS SUBR.

R1 - CONTAINS THE ADDRESS OF ERROR MESSAGE.
CARRY BIT - SET INDICATES SUCCESS.

RTS PC
**GLOBALSUB** **SUBROUTINE** **CBX:CMX**

- **INSTRUCTIONS**
  - THIS SUBROUTINE IS USED IN THE RXTIMER TEST AND IT CHECKS THAT THE RX-INT WAS DELAYED BY 1% - 20% OF THE VALUE SET IN THE RXTIMER REG.
  - IF THE ACTUAL DELAY TIME IS NOT WITHIN THIS MARGIN THEN AN ERROR IS REPORTED.

- **INPUTS:**
  - R1 - REMAINING NUMBER OF MILLI SECS OF THE TIME OUT VALUE.
  - R2 - RXTIMER VALUE.
  - R3 - TIME-OUT VALUE BEING 4*(RXTIMER VALUE + 1).
  - R4 - ERNRB - SET TO ERROR NUMBER OF FIRST ERROR IN THIS ROUTINE.
  - R5 - ERRBLK - SET UP BY THIS ROUTINE.
  - R6 - EXDERR - "EXIT ON ERROR" FLAG.

- **OUTPUTS:**
  - ERRBLK - MAY BE ALTERED.
  - EXDERR - "EXIT ON ERROR" FLAG SET IF AN ERROR DETECTED AND EXTENDED ERROR REPORTING NOT REQUESTED.

- **CALLING SEQUENCE:**
  - JSR PC,CBX:CMX

- **COMMENTS:**
  - THIS ROUTINE REPORTS ONE ERROR WITH THE NUMBER IN ERNRB.
  - SUBORDINATE ROUTINES USED: ER95901,UNSDIV.

---

**CBX:CMX:**

- **SAVE**
  - MOV R2,R4
  - INC R4
  - ASL R4
  - ASL R4
  - SUB R1,R4

- **CALCULATE 20% OF THE RXTIMER VALUE.**
  - MOV #5,R1
  - MOV R2,R5
  - CLR R2
  - JSR PC,UNSDIV
  - MOV R3,R2
  - MOV R3,R5
  - SUB R1,R3
  - ADD R1,R5
  - ADD #2,R5

- **CHECK IF THE RX-INT OCCURRED WITHIN THE CALULCATED PERMISSIBLE LIMITS.**
  - CMP R5,R4
  - CMP R3,R4
  - CMP R5,R4
  - CMP #2,R4

- **REPORT THE ERROR, INCORRECT DELAY ON RX INT.**
2630 014124 010201 24: MOV R2,R1 ;PASS THE EXPECTED VALUE OF THE RX INT DELAY.
2691 014126 010402 MOV R4,R2 ;PASS THE ACTUAL VALUE OF THE RX INT DELAY.
2692 014130 012703 007003 MOV #EM5802,R3 ;PASS THE MESSAGE.
2693 167714 RXTIMER BAD, INCORRECT MILLI SEC DELAY ON RX INT.
2694 014134 012767 012170 ERROR SET UP THE ERROR REPORTING ROUTINE.
2695 167714 MOV @ER5901,ERRBLK TRAP C1ERROR
2696 014142 104460 2697
2698 014144 032767 000100 166036 BIT #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
2699 014152 001003 BNE 601 ;AVOID SETTING THE FLAG IF IT HAS.
2700 014154 012767 000001 166070 MOV #1,EXOERR ;SET THE "EXIT ON ERROR" FLAG.
2701 014162 601 : PASS JSR PC,(SP) ;RETURN TO PREGOS SUBRT.
2702 014164 000207 RTS PC ;RETURN FROM SUBROUTINE.
GLOBAL SUBROUTINE

.SBTL GLOBAL SUBROUTINE

CKTRAP

***********************************************************************

;** CHECK TRAP ROUTINE
;** THIS SUBROUTINE IS USED TO CHECK FOR A BUS TIME OUT TRAP (004 TRAP)
;** WHICH IS CAUSED BY AN ACCESS TO A NON EXISTENT MEMORY OR I/O LOCATION.
;** IF THE TRAP DOES NOT OCCUR, THIS ROUTINE RETURNS A SUCCESS INDICATION.

;** INPUTS:
;** RO SOURCE ADDRESS FOR MOVE
;** R1 DESTINATION ADDRESS FOR MOVE
;** (RO) SOURCE FOR THE MOVE

;** OUTPUTS:
;** (R1) - WRITTEN TO THE CONTENTS OF (RO).
;** CARRY FLAG - SET ON RETURN IF NO 004 TRAP DETECTED.
;** TP4FLG - NONZERO IF TRAP OCCURRED, CLEARED OTHERWISE.

;** CALLING SEQUENCE: JSR PC,CKTRAP

;** COMMENTS: IF THIS SUBROUTINE CAUSES A TRAP, EITHER THE ADDRESS WHICH
;** IS LABELED ADRPTR WILL BE THE TRAP PC ADDRESS ON THE STACK.

;** SUBORDINATE ROUTINES CALLED: NONE.

***********************************************************************

CKTRAP:: SAVE

JSR R5,PREG5S ;CALL REGISTER SAVE SUBRT.

CLR TP4FLG ;CLEAR THE 004 TRAP FLAGS.

MOV (RO),(R1) ;PERFORM THE MOVE IN QUESTION.

ADRPTR:: IST TP4FLG ;CHECK FOR OCCURRENCE OF TRAP.

SEC ;INDICATE SUCCESS.

BEQ 60$ ;EXIT WITH SUCCESS IF TRAP DID NOT OCCUR.

CLC ;INDICATE FAILURE.

60$: PASS ;RESTORE GPRS.

JSR PC,PREGS ;RETURN TO PREG5S SUBRT.

RTS PC
.SBITL GLOBAL SUBROUTINE CLNAST

***********************************************************************

* CLEAN RESET OF THE DEVICE UNDER TEST —
*    - THIS SUBROUTINE IS USED TO RESET THE DUT TO A KNOWN STATE.
*    - THE DUT'S SELF-TEST IS SKIPPED, AND THE FIFO IS PURGED OF ANY ERROR
*    - CODES, ETC.
*    - IF THE RESET DOES NOT SUCCESSFULLY COMPLETE, THEN THE CARRY BIT IS
*      PASSED BACK TO THE CALLING ROUTINE (CLEAR).
*    
*    INPUTS:    CSR - CONTAINS THE ADDRESS OF THE CSR
*                TXIFCA - CONTAINS ADDRESS OF DUT DMA BUFFER COUNT REGISTER.
*                ERRBR - ERROR NUMBER FOR POSSIBLE ERROR REPORT.
*                ERRTP - ERRTP, ERRBR, AND ERRMSG SET UP CORRECTLY.
*    
*    OUTPUTS:   THE DUT PERFORMS ITS RESET FUNCTION INTO A KNOWN STATE.
*                CARRY - CLEAR INDICATES THE TEST IS TO BE ABORTED.
*                ERRBR - VALUE MAY BE DESTROYED.
*                IESTAT - TX AND RX INTERRUPT FLAGS ARE CLEARED.
*                TX AND RX INTERRUPT ENABLE BITS IN THE DUT'S CSR ARE CLEARED.
*    
*    CALLING SEQUENCE:  JSR PC,CLNAST
*    
*    COMMENTS:  THIS SUBROUTINE CAN REPORT ERRORS WITH NUMBERS ERRBR.
*                THIS ROUTINE DOES NOT DESTROY THE VALUE OF ERRBR.
*    
*    SUBORDINATE ROUTINES CALLED: DELAY, MSGET, PUFIFO, RESET T.

***********************************************************************

CLNAST:  SAVE JSR R5, PREGOS ; CALL REGISTER SAVE SUBRT.

*  RESET THE DUT.
*  THIS ROUTINE REPORTS ERRORS WITH NUMBERS FROM ERRBR THRU ERRBR+2.
*  
*  JSR PC, RESET T ; RESET THE DUT TO A KNOWN STATE.
*  BCC 601 ; EXIT ROUTINE WITH ABORT TEST INDICATOR.
*  
*  PURGE THE FIFO OF ERROR CODES, SAVE ANY BMP CODES FOUND.
*  
*  JSR PC, PUFIFO ; PURGE THE FIFO.

601:  PASS JSR PC, B(S5) ; RETURN TO PREGOS SUBRT.
      RTS PC ; CARRY BIT: IF CLEAR, THEN ABORT THE TEST.
GLOBAL SUBROUTINE - CLR16W

* THIS SUBROUTINE CLEAR 16 WORDS STARTING WITH THE SPECIFIED WORD.
* INPUTS: RO - ADDRESS OF THE FIRST WORD TO CLEAR.
* OUTPUTS: (RO) TO (RO+15) - 16 WORDS OF MEMORY ARE CLEARED TO 0.
* CALLING SEQUENCE: JSR PC, CLR16W

* COMMENTS:
* SUBORDINATE ROUTINES CALLED: NONE.

CLR16W: ; SAVE
        JSR R5, PREGOS ; CALL REGISTER SAVE SUBRT
        MOV @16, R1 ; SET THE LOOP COUNTER TO 16.
        CLR (R0) ; CLEAR A WORD OF MEMORY.
        DEC R1 ; COUNT THIS LOOP.
        BNE 2$ ; LOOP IF NOT 16 WORD CLEARED.
        PASS 60$ ; RESTORE GPRTS.
        JSR PC, @SP, PREGOS ; RETURN TO PREGOS SUBRT.
SBITL GLOBAL SUBROUTINE - DELAY

*******************************************************************************

<table>
<thead>
<tr>
<th>#</th>
<th>GLOBAL SUBROUTINE - DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>THIS SUBROUTINE IS USED TO DELAY A VARIABLE NUMBER OF MILLI-SECONDS.</td>
</tr>
<tr>
<td>2</td>
<td>INPUTS: R4 - CONTAINS THE NUMBER OF MS TO DELAY.</td>
</tr>
<tr>
<td>3</td>
<td>MSCLNT.</td>
</tr>
<tr>
<td>4</td>
<td>OUTPUTS: NONE.</td>
</tr>
<tr>
<td>5</td>
<td>CALLING SEQUENCE: JSR PC,DELAY</td>
</tr>
<tr>
<td>6</td>
<td>COMMENTS: IF NO HARDWARE CLOCK INTERRUPTS ARE OCCURING, CONTROL-CC WILL</td>
</tr>
<tr>
<td>7</td>
<td>NOT BE HONORED FOR THE DURATION OF THE DELAY.</td>
</tr>
<tr>
<td>8</td>
<td>SUBORDINATE ROUTINES CALLED: NONE.</td>
</tr>
</tbody>
</table>

*******************************************************************************

DELAY:: SAVE JSR R5, PREGO$ ;CALL REGISTER SAVE SUBRT.
    MOV R4,R1 ;SAVE CONTENTS OF GPRS R0 THRU R5.
    MOV 0-1,R2 ;PASS NUMBER OF MS DELAY AS TIME-OUT VALUE.
    CLR R3 ;TELL MSLOOP ROUTINE TO CHECK ALL BITS.
    MOV 4621,R4 ;TELL MSLOOP RTN TO CHECK FOR ALL BITS CLEAR.
    MOV R4,MSLOOP ;TELL MSLOOP TO CHECK DUMMY NON ZERO WORD.
    JSR PC,MSLOOP ;DELAY THE REQUESTED # OF MS.
    BCC 601 ;EXIT ROUTINE IF WE TIMED-OUT.
    JSR PC,OOPS ;EXIT ROUTINE IF WE TIMED-OUT, BAD PROGRAM OR HOST MACHINE.
    RTS PC ;RESTORE GPRS.
    JSR PC,(SCSP) ;RETURN TO PREGO$ SUBRT.
    62$ ; DUMMY, NON-ZERO WORD.
GLOBAL SUBROUTINE

; INITIATE DMA TRANSMISSION ROUTINE.
; THIS ROUTINE WRITES THE DMA PARAMETERS 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 IUT CSR.
; IESTAT - STORAGE FOR STATES OF THE INTERRUPT ENABLE BITS.
; TXADR1 - CONTAINS ADDRESS OF DMA TX BUFFER ADDRESS REG #1.
; TXADR2 - 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).
; OUT TBUFFAD1 - LS 16 BIT OF DMA BUFFER ADDRESS (INITIALIZED).
; OUT TBUFFAD2 - MS 6 BITS OF DMA BUFFER ADDRESS (INITIALIZED).
; DMA START BIT SET.
; OUT TBUFFCT - DMA BUFFER CHARACTER COUNT (INITIALIZED).

; CALLING SEQUENCE:

; JSR PC,DODMA

; COMMENTS:
; THIS ROUTINE ASSUMES MEMORY MANAGEMENT IS DISABLED AND
; CLEARS THE TWO MSB OF THE DMA ADDRESS, I.E. BITS 0 AND 1
; OF THE TBUFFAD2 REG.

SUBORDINATE ROUTINES CALLED: NONE.


DODMA::: SAVE
    JSR R5,PREG5
    CALL REGISTER SAVE SUBRUT.
    MOV #200,R4
    ; PREPARE TO CLEAR UPPER 6 BITS OF DMA BUFF ADR.

    ; WRITE THE DMA PARAMETERS OUT TO THE OUT DMA REGISTERS.
    ; DISABLE INTERRUPTS.
    ; SET UP OUT CSR IND.ADR.REG FIELD.
    ; WRITE THE DMA TRANSMIT CHARACTER COUNT.
    ; WRITE THE LEAST SIGNIFICANT 16 BITS OF THE DMA BUFFER START ADDRESS.
    ; WRITE THE MOST SIGNIFICANT 6 BITS OF THE ADDRESS.
    ; SETTING THE DMA.START BIT, AND INITIATING THE DMA TRANSMISSION.

61: GETPRI R5
    ; GET THE PRESENT PROCESSOR PRIORITY.
    TRAP CICPRI
    MOV R0,R5

    SETPRI #PRI07
    ; DISABLE ALL HARDWARE INTERRUPTS.
    MOV #PRI07,R0
    TRAP CISPRI

    BIS IESTAT,R1
    ; PREPARE FOR SETUP OF LINE NUMBER IN OUT CSR.
    MOV R1,WCSRA
    ; SET UP THE OUT CSR IND.ADR.REG FIELD.
    TSTB @TXAD2A
    ; TEST THE OUT DMA START BIT.
    CLC
    ; INDICATE FAILURE IN CASE DMA:HO BIT IS SET.
    BMI 601
    ; EXIT WITH FAILURE IF DMA:HO BIT IS SET.
    MOV R3, @TXBFCA
    ; WRITE THE DMA CHARACTER COUNT.
    MOV R2, @TXAD1A
    ; WRITE THE LS 16 BITS OF BUFFER ADDRESS.
    MOVB R4, @TXAD2A
    ; WRITE MS 6 BITS OF ADR AND START DMA TX.
2938 014456
 014456 010500
 014460 104441
2939 014462 000261
2940 014464 004736
2941 014466 000207

     SETPRI R5
     ; RESTORE THE PROCESSOR PRIORITY.
     MOV   R5,RO
     TRAP  CISPR1

     SEC
     ; INDICATE SUCCESS.

     601:  PASS JSR
     ; RESTORE GPRS.
     PC,B(ESP)* ; RETURN TO PREVIOUS SUBRT.

     RTS PC
     ; CARRY SUCCESS FLAG (SET IF SUCCESS).
GLOBAL SUBROUTINE

**FINACT**

```
**FUNCTION**
**FINACT**

**DESCRIPTION**

- FIND FIRST ACTIVE LINE
- THIS SUBROUTINE CALCULATES THE NUMBER OF THE FIRST ACTIVE LINE THAT IS FOUND IN THE ACTIVE LINE BIT MAP ACTLNS.

**INPUTS:**
- ACTLNS - CONTAINS THE ACTIVE LINE BIT MAP.

**OUTPUTS:**
- R1 - CONTAINS THE NUMBER OF THE FIRST ACTIVE LINE.
- R5 - CONTAINS THE BIT MAP REPRESENTATION OF THE ACTIVE LINE.
- CARRY SET INDICATES SUCCESS.

**CALLING SEQUENCE:**

```
JSR PC,FINACT
```

**COMMENTS:**

- NO SUBORDINATE ROUTINES CALLED.

---

```
FINACT:: SAVE

JSR R5,PREGOS ;CALL REGISTER SAVE SUBRT.

FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.

CLR R1 ;CLEAR THE LINE NUMBER COUNTER.

MOV @NUMNLS,R3 ;GET MAX LINE NUMBER.

MOV ACTLNS,RO ;GET THE ACTIVE LINE BIT MAP.

MOV #1,R5 ;SET UP A LINE BIT MASK.

MOV R5,R5 ;LOOK FOR AN ACTIVE LINE.

28: BNE 41 ;BRANCH TO BEGIN TEST IF A LINE HAS BEEN FOUND.

ASL R5 ;SHIFT THE BIT MASK FOR THE NEXT LINE.

INC R1 ;INCREMENT THE LINE NUMBER COUNTER.

CMP R1,R3 ;CHECK IF ALL LINES HAVE BEEN TRIED.

BLT 28 ;LOOP TO TRY THE NEXT LINE.

CLC R1,R3 ;CLEAR CARRY BIT, NO ACTIVE LINE FOUND.

BA 601 ;EXIT WITH FAILURE.

41: SEC R1,R3 ;SET CARRY, SUCCESS.

601: PASS R1,R5 ;RESTORE GPRS, EXCEPT

MOV R1,RJLSLOT(SP) ;PUT R1 IN STACK SLOT.

MOV R5,R5,RJLSLOT(SP) ;PUT R5 IN STACK SLOT.

JSR PC,(SP) ;RETURN TO PREGOS SUBRT.

R1 - CONTAINS THE NUMBER OF FIRST ACTIVE LINE.

R5 - CONTAINS THE BIT MAP OF THE ACTIVE LINE.

CARRY - SET INDICATES SUCCESS.
```

```
GLOBAL SUBROUTINE

INDATP

SBTTL GLOBAL SUBROUTINE

INDATP

1* **************************************************************************************************************************
1* - INITIALISE DATA PATTERN -
1* - THIS SUBROUTINE IS USED TO INITIALISE AN INCREMENTAL BYTE DATA PATTERN
1* - IN THE GENERAL BUFFER AREA.
1* - THE DATA PATTERN WILL BE SEQUENTIAL FROM 0 TO 255 (DECIMAL).
1* -
1* INPUTS:  BUFHAS - ADDRESS OF THE START OF THE GENERAL BUFFER AREA.
1* -  BUFHID - ADDRESS OF THE 255 TH LOCATION.
1* -
1* OUTPUTS: THE FIRST 255 LOCATIONS OF THE GENERAL BUFFER AREA CONTAIN DATA
1* -
1* CALLING SEQUENCE:  JSR  PC,INIDATP
1* -
1* COMMENTS:
1* -
1* SUBORDINATE ROUTINES CALLED:  NONE.
1* -

INDATP:: SAVE

JSR  RS,PREG05  ;CALL REGISTER SAVE SUBRT.

014550 014550 004567 167304

014550 012702 002650

014560 005000

014560 110322

014560 005203

014560 020227 003250

014572 103773

014574 103773

014574 004735

014576 000207

MIV @BUFHAS,R2  ;INITIALIZE THE DATA PATTERN IN THE GENERAL

014554 012702 002650

CLR R3  ;DATA BUFFER TO A 256 BYTE PATTERN.

014560 005000

MOV @BUFHID \(R2)\*  ;SELECT THE NEXT CHARACTER.

014560 005203

INC R3

014560 020227 003250

CMP R2,@BUFHID  ;CHECK IF WE HAVE 256 DATA PATTERNS.

014560 005203

BLO 21

014560 020227 003250

PASS

014572 103773

JSR  PC,(@SP)\*  ;RETURN TO PREG05 SUBRT.

014574 103773

RTS  PC
GLOBAL SUBROUTINE INDXP

.8BITL GLOBAL SUBROUTINE INDXP

; :: INITIALSE DATA PATTERN WITHOUT XON OR XOFF .
; :: THIS SUBROUTINE IS USED TO INITIALISE AN INCREMENTAL BYTE DATA PATTERN .
; :: IN THE GENERAL BUFFER AREA .
; :: THE DATA PATTERN WILL BE FROM 0 TO 255, BUT WILL EXCLUDE THE FOLLOWING .
; :: TWO CHARACTERS: (ASCII DC1, DC3) XON AND XOFF. THIS WILL CAUSE THE .
; :: LAST TWO DATA CHARACTERS TO BE THE SAME AS THE FIRST TWO .
; :: BUFIN - ADDRESS OF THE 255th LOCATION .
; :: OUTPUTS: THE FIRST 255 LOCATIONS OF THE GENERAL BUFFER AREA CONTAIN DATA .
; :: CALLING SEQUENCE: JSR PC,INDXP .
; :: COMMENTS: .
; :: SUBORDINATE ROUTINES CALLED: NONE .

; :: INITIALSE 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 .

INDXP:  ; SAVE
        ; SAVE CONTENTS OF GPRS R0 THRU RS.
        JSR RS,PREGOS  ; CALL REGISTER SAVE SUBRT.

        ; INITIALIZE THE 256 BYTE DATA PATTERN .
        ; ENSURE THE DATA PATTERN IS FREE FROM XON'S OR XOFF'S TO PREVENT ERRORS .
        .

        ; MOVE   R0,BUFBAS,R2  ; INITIALIZE THE DATA PATTERN IN THE GENERAL
        MOV   R0,BUFBAS,R2
        CLEAR R3  ; DATA BUFFER TO A 256 BYTE PATTERN .
        CLR   R3

28:  ; MOVE   R3,(R2) ,
        MOV   R3,(R2)

        ; INC   R3  ; SELECT THE NEXT CHARACTER .
        INCR   R3

        ; CMPB #21,R3  ; CHECK FOR AN XON CHARACTER .
        CMPB #21,R3

        ; BNE   41  ; BRANCH IF CHAR NOT AN XON .
        BNNE   41

        ; INC   R3  ; FORCE THE NEXT CHARACTER .
        INCR   R3

48:  ; CMPB #23,R3  ; CHECK FOR AN XOFF CHARACTER .
        CMPB #23,R3

        ; BNE   61  ; BRANCH IF NOT AN XOFF CHARACTER .
        BNNE   61

        ; INC   R3  ; FORCE THE NEXT CHARACTER .
        INCR   R3

64:  ; CMP   R2, #BUFIN  ; CHECK IF WE HAVE 256 DATA PATTERNS .
        CMP   R2, #BUFIN

        ; BLO   21  ;
        BLONE  21

601: ; PASS
        PSE   1

        ; JSR   PC,(SP) ,
        JSR   PC,(SP),

        ; RETURN TO PREGOS SUBRT.
        RTS   PC
**GLOBAL SUBROUTINE**

```
SBTL GLOBAL SUBROUTINE  LINBIT
|
| ******************************************************************************
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | **- LINE NUMBER TO BIT MAP CONVERSION SUBROUTINE**
| | **- THIS SUBROUTINE IS USED TO GENERATE A BIT MAP (ONE BIT OF 16 SET)**
| | **- BASED ON A LINE NUMBER (RANGE: 1 TO 16). ONLY THE LS 4 BITS OF THE**
| | **- LINE NUMBER WORD ARE USED, THE OTHERS ARE MASKED OUT (SO UNMASKED**
| | | **- MSBYTES OF OUT CSRS CAN BE PASSED TO THIS ROUTINE WITHOUT ERROR).**
| | **- INPUTS:**
| | | **- R1 - LINE NUMBER (ONLY LS 4 BITS USED, OTHERS DISREGARDED).**
| | | **- BITBL - BASE LABEL OF A 16 WORD BIT TABLE.**
| | **- OUTPUTS:**
| | | **- RO BIT MAP, BIT CORRESPONDING TO LINE NUMBER IS SET:**
| | | | **- IF LINE NUMBER IS 3, THEN BITS IS SET, ETC.**
| | **- CALLING SEQUENCE:**
| | | **- JSR PC, LINBIT**
| | **- COMMENTS:**
| | | **- NO CHECKING IS PERFORMED TO VERIFY THAT THE LINE NUMBER IS**
| | | **- A LEGAL LINE NUMBER FOR THE OUT (IE - LESS THAN NUMLN).**
| | | **- NOTE: THE LINE NUMBER IS NOT DESTROYED OF ALTERED, SO THIS**
| | | **- ROUTINE CAN BE USED EASILY IN LOOPS.**
| | **- SUBORDINATE ROUTINES CALLED: NONE.**
| ***********************************************************************
```

LINBIT:: SAVE JSR R5,R5,PREGS |CALL REGISTER SAVE SUBRT.
| BIC @177760,R1 |MASK OUT ALL BUT 4 LSBITS OF THE LINE #.
| ASL R1 |MULTIPLY LINE # BY 2 TO GET WORD TABLE OFFSET.
| MOV BITBL(R1),RO |GET THE SINGLE BIT BIT MAP.
| 609: PASS RO |RESTORE GPRS, EXCEPT THE FOLLOWING.
| MOV RD,R0,SP,(SP) |PUT RO IN STACK SLOT.
| JSR PC,RO,SP+ |RETURN TO PREGS SUBRT.
| RTS PC |RO BIT MAP WITH LINE # BIT SET.

```

3061 3062 3063 3064 3065 3066 3067 3068 3069 3070 3071 3072 3073 3074 3075 3076 3077 3078 3079 3080 3081 3082 3083 3084 3085 3086 3087 3088 3089 3090 3091
GLOBAL SUBROUTINE  - MAPCNT

;**  **********************************************************************
;**  | COUNT BITS IN BIT MAP ROUTINE |
;**  | THIS SUBROUTINE COUNTS THE NUMBER OF BITS WHICH ARE SET IN A BIT MAP. |
;**  | | INPUTS:  R2 - THE BIT MAP FOR WHICH TO COUNT THE BITS. |
;**  | | OUTPUTS: R2 - COUNT OF THE NUMBER OF BITS THAT WERE SET. |
;**  | | CALLING SEQUENCE:  JSR   PC,MAPCNT |
;**  | | COMMENTS: |
;**  | | SUBORDINATE ROUTINES CALLED: NONE. |
;**  **********************************************************************

;MAPCNT::  SAVE   JSR   R5,PREGOS  ;CALL REGISTER SAVE SUBRT.
       MOV   R2,R1
       BEQ   60:
       CLR   R2
       SEC   R2
       ;CLEAR THE BIT COUNT.
       ;COUNT THE LAST BIT TO BE SHIFTED OUT.
       ;EXIT WITH ZERO IF NO BITS ARE SET IN MAP.
       ;COUNT THE BIT IF IT WAS SET.
       ;SHIFT ANOTHER BIT OUT OF THE MAP.
       ;LOOP IF ALL BITS NOT SHIFTED OUT OF MAP.
       2$:  ADC   R2
       ASL   R1
       BNE   2:$
       ;RESTORE GPRS, EXCEPT THE FOLLOWING:
       ;PUT R2 IN STACK SLOT.
       ;RETURN TO PREGOS SUBRT.
       60$:  PASS  R2
               MOV   R2,R2,SLT(SP)  ;PUT R2 IN STACK SLOT.
               JSR   PC,(SP)+
               RTS   PC
               ; R2 - COUNT OF BITS SET IN BIT MAP.
GLOBAL SUBROUTINE

***********************************************************************
! 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 TRACKS IN WHICH BIT 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 THERE AFTER.
! 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 FIELD TO 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
! WAIT 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.

MLGET:

: SAVE
JSR R5,PREGOS 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.

: MOV R2, R3

: BIC R2, R3

: HANDLE THE TEST AND EXIT IF WE HAVE A 0 TIME-OUT VALUE.

: MOV R0,62

: SEC

: BEQ 6:

: TST R1

: BNE 2:

: MOV (R4), R0

: MOV R0,62

: BIC R2, R0

: CMP R0, R3

: SEC

: BEQ 6:

: TST R1

: BNE 2:

: MOV (R4), R0

: MOV R0,62

: BIC R2, R0

: CMP R0, R3

: SEC

: BEQ 6:
CLC
BR 61
;INDICATE FAILURE (TIME-OUT).

;NON-ZERO TIME-OUT VALUE. LOOP, WAITING FOR CONDITION OR TIME OUT.

21: MOV MSLCNT,R5
;LOAD MS LOOP COUNT.

41: MOV (R4),RO
;GET THE WORD TO TEST.

MOV RO,621
;SAVE WORD IN CASE THIS IS THE LAST.

BIC R2,RO
;MASK OUT UNUSED BITS OF WORD.

CMP RO,R3
;COMPARE AGAINST DESIRED STATE WORD.

SEC
;SET CARRY IN CASE OF SUCCESS.

BEQ 64
;EXIT WITH SUCCESS IF WORDS ARE EQUAL.

DEC R5
;COUNT DOWN THE INSIDE MS LOOP COUNT.

BNE 41
;LOOP IF MS NOT UP.

DEC R1
;DECREMENT THE MS TIME COUNT.

BNE 23
;IF TIME NOT UP, LOOP TO COUNT ANOTHER MS.

CLC
;CLEAR CARRY, WE TIMED-OUT.

;HAVE EITHER FOUND CONDITION, OR TIMED-OUT (POSSIBLY FROM 0 TIME OUT VALUE).

;RESTORE THE LAST CONTENTS READ FROM THE TEST WORD. EXIT ROUTINE.

63: MOV 621,RO
;PASS OUT THE LAST READ WORD.

60: PASS RO,R1
;RESTORE GPRS, EXCEPT THE FOLLOWING:

MOV RO,RO SLOT(SP)
;PUT RO IN STACK SLOT.

MOV R1,R1 SLOT(SP)
;PUT R1 IN STACK SLOT.

JSR PC,(SP)
;RETURN TO PREGOS SUBRT.

;RO - LAST READ WORD CHECKED FOR CONDITION.

;R1 - REMAINING TIME (0 IF TIME OUT OCCURED).

;CARRY - SET IF SUCCESS, CLEAR IF TIME OUT.

;LOCAL STORAGE.

62: .WORD 0
;STORAGE FOR THE LAST READ WORD.
GLOBAL SUBROUTINE

.SBTIL GLOBAL SUBROUTINE MSLOOP

******************************************************************************

- TEST LOOP SUBROUTINE

- 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.

- 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:
  CARRY - SUCCESS FLAG (SET IF CONDITION IS MET BEFORE TIME-OUT).

- CALLING SEQUENCE:
  JSR PC,MSLOOP

- 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: MSLGET.

******************************************************************************

MSLOOP: SAVE
JSR R5,PREGOS ;CALL REGISTER SAVE SUBRT.

; CALLING THE MSLGET ROUTINE FROM THE MSLOOP ROUTINE ISOLATES THE CALLER OF
; MSLOOP FROM THE RETURNED TEST WORD AND REMAINING TIME-OUT VALUES.

; JSR PC,MSLGET ;CALL THE MULTI-PURPOSE MS LOOP AND SEARCH RTN.

PASS 60:

; RESTORE GPRS.
;
;
RTS PC
; CARRY - SET IF SUCCESS, CLEAR IF TIME OUT.
GLOBAL SUBROUTINE

**OOPS**

---

.** PROGRAM ABORT SUBROUTINE

** THIS SUBROUTINE IS USED TO ABORT THE CURRENT PROGRAM WHEN A FATAL ERROR IS DETECTED IN THE PROGRAM OR IF THE HOST SYSTEM HARDWARE OR SOFTWARE BUG IS ENCOUNTERED. PRINTER MESSAGES ARE PRINTED GIVING SOME INFORMATION ABOUT THE NATURE OF THE ABORT.

** INPUTS: **

R1 - ERROR CODE GIVING REASON FOR ABORT.

** OUTPUTS: **

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 GPSCR TO R5.

JSR RS,PREG0S, ; CALL REGISTER SAVE SUBRT.

; REPORT "SOFTWARE BUG ENCOUNTERED." ERROR.

ERRS 101,EM0101

TRAP C1ERSSF

.WORD 101

.WORD EM0101

.WORD 0

; REPORT "SOFTWARE HUNG, WAITING FOR A CONTROL-C."

PRINTF #EM0102

MOV @EM0102, (SP)

MOV @1, (SP)

MOV @SP,R0

TRAP C1PRINTF

ADD @4,SP

BR 10

BREAK

; LOOK FOR OPERATOR CONTROL-C INPUT.

TRAP C1BRK

; INFINITE LOOP.

PASS

JSR PC,8CSP

; RETURN TO PREG0S SUBRT.

RTS PC

; ROUTINE IN THE FUTURE, SO BE CONSISTANT.

EM0101: .ASCIZ /MOST COMPUTER HARDWARE OR SOFTWARE BUG ENCOUNTERED./
EM0102:: ASCIZ "PROGRAM HUNG, WAITING FOR A CONTROL-C." ********************

015202  105  104  056
015207  000
015210  045  116  045
015213  101  120  122
015216  117  107  122
015221  101  115  040
015224  110  125  116
015227  107  054  040
015232  127  101  111
015235  124  111  116
015240  107  040  106
015243  117  122  040
015246  101  040  103
015251  117  116  124
015254  122  117  114
015257  055  103  056
015262  040  074  052
015265  052  052  052
015270  052  052  052
015273  052  052  052
015276  052  052  052
015301  045  116  045
015304  116  000  .EVEN
GLOBAL SUBROUTINE PRILPR

; PRINT THE CONTENTS OF THE LPR.
; THIS ROUTINE IS USED TO PRINT OUT EXTENDED INFORMATION ON THE
; CONTENTS OF THE LINE PARAMETER REGISTER (LPR).

; INPUTS:
; AS CONTAINS THE NUMBER OF THE LINE YOU WISH TO EXAMINE.
; CSRA - CONTAINS THE ADDRESS OF THE OUT'S CSR.
; IESTAT - CONTAINS THE CURRENT STATUS OF THE IX AND RX INTERRUPT
; ENABLE BITS IN THE OUT'S CSR.
; LPRA - CONTAINS THE ADDRESS OF THE OUT'S LPR REGISTER.

; OUTPUTS:
; AN EXTENDED INFORMATION MESSAGE IS PRINTED ON THE OPERATORS
; CONSOLE.

; CALLING SEQUENCE: JSR PC,PRILPR

; COMMENTS: THIS ROUTINE CHANGES THE INDIRECT ADDRESS FIELD OF THE DEVICE
; UNDER TEST'S CSR.

; SUBORDINATE ROUTINES CALLED: NONE.

PRILPR::SAVE

MOV CSRA,R1
JSR R5,PREGOS
CALL REGISTER SAVE SUBRT.

MOV CSRA,R2
GET THE CSR ADDRESS.

MOV LPRA,R2
GET THE LPR ADDRESS.

BIC @177760,R3
CLEAR ANY UNWANTED BITS.

BIC IESTAT,R3
SET STATE OF IX AND RX INTERRUPT ENABLE BITS.

MOV R3,(R1)
SELECT LINE.

MOV (R2),R4
GET CONTENTS OF THE LPR.

PRINT MESSAGE"CONTENTS OF THE LPR:????????"

PRINT @EF9019,@EM9026,R4;PRINT OUT MESSAGE ON OPERATORS CONSOLE.

MOV R4,(SP)

MOV @EM9026,(SP)
MOV @EF9019,(SP)
MOV @EM9026,(SP)
MOV @EF9019,(SP)
MOV R3,(SP)
MOV SP,R0
TRAP 81PNTX
ADD @10,SP

604: PASS JSR PC,8(SP).
;RESTORE GPRS.
;RETURN TO PREGOS SUBRT.

RTS PC
GLOBAL SUBROUTINE PUFF0

; SUBRT GLOBAL SUBROUTINE
; PURGE THE FIFO
; THIS ROUTINE TRIES TO REMOVE ALL THE CHARACTERS FROM THE FIFO.
; ANY BMP CODES THAT ARE FOUND ARE SAVED ON THE BMP CODE QUEUE.
; INPUTS: RBUFA: CONTAINS THE ADDRESS OF THE RECEIVER.
; OUTPUTS: CARRY BIT: INDICATES THE STATE OF THE FIFO. SET: = PURGED.
; BMPQ: THE CONTENTS OF THE BMP CODE QUEUE MAY BE UPDATED.
; CALLING SEQUENCE: JSR PC, PUFFO
; COMMENTS:
; SUBORDINATE ROUTINES CALLED: SAVBMP.

PUFFO: SAVE RS, REGS
; SAVE CONTENTS OF GPRS RO THRU RS.
JSR RS, PREGS
; CALL REGISTER SAVE SUBRT.
MOV #512, R1
; SET MAXIMUM TRY COUNT OF 512.
MOV RBUFA, R4
; GET ADDRESS OF THE RECEIVER BUFFER REGISTER.
2: MOV (R4), R2
; GET THE CONTENTS OF THE RECEIVER BUFFER REG.
BPL 64
; IF CONTENTS IS NOT NULL, DATA, VALID CLR.
; CHECK IF THE READ CHARACTER IS ACTUALLY A BMP CODE.
; IF IT IS, THEN SAVE IT ON THE BMP CODE QUEUE TO BE REPORTED LATER.
   MOV #70000, RO
; GENERATE A BIT MAP OF CHAR ERROR BITS
BIC R2, RO
; WHICH ARE NOT SET FOR CHAR.
BNE 41
; THROW CHAR AWAY IF NOT BMP OR SELFTEST CODE.
   CHECK IF THE READ DATA IS MODSTX STATUS, BMP OR SELFTEST?
   MOV #301, RO
; CHECK IF BMP.
BIC R2, RO
; TRY TO CLEAR BMP FLAGS IN THE READ DATA.
BNE 41
; IF IT IS MODSTX OR SELFTEST CODE THROW IT AWAY.
JSR PC, SAVBMP
; SAVE BMP CODE ON THE QUEUE.
4: DEC R1
; DECREMENT THE TRY COUNT.
BNE 28
; LOOP TO TRY AGAIN.
CLC
; CLEAR CARRY, TO INDICATE FIFO NOT PURGED.
BR 601
; EXIT WITH CARRY CLEAR.
6: SEC
; SET CARRY, TO INDICATE FIFO PURGED.
601: PASS JSR PC, (KSP)
; RESTORE GPRS.
; RETURN TO PREGS SUBRT.
RTS PC
.SBITL GLOBAL SUBROUTINE

- PURGE FIFO REPORT ANY ERRORS FOUND.
- THIS ROUTINE REMOVES ALL DATA FROM THE FIFO. ANY BMP CODES THAT ARE
- FOUND ARE SAVE IN THE QUEUE IN THE FIFO REPORT TEST.
- ANY UNEXPECTED DATA (IE ANY NON-STATUS INFORMATION) 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: ERRBLK, ERRATYPE, ERRMSG, ERRNBR ARE SET UP CORRECTLY.
ERRBUF CONTAINS THE ADDRESS OF THE RECEIVER.

OUTPUTS: CARRY BIT - ABORT TEST FLAG, CLR - ABORT TEST, SET - OK.
ERRBLK - VALUE WILL BE DESTROYED.
BMPCOP - THE BMP CODE QUEUE POINTER MAY BE UPDATED.
THE CONTENTS OF THE BMP CODE QUEUE MAY BE UPDATED.

CALLING SEQUENCE: JSR PC,PUIFR

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.

PUIFR: SAVE

;SAVE CONTENTS OF GPRs RO THRU RS.

JSR PC,PSR; CALL REGISTER SAVE SUBR.

MOV ERNBR,-(SP) ;SAVE THE CONTENTS OF THE ERROR NUMBER.
MOV #512,-RS ;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.

21: MOV ERBUF,R2 ;GET THE CONTENTS OF THE RECEIVER BUFFER REG.
       SPL #1 ;EXIT IF DATA VALID CLEAR, IE. FIFO PURGED.

CHECK IF READ DATA IS STATUS OR UNEXPECTED CHAR.

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,RO ;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 #1 ;BRANCH TO CHECK READ COUNT.

CHECK IF THE READ DATA IS MODEM, SELFTEST OR UNEXPECTED DATA.
3427 015530 032702 000001  41:   BIT @BIT0,R2   ; TEST THE MODEM STATUS INDICATION BIT.
3428 015534 001425   BEQ 61   ; DO NOT REPORT ANY ERROR IF MODEM STATUS.
3429 015536 012701 011124   MOV @EM9104,R1   ; PASS THE CORRECT ERROR MESSAGE TO REPORT.
3430 015542 010203   MOV R2,R3   ; EXTRACT THE LINE NUMBER FROM
3431 015544 000303   SWAP R3   ; THE READ DATA.
3432 015546 042703 177760   BIC @177760,R3   ;
3433 015552 006303   ASL R3   ; FORM LINE NUMBER TIMES 2 FOR ER9002 ROUTINE.
3434 015554 052704 100000   BIS @BIT15,R4   ; SET THE "NONE" EXPECTED MESSAGE FLAG.
3435 015560 005267 166266   INC ERRNR   ; SET ERROR NUMBER TO INITIAL ERRNR+1.
3436 015564 012767 012644 166264   MOV @ER9002,ERBLK   ; SELECT THE CORRECT ERROR REPORTING ROUTINE.
3437 015572 0104460   ; REPORT ERROR "UNEXPECTED DATA FOUND IN FIFO : ERROR : "
3438 015572 104460   TRAP CERROR
3439 1   EXIT WITH FAILURE IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
3440 1
3441 1
3442 015574 032767 000100 164406   BIT @BIT06,OPTION   ; EXIT WITH TEST FAILURE MESSAGE IF
3443 015602 001415   BEQ 71   ; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
3444 1   DURING THE SOFTWARE QUESTIONS.
3445 1
3446 015604 005367 166242   DEC ERRNR   ; RESTORE ERROR NUMBER TO INITIAL ERRNR.
3447 1
3448 015610 005305 61:   DEC R5   ; DECREMENT READ COUNTER.
3449 015612 001325   BNE 29   ; LOOP TO READ NEXT CHAR FROM FIFO IF COUNT > 0.
3450 1
3451 1   ; THE FIFO WILL NOT CLEAR, REPORT THE ERROR AND INDIcate THAT THE TEST IS TO
3452 1   ; BE ABORTED.
3453 1
3454 015614 062767 000002 166230   ADD @2,ERRNR   ; SET ERROR NUMBER TO INITIAL ERRNR+2.
3455 015622 012767 012020 166226   MOV @ER1603,ERBLK   ; SELECT THE CORRECT ERROR REPORTING ROUTINE.
3456 015630 012701 010767   MOV @EM9017,R1   ; PASS THE MESSAGE TO BE REPORTED.
3457 1   ; REPORT THE ERROR "FIFO WILL NOT PURGE. (DATA VALID STUCK SET)"
3458 1   "?????? TEST ABORTED".
3459 1
3460 015634 104460   71:   CLC   ; TRAP CERROR
3461 015636 000241   BR 101   ; INDICATE THE TEST IS TO BE ABORTED.
3462 1   ; EXIT THIS ROUTINE AND ABORT THE CURRENT TEST.
3463 015640 000401
3464 015642 000261   81:   SEC   ; SET THE CARRY, DO NOT ABORT THE TEST.
3465 015644 012667 166202   101:   MOV (SP)+,ERRNR   ; RESTORE INITIAL ERROR NUMBER.
3466 015650 004736   601:   PASS   ; RESTORE GPARS.
3467 015650 016500   JSR PC,(SP)   ; RETURN TO PREVIOUS SUBR.
3468 1   ; CARRY BIT, SET INDICATES FIFO PURGED, DO NOT
3469 015652 000207   RTS PC   ; ABORT THE TEST.
**.SBTRL GLOBAL SUBROUTINE READBX**

- READ CHARACTERS FROM THE FIFO AND CHECKS FOR BMPs AND XONs
- THIS SUBROUTINE IS USED IN THE FIFAVL.TST.
- IT READS THE SPECIFIED NUMBER OF CHARACTERS FROM THE FIFO AND CHECKS
  FOR BMP CODES AND XON CHARACTERS.

**Inputs:**
- R0 - CONTAINS THE NUMBER OF CHARs TO READ FROM THE FIFO.
- R1 - CONTAINS ADDRESS OF ERROR MESSAGE TO BE REPORTED
- CLEAR IF NO ERROR FOUND.
- CARRY USED TO INDICATE IF FIFO WAS FOUND EMPTY, CARRY CLEAR.

**Calling Sequence:**
- JSR PC, READ

**Comments:**
- SUBORDINATE ROUTINES CALLED: CHKXON.

```
READBX: SAVE
       JSR R5, PREGOS ; CALL REGISTER SAVE SUBRT.
       CLR R1 ; CLEAR GPRT THAT HOLDS THE ADDRESS OF ERRMSG.
       MOV RBUF, R3 ; GET THE ADDRESS OF THE RECEIVER BUFFER REG.
       BPL 80 ; BRANCH IF FIFO IS EMPTY.

; CHECK IF THE READ CHARACTER IS A BMP CODE.
; IF IT IS A BMP CODE SAVE IT ON THE QUEUE TO BE REPORTED LATER, AND
; ABORT THE TEST.

JSR PC, CHKXON ; CHECK IF CHARACTER IS A BMP CODE.
BCS 61 ; BRANCH IF A BMP CODE WAS FOUND.
CMP R2, 021 ; CHECK IF IT IS AN XON.
BNE 41 ; BRANCH IF NOT AN XON.
MOV REN5402, R1 ; PASS THE MESSAGE TO BE REPORTED.
BR 61 ; GO EXIT TEST.
DEC R0 ; DECREMENT THE READ COUNT.
BNE 20 ; SET CARRY TO INDICATE SUCCESS.
SEC 61 ; EXIT.
CLC 81 ; CLEAR CARRY BIT TO INDICATE FAILURE.
PASS R1 ; RESTORE GPRS.
MOV R1, RLSLOT(SP) ; PUT R1 IN STACK SLOT.
RTS PC
```
Global Subroutine

**SBITL**

---

*REPORT DATA ERRORS*

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3516</td>
<td><strong>SBITL</strong></td>
</tr>
<tr>
<td>3517</td>
<td><strong>GLOBAL SUBROUTINE</strong></td>
</tr>
<tr>
<td>3518</td>
<td><strong>REPDER</strong></td>
</tr>
<tr>
<td>3519</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3520</td>
<td>THIS SUBROUTINE IS USED TO REPORT INCORRECT CHARACTERS AND LINE</td>
</tr>
<tr>
<td>3521</td>
<td>NUMBERS IN A WORD OF DATA READ FROM THE RX FIFO. THIS ROUTINE</td>
</tr>
<tr>
<td>3522</td>
<td>CHECKS FOR THE NUMBER OF INDIVIDUAL DATA ERRORS ON A LINE EXCEEDING</td>
</tr>
<tr>
<td>3523</td>
<td>THE REPORT AMOUNT AND SETS THE APPROPRIATE ERROR SUMMARY FLAG. IT</td>
</tr>
<tr>
<td>3524</td>
<td>THEN STOPS REPORTING ANY FURTHER ERRORS ON THAT LINE. ANY BMP CODES</td>
</tr>
<tr>
<td>3525</td>
<td>FOUND ARE SAVED ON THE BMP CODE QUEUE TO BE REPORTED LATER AND ANOTHER</td>
</tr>
<tr>
<td>3526</td>
<td>CHARACTER IS READ FROM THE RX FIFO.</td>
</tr>
<tr>
<td>3527</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3528</td>
<td><strong>INPUTS:</strong></td>
</tr>
<tr>
<td>3529</td>
<td>R1 - CONTAINS THE NUMBER OF THE OUT.</td>
</tr>
<tr>
<td>3530</td>
<td>R3 - LOW BYTE CONTAINS THE EXPECTED CHAR.</td>
</tr>
<tr>
<td>3531</td>
<td>ERRCNTB - ADDRESS OF THE BASE OF THE ERROR SUMMARY TABLE.</td>
</tr>
<tr>
<td>3532</td>
<td>ERRBLK - ADDRESS OF ERROR REPORTING ROUTINE IN ERROR TABLE.</td>
</tr>
<tr>
<td>3533</td>
<td>ERRDIS - SET TO THE ERROR NUMBER OF THE FIRST ERROR IN THIS ROUTINE.</td>
</tr>
<tr>
<td>3534</td>
<td>EXDERR - ADDRESS OF &quot;EXIT ON ERROR&quot; FLAG.</td>
</tr>
<tr>
<td>3535</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3536</td>
<td><strong>OUTPUTS:</strong></td>
</tr>
<tr>
<td>3537</td>
<td>CARRY - CLEAR IF RX FIFO WAS EMPTY, SET OTHERWISE.</td>
</tr>
<tr>
<td>3538</td>
<td>ERRCNTB - CONTENTS UPDATED TO REFLECT THE CURRENT STATE.</td>
</tr>
<tr>
<td>3539</td>
<td>ERRBLK - CONTENTS MAY BE ALTERED.</td>
</tr>
<tr>
<td>3540</td>
<td>ERRISRF - ERROR SUMMARY FLAG UPDATED.</td>
</tr>
<tr>
<td>3541</td>
<td>EXDERR - 1 IF AT LEAST ONE ERROR OCCURRED.</td>
</tr>
<tr>
<td>3542</td>
<td>0 IF NO ERRORS OCCURRED.</td>
</tr>
<tr>
<td>3543</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3544</td>
<td><strong>CALLING SEQUENCE:</strong></td>
</tr>
<tr>
<td>3545</td>
<td><strong>JSR PC,REPDER</strong></td>
</tr>
<tr>
<td>3546</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3547</td>
<td><strong>COMMENTS:</strong></td>
</tr>
<tr>
<td>3548</td>
<td>THIS ROUTINE REPORTS ERRORS WITH NUMBERS INITIAL ERRDIS THRU INITIAL ERRDIS+1.</td>
</tr>
<tr>
<td>3549</td>
<td><strong>--------------------------------------------------------------</strong></td>
</tr>
<tr>
<td>3550</td>
<td><strong>SUBORDINATE ROUTINES CALLED:</strong></td>
</tr>
<tr>
<td>3551</td>
<td><strong>NONE</strong></td>
</tr>
</tbody>
</table>

**REPDER::**

```assembly
REPDER:: SAVE R5, RREGS ; CALL REGISTER SAVE SUBRT.
CLR 62 ; CLEAR THE "AN ERROR OCCURED" INDICATOR.
MDV @128, R5 ; SET THE MAX BMP READ COUNT.
CLC 21 ; CLEAR THE CARRY IN CASE THE FIFO IS EMPTY.
MDV @BUBA, R2 ; READ THE RX FIFO.
BPL 61 ; EXIT THIS SUBROUTINE WITH CARRY CLEAR IF FIFO EMPTY.

; CHECK IF THE CHARACTER IS A BMP CODE.
MOV @170301, R0 ; SET UP THE BIT MASK FOR A BMP CODE.
BIC R2, R0 ; TRY AND CLEAR THE BIT MASK.
DEC R5 ; AVOID SAVING THE CODE IF IT IS NOT A BMP CODE.
JSR PC, SAVBMP ; SAVE THE BMP CODE ON THE QUEUE.
BNE 21 ; READ ANOTHER CHARACTER FROM THE RX FIFO.

; VERIFY THAT THE READ CHARACTER IS CORRECT.
MOV R2, R5 ; SAVE THE READ DATA.
BIC @17400, R2 ; CLEAR THE CLUTTER FROM THE DATA CHAR.
cmp R2, R3 ; COMPARE THE READ CHAR WITH THE EXPECTED CHAR.
BEQ 61 ; AVOID THE ERROR REPORT IF THE DATA IS CORRECT.
MOV @1, 62 ; INDICATE AN ERROR HAS OCCURED.
```
GLOBAL SUBROUTINE  REPER  MOV  R1,R4    COPY THE LINE NUMBER.
3572  016022  010104  ASL  R4    CONVERT THE LINE NUMBER TO AN OFFSET VALUE.
3573  016024  006304  BIT  BITBL(R4),ERSHFR    TEST WHETHER ERROR SUMMARY REPORTING IS IN
3574  016026  036467  002356  164556  EFFECT FOR THIS LINE.
3575
3576  016034  001021  BNE  61  AVOID THE ERROR REPORT IF IT IS.
3577  016036  010104  MOV  R3,R4    PASS THE LINE NUMBER TO THE ERROR ROUTINE.
3578  016040  110301  MOVB  R3,R1    PASS THE EXPECTED DATA CHAR.
3579  016042  012703  010120  MOV  @EM6004,R3    PASS THE ERROR MESSAGE.
3580
3581  016046  012767  012276  166002  MOV  @ER6001,ERRBLK    SET UP THE ERROR REPORTING ROUTINE.
3582  016054  104460  "TXFIFO BAD, RECEIVED CHAR INCORRECT, ON LINE AX .
ERROR    REPORT THE ERROR.
3583  016056  010401  MOV  R4,R1    RESTORE THE LINE NUMBER.
3584  016060  032767  000100  164122  BIT  @BIT06,OPTION    HAS EXTENDED ERROR REPORTING BEEN REQUESTED.
3585  016066  001433  BNE  61  AVOID THE WRITING IF IT HAS.
3586  016070  012767  000001  164154  MOV  @1.EX0ERR    INDICATE THAT THE TEST IS TO BE EXITED.
3587  016076  004662  BR  601  EXIT THIS SUBROUTINE.
3588
3589
3590
3591
3592  016100  010502  61:  MOV  R5,R2    RESTORE THE DATA READ FROM THE RXFIFO.
3593  016102  003002  SWAB  R2    PLACE THE LINE NUMBER IN THE LOW BYTE.
3594  016104  042702  177760  BIC  @177760,R2    CLEAR THE CLUTTER FROM THE LINE NUMBER.
3595  016110  020201  CMP  R2,R1    COMPARE WITH THE EXPECTED LINE NUMBER.
3596  016112  001433  BEQ  61  AVOID THE ERROR REPORT IF THE NUMBER IS CORRECT.
3597  016114  012767  000001  000130  MOV  @1,621    INDICATE THAT AT LEAST ONE ERROR HAS BEEN FOUND.
3598  016116  010401  MOV  R1,R4    COPY THE LINE NUMBER.
3599  016118  006304  ASL  R4    CONVERT THE LINE NUMBER TO AN OFFSET VALUE.
3600  016126  036467  002356  164456  BIT  BITBL(R4),ERSHFR    TEST WHETHER ERROR SUMMARY REPORTING IS IN
3601
3602  016134  001022  BNE  61  AVOID THE ERROR REPORT IF IT IS.
3603  016136  005267  165710  INC  ERRBR    SET THE ERROR NUMBER TO INITIAL+1.
3604  016142  012703  010165  MOV  @EM6005,R3    PASS THE MESSAGE.
3605
3606  016146  012767  012170  165702  MOV  @ER5901,ERRBLK    SET UP THE ERROR REPORTING ROUTINE.
3607  016154  104460  ERROR    REPORT THE ERROR.
3608  016156  005367  165670  DEC  ERRBR    SET THE ERROR NUMBER TO INITIAL.
3609  016160  032767  000100  164020  BIT  @BIT06,OPTION    HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
3610  016166  001433  BNE  61  AVOID SETTING THE "EXIT ON ERROR" FLAG IF IT HAS.
3611  016170  012767  000001  164052  MOV  @1.EX0ERR    SET THE FLAG.
3612  016200  00421  BR  601  EXIT THIS SUBROUTINE.
3613
3614
3615
3616
3617
3618  016202  005767  164004  81:  TST  NDRPT    TEST IF SUMMARY REPORTING HAS BEEN REQUESTED.
3619  016204  001437  BEQ  601  EXIT THIS SUBROUTINE IF IT HASN'T.
3620  016210  005767  000036  TST  621  TEST IF AN ERROR HAS BEEN FOUND.
3621  016214  001437  BEQ  603  EXIT THIS SUBROUTINE IF NO ERRORS.
3622  016216  010104  MOV  R1,R4    COPY THE LINE NUMBER.
3623  016220  006304  ASL  R4    CONVERT THE LINE NUMBER TO AN OFFSET VALUE.
3624  016224  002614  ENCNTB(R4)    INCORRECT THE APPROPRIATE ERROR COUNT
3625  016226  026764  163760  002614  CMP  NDRPT,ERCNTB(R4)    COMPARE THE NUMBER OF ERRORS REPORTED WITH.
3626
3627 016234 101003
3628
3629
3630 016236 056467 002336 164346
3631
3632 016244 000261
3633 016246
3634 016246 004736
3635 016250 000207
3636 016252 000000

BHI 601: AVOID SETTING THE ERROR SUMMARY FLAG IF THE
NUMBER OF ERRORS REPORTED IS LESS THAN THE
NUMBER REQUESTED.

BIS BITBL(R4), ERSMRF; OTHERWISE SET THE APPROPRIATE ERROR FLAG.

601: SEC; SET THE CARRY FLAG AND INDICATE SUCCESS.
611: PASS

JSR PC,(SP)+; RETURN TO PREGOS SUBRT.

621: .WORD 0; STORAGE FOR THE AT LEAST ONE ERROR" INDICATOR.
GLOBAL SUBROUTINE

.SBTTL GLOBAL SUBROUTINE

REPSMR

******************************************************************************

** THIS SUBROUTINE REPORTS AN ERROR SUMMARY FOR THOSE LINES WHICH HAVE
** EXCEEDED THE NUMBER OF INDIVIDUAL ERRORS TO REPORT FOR A SINGLE LINE
** IN A SINGLE TEST. THIS PARAMETER CAN BE SPECIFIED BY THE OPERATOR IF
** HE/SHE ANSWERS THE SOFTWARE PARAMETER QUESTIONS.

** INPUTS:
** ERNCNTB - LABEL AT BASE OF LINE ERROR COUNTERS TABLE.
** ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE.
** ERRMNR - ERROR NUMBER OF ERRORS IN THIS ROUTINE.
** ERRSRF - "REPORT ERROR SUMMARY FOR LINE" FLAGS.

** OUTPUTS:
** ERRBLK - ADDRESS OF ERROR REPORTING ROUTINE (DESTROYED).
** SUMMARY MESSAGES MAY BE PRINTED AT THE OPERATOR CONSOLE.

** CALLING SEQUENCE: JSR PC,REPSMR

** COMMENTS:
** IF NO LINES HAVE EXCEEDED THE MAXIMUM NUMBER OF INDIVIDUAL
** ERRORS TO REPORT, NO MESSAGES ARE PRINTED BY THIS ROUTINE.
** ERROR SUMMARIES IN THIS ROUTINE ARE REPORTED AS ERRORS.
** THE CONTENTS OF ERRBLK ARE DESTROYED.

** SUBORDINATE ROUTINES CALLED:

******************************************************************************

REPSMR:: SAVE JSR 16254 RS,PC,REPSMR 016254 0004567 165600
16254 0005767 164326
16260 001404
16264 016254 016266 012767 013022 165562
16268 016266 012767 013022 165562
16270 016266 012767 013022 165562
16273 016274 016274 104460
16274 016274 104460
16275 016276
16276 016276 004736
16276 016300 000207

MOV #ER9000,ERRBLK 16276 016276 004736
16276 016300 000207

TRAP CIERROR

601: PASS JSR 016274 RS,PC,REPSMR
16274 016274 104460
16275 016276
16276 016276 004736
16276 016300 000207

RETURN TO PREGOS SUBRT.
GLOBAL SUBROUTINE

.retl

GLOBAL SUBROUTINE

RESSET

;*******************************************************************************
; RESET DEVICE UNDER TEST.
; This subroutine is used to reset the device to a known state.
; If reset does not successfully complete, if time-out occurs, then
; an abort test error message is reported.
; INP: CSRA CONTAINS THE ADDRESS OF THE CSR.
; TXBFCA - CONTAINS ADDRESS OF DUT DMA BUFFER COUNT REGISTER.
; ERRBL - ERRTP, ERRBR, AND ERRMSG SET UP CORRECTLY.
; OUT: DUT PERFORMS ITS RESET FUNCTION INTO A KNOWN STATE.
; CARRY - CLEAR INDICATES THE TEST IS TO BE ABORTED.
; ERRBLK - VALUE MAY BE DESTROYED.
; IESTAT: TX AND RX INTERRUPT FLAGS ARE CLEARED.
; RX AND RX INTERRUPT ENABLE BITS IN THE DUT CSR ARE CLEARED.
; CALLING SEQUENCE:  JSR PC,RESSET
; COMMENTS: THIS SUBROUTINE CAN REPORT ERRORS WITH NUMBERS INITIAL ERRBR
; THIS ROUTINE DOES NOT DESTROY THE VALUE OF ERRBR.
; SUBORDINATE ROUTINES CALLED: DELAY,MSLGET.

RESSET:: SAVE
    ;SAVE CONTENTS OF GPRS RO THRU R5.
    ; JSR R5PREGO5 ;CALL REGISTER SAVE SUBRT.
    MOV @BIT05,R2 ;SET BIT MASK OF MASTER RESET BIT.

    ; TEST THE STATE OF THE MASTER RESET BIT IN THE CSR.
    ; IF MR IS SET THEN WAIT FOR SELF-TEST TO COMPLETE.
    ; IF TIME-OUT OCCURS, REPORT THE ERROR AND PASS-OUT ABORT TEST INDICATOR.

    ; MOV CSRA,R4 ;GET THE ADDRESS OF THE DUT'S CSR.
    ; BIT R2,(R4) ;CHECK STATE OF MASTER RESET BIT.
    BEQ 24 ;DON'T DELAY IF MR IS Already CLEAR.
    CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
    JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
    BCC 41 ;GO REPORT ERROR IF TIMEOUT OCCURRED.

    ; SET MASTER RESET BIT IN CSR, CLEAR TX AND RX ENABLE BITS, ETC.
    ; SKIP THE SELFTEST.
    ; TIME-OUT OF 5 SECS, JUST IN CASE THE SELF-TEST EXECUTES.

    ; MOV R2,BCSRA ;SET MASTER RESET BIT, DISABLE TX AND RX INTS.
    JSR PC,SKPTS ;TRY TO SKIP THE SELFTEST.

    ; SET SELF-TEST TIME-OUT OF 5 SECONDS, AND WAIT FOR MR TO CLEAR.
    ; IF TIME-OUT OCCURS, THEN REPORT THE FATAL ERROR AND PASS-OUT THE ABORT
    ; TEST INDICATOR.

    ; CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
    ; MOV #5000,R1 ;PASS TIME-OUT VALUE OF 5 SECONDS.
    JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
    BCS 61 ;SKIP ERROR REPORT IF MR CLEARED IN TIME.
1  SET UP ERROR MESSAGE TO REPORT "FATAL ERROR FOUND DURING RESET, TEST ABORTED".
   ; INDICATE TEST IS TO BE ABORTED BY CLEARING THE CARRY BIT.

2  MOV @EM1601, R1 ; PASS ERROR MESSAGE TO REPORT.
3  MOV @ER1603, ERRBLK ; PASS ADDRESS OF ERROR HANDLING ROUTINE.
4  REPORT ERROR "TIME-OUT OCCURRED WAITING FOR MASTER RESET TO CLEAR"
5  ; "TEST ABORTED"
6  ERROR
7  ; TRAP CIERROR
8  CLEAR TX AND RX INTERRUPT STATUS FLAGS IN IESTAT.
9  EXIT WITH CONTINUE TEST INDICATOR SET (IE, CARR SET).

CLR IESTAT ; CLEAR TX AND RX INTERRUPT STATUS FLAGS.
SEC ; INDICATE SUCCESS, CONTINUE TEST.
PASS JSR PC, (SP)
; RETURN TO PREGOS SUBR.
CARRY BIT: IF CLEAR, INDICATES ABORT TEST.
SRTL GLOBAL SUBROUTINE RXIEO

;**  ******************************************************
;  - RECEIVER INTERRUPT DISABLE -
;**  THIS ROUTINE IS USED TO DISABLE RECEIVER INTERRUPTS IN THE DMU1.
;**
;**  Inputs:   NONE.
;**  Outputs:  THE RX.INT.ENBL BIT IS CLEARED IN THE OUT CSR.
;**  IESTAT CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT
;**  ENABLE BITS.
;**
;**  Calling Sequence:  JSR PC, RXIEO
;**
;**  Comments:   THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
;**               THE OUT CSR ARE DESTROYED.
;**
;**  Subordinate Routines Called: NONE.
;**

RXIEO:   MOV RO, -(SP)  ; SAVE CONTENTS OF RO ON THE STACK.
         GETPRI -(SP)   ; SAVE PROCESSOR PRIORITY ON STACK.
         TRAP C4SPRI
         MOV RO, (SP)  ; SAVE PROCESSOR PRIORITY ON STACK.
         SETPRI #PRI07  ; IGNORE ANY INTERRUPT THAT MAY BE GENERATED.
         MOV #PRI07,RO
         TRAP C4SPRI
         BIC 013777, IESTAT ; CLEAR RX.INT.ENBL BIT IN IESTAT.
         MOV IESTAT, &CSRA ; DISABLE RX INTERRUPTS.
         SETPRI (SP)* ; ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
         MOV (SP)++, RO
         TRAP C4SPRI
         RTS PC ; RESTORE RO.
GLOBAL SUBROUTINE RXIE1:

.SBTTL GLOBAL SUBROUTINE - RXIE1 -

* THIS ROUTINE IS USED TO ENABLE RECEIVER INTERRUPTS IN THE DHU11.

* INPUTS:  NONE.

* OUTPUTS:  THE RX.INT.ENBL BIT IS SET IN THE DUT CSR.

* IESTAT - CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT

* ENABLE BITS.

* CALLING SEQUENCE:  JSR PC,RXIE1

* COMMENTS:  THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN

* THE DUT CSR ARE DESTROYED.

* SUBORDINATE ROUTINES CALLED:  NONE.

RXIE1:

BIS #0BIT0,IESTAT ; SET RX.INT.ENBL BIT IN IESTAT.
BIC #0137677,IESTAT ; CLEAR ALL OTHER BITS, EXCEPT TX AND RX I.E.
MOV IESTAT,&CSRA ; ENABLE RX INTERRUPTS.
RTS PC
GLOBAL SUBROUTINE

*-------------------------------------------------------------*
* THIS ROUTINE SAVES THE PARAMETER PASSED IN, ONTO THE BMP CODE QUEUE *
* TOGETHER WITH THE NUMBER OF THE CURRENTLY EXECUTING TEST.     *
*-------------------------------------------------------------*
* INPUTS:                                                           *
* R2 - CONTAINS THE BMP CODE THAT IS TO BE PLACED ON THE QUEUE.    *
* BMPDOP - CONTAINS ADDRESS OF NEXT LOCATION IN THE BMP QUEUE.     *
* BMPDQ - LABEL AT BASE OF THE BMP CODE QUEUE.                    *
* TSNUM - LABEL OF NEXT LOCATION AFTER THE END OF THE BMP QUEUE.  *
* TSNUM CONTAINS THE NUMBER OF THE CURRENT TEST.                  *
*-------------------------------------------------------------*
* OUTPUTS:                                                          *
* BMPDOP - INCREMENTED BY 4.                                        *
* THE CONTENTS OF THE BMP CODE QUEUE ARE UPDATED.                  *
*-------------------------------------------------------------*
* CALLING SEQUENCE:                                                 *
* JSR PC, SAVBMP                                                   *
*-------------------------------------------------------------*
* COMMENTS:                                                         *
* IF THE OVERFLOW OCCURS THEN THE LAST LOCATION WILL BE           *
* OVERWRITTEN BY ANY SUBSEQUENT ATTEMPTS TO UPDATE THE QUEUE.     *
*-------------------------------------------------------------*
* SUBORDINATE ROUTINES CALLED: NONE.                               *
*-------------------------------------------------------------*

SAVBMP:: SAVE
JSR R5, PREGOS  ; CALL REGISTER SAVE SUBRT.
MOV BMPDQ, R4    ; GET THE POINTER TO THE NEXT LOCATION IN QUEUE.
MOVB TSNUM, (R4) ; SAVE THE CURRENT TEST NUMBER ON THE QUEUE.
INC R4           ; INCREMENT THE POINTER TO GIVE AN EVEN ADDRESS.
BIC $177400, R2  ; CLEAR THE UNWANTED BITS FROM THE BMP CODE.
MOV R2, (R4)     ; SAVE THE BMP CODE ON THE QUEUE.
CMP R4, #BMPDOP  ; CHECK IF OVERFLOW WILL OCCUR THE NEXT TIME.
BLO 21           ; GO SAVE THE POINTER IF WE WILL NOT OVERFLOW.
SUB R4, R4       ; RESET THE POINTER TO THE LAST LOCATION IN QUEUE.
MOV R4, BMPDOP   ; SAVE THE POINTER.

10: MOV R4, BMPDOP     ; SAVE CONTENTS OF GPRS 10 THRU R5.
    JSR R5, PREGOS  ; CALL REGISTER SAVE SUBRT.
    RTS PC           ; RETURN TO PREGOS SUBRT.

604: PASS

RTS PC
.SBTL GLOBAL SUBROUTINE - SETPAR

;******************************************************************************
;*  - SET TX AND CONTROL PARAMETERS -
;******************************************************************************
;*  THIS SUBROUTINE IS USED IN THE FINAVL.TST.
;*  IT INITIALISES THE SELECTED LINE TO THE FOLLOWING STATE:
;*  INTERNAL LOOPBACK, IAUTO ENABLED, LPR:38.4K, 8 BITS/CHAR, 2 STOP,
;*  ODD PARITY.
;*  INPUTS:       R1 - CONTAINS NUMBER OF THE LINE TO BE INITIALIZED.
;*  OUTPUTS:      LNCTRL AND LPR REGISTERS FOR THE SELECTED LINE ARE DESTROYED.
;*  CALLING SEQUENCE:  JSR    PC, SETPAR
;*  COMMENTS:      
;*  SUBORDINATE ROUTINES CALLED: DELAY, WTWLNC, WTWLPR.
;******************************************************************************

SETPAR:: SAVE

JSR    R5, PREGOS   ;CALL REGISTER SAVE SUBRT.

JSR    PC, LINBIT   ;GET A BIT MAP FOR THIS LINE.

MOV    R0, R5       ;COPY THE LINE BIT MAP.

MOV    #206, R0     ;PASS INTERNAL LOPBCK, ENABLE RX AND IAUTO.

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 MILLISEC.

JSR    PC, DELAY    ;WAIT FOR LNCTRL AND LPR REGS TO BE UPDATED.

601:  PASS

JSR    PC, @ (SP)   ;RESTORE GPRS.

RTS    PC           ;RETURN TO PREGOS SUBRT.

3846  016546
3847  016546       004567   165306
3848  016552       004767   176072
3849  016556       010005
3850  016560       012700   000206
3851  016564       004767   000762
3852  016570       012700   177670
3853  016574       004767   001002
3854  016600       012704   000012
3855  016604       004767   175452
3856  016610
3857  016610       004736
3858  016612       000207
**GLOBAL SUBROUTINE**

**SKPSTS**

```
*** SUBROUTINE IS USED TO SKIP THE SELFTEST AFTER A DUT RESET HAS BEEN INITIATED. IT MUST BE ENTERED IMMEDIATELY AFTER SETTING THE DUT MASTER RESET ROUTINE OR AFTER THE EXECUTION OF A BUS RESET (BECAUSE OF TIMING CONSIDERATIONS). ***

INPUTS: CSRA - CONTAINS ADDRESS OF THE OUT CSR.
        TXBFCA - CONTAINS ADDRESS OF DUT DMA BUFFER COUNT REGISTER.

OUTPUTS: SKIP SELFTEST CODES ARE WRITTEN TO THE DUT REGISTERS.

CALLING SEQUENCE: JSR PC, SKPSTS

COMMENTS:

SUBORDINATE ROUTINES CALLED: DELAY.

SKPSTS:: SAVE
    MOV @10, R4
    JSR PC, SAVE SUBRAT.
    JSR PC, DELAY FOR 10 MILLI-SECONDS.

WRITE SKIP SELF-TEST CODE (52525) TO ALL THE INDEXED DUT REGISTERS.

MOV #52525, R3
DEC R1
MOV CSRA, R4
GET THE ADDRESS OF THE CSR OF THE DUT.
MOV R1, R4
SELECT A BANK OF DUT REGISTERS.
MOV R3, R4
WRITE THE CODE TO A DUT REGISTER.
CMP R4, TXBFCA
COMPARE POINTER WITH LAST REGISTER ADDRESS.
BLO 64
LOOP IF NOT ALL REGS DONE IN THIS BANK.
BIT #17, R1
TEST FOR IND.ADR.REG FIELD DECREMENTED TO 0.
BNE 48
LOOP UNTIL ALL REGISTERS CONTAIN THE CODE.

PASS
    JSR PC, (SP), RETURN TO PREGOS SUBRAT.
    RTS PC
```
**GLOBAL SUBROUTINE**

```
..SBTLT  GLOBAL SUBROUTINE         TSABRT

* THIS SUBROUTINE IS USED WHEN A NON-TEST RELATED ERROR HAS BEEN FOUND
* DURING THE EXECUTION OF THE CURRENT TEST.
* IT IS USED TO INFORM THE OPERATOR THAT THE CURRENT TEST HAS BEEN
* ABORTED.

* INPUTS:   ERRMSG - CONTAINS THE NAME OF THE CURRENT TEST.
            ERRNBR - CONTAINS THE CORRECT ERROR NUMBER.
            THE REMAINDER OF THE ERRCTL IS CORRECTLY INITALISED.

* OUTPUTS:  MESSAGES ARE REPORTED TO THE OPERATOR.

* CALLING SEQUENCE:  JSR  PC,TSABRT

* COMMENTS:

* SUBORDINATE ROUTINES CALLED: ER1603.
```

```
TSABRT::  SAVE
          JSR  R5,PREG5  CALL REGISTER SAVE SUBRT.
          MOV  R21,R1  PASS ADDRESS OF FIRST MESSAGE TO BE REPORTED.
          MOV  #ER1603,ERRBLK  SET-UP THE ERROR REPORTING ROUTINE.
          TRAP  C|ERROR

BR  601
```

```
601:   .ASCIZ / NON-RELATED TEST ERROR FOUND DURING TEST EXECUTION/
```

```
.EVEN
```

```
601:   PASS
          JSR  PC,KSP  RESTORE GPRS.
          RTS  PC  RETURN TO PREG5 SUBRT.
```
GLOBAL SUBROUTINE TXDATP

**SBTTL** GLOBAL SUBROUTINE TXDATP

- TRANSMIT DATA PATTERN.
- THIS SUBROUTINE IS USED IN THE FHAVL.TST.
- IT TRANSMITS A SPECIFIED NUMBER OF DATA BYTES ON THE SPECIFIED LINE.

**INPUTS:**
- RO - CONTAINS THE NUMBER OF DATA BYTES TO TX.
- R1 - CONTAINS LINE NUMB ON WHICH TRANSMISSION IS TO TAKE PLACE.
- BUF BAS TO BUF M ID CONTAINS A 256 BYTE DATA PATTERN.

**OUTPUTS:**
- DATA IS SENT OUT ON THE SPECIFIED LINE.
- CARRY SET = TX SUCCESSFUL.

**CALLING SEQUENCE:** TXDATP

**COMMENTS:**

**SUBORDINATE ROUTINES CALLED:** DODMA.

---

```
TXDATP:: SAVE JSR RS,PREGOS ; CALL REGISTER SAVE SUBRT.
    MOV RO,R3 ; PASS THE NUMBER OF CHARs TO TX.
    MOV #BUF BAS,R2 ; PASS THE START OF THE DATA PATTERN TO TX.
    JSR PC,DODMA ; TRANSMIT THE DATA PATTERN.

601: PASS JSR PC,DODMA ; RESTORE GPRS.
    RTS PC ; RETURN TO PREGOS SUBRT.
```

---

3951 3952 3953 3954 3955 3956 3957 3958 3959 3960 3961 3962 3963 3964 3965 3966 3967 3968 3969 3970 3971 017004 017004 004567 165050 3972 017010 010003 3973 017012 012702 002650 3974 017016 004767 175356 3975 017022 004736 3976 017024 000207
```
.SBTTL GLOBAL SUBROUTINE - TXDSBL

** THIS SUBROUTINE IS USED TO DISABLE TRANSMISSION ON SELECTED LINES BY CLEARING THE ASSOCIATED TX.ENABLE BIT ON THE DUT. **

** INPUTS: **
- R5 - BIT'S SET CORRESPOND TO LINES ON WHICH TO CLEAR TX.ENABLE.
- CSR6 - CONTAINS THE ADDRESS OF THE DUT CSR.
- IESTAT - CONTAINS THE STATE OF TXIE AND RXIE BITS IN THE CSR.
- NUMLNS - EQUATED TO BE THE MAXIMUM NUMBER OF LINES AVAILABLE.
- TXAD2A - CONTAINS THE ADDRESS OF THE TBUFFAD2 REGISTER.

** OUTPUTS: **
- R5 - BIT'S SET INDICATE THE INITIAL STATES OF ALL TX.ENABLE BITS.
- TBUFFAD2 - THE STATE OF THE TX.ENABLE BIT MAY BE ALTERED.
- THE CONTENTS OF THE INQ,ADD,REG FIELD IN THE CSR ARE DESTROYED.

** CALLING SEQUENCE: **
- JSR PC, TXDSBL

** COMMENTS: **
- SUBORDINATE ROUTINES CALLED: NONE.

TXDSBL:: SAVE RXS,PRGREGS ; CALL REGISTER SAVE SUBRT.

MOV R5,R0 ; COPY BIT MAP OF LINES TO DISABLE TRANSMISSION.
MOV #0100,R1 ; INITIALIZE THE SELECTED LINE BIT MASK.
MOV TXAD2A,R2 ; GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
INC R2 ; ADD 1 TO ADDRESS OF TSBUF2 REGISTER.
MOV @NUMLNS,R3 ; GET MAXIMUM LINE NUMBER PLUS ONE.
CLR R5 ; CLEAR POSSIBLE TX DISABLED ON ALL LINES.

SELECT EVERY LINE IN TURN, AND LOG THE STATE OF EACH TX.ENABLE BIT.

MOV R4,CSR6 ; WRITE TO DUT CSR TO SELECT LINE REGISTERS.
TSB (R2) ; CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
BIS R1,R5 ; LOG TX.ENABLE BIT SET FOR SELECTED LINE.

CLEAR TX.ENABLE ON LINES THAT HAVE A CORRESPONDING BIT SET IN THE TX DISABLE LINE BIT MAP.

BIT R1,R0 ; CHECK STATE OF DISABLE LINE BIT MAP.
BNE 61 ; BRANCH IF THIS LINE TO REMAIN UNALTERED.
BICS #0100,R2 ; CLEAR TX.ENABLE BIT ON SELECTED LINE.
INC A4 ; PREPARE TO SELECT REGISTERS FOR NEXT LINE.
ASL R1 ; SHIFT BIT MAP FOR NEXT LINE.
DEC R3 ; DECREMENT LINE NUMBER.
BNE 21 ; LOOP TO CHECK NEXT LINE.

RESTORE GPR'S EXCEPT RXS : RETURN TO PREVIOUS STATES OF ALL TX.ENABLE RITS.

PASS R5
MOV R5,RSSLOT(SP) ; PUT R5 IN STACK SLOT.
JSR PC,(SP) ; RETURN TO PREGREGS SUBRT.

RTS PC
.SBTL GLOBAL SUBROUTINE TXENBL

;**---------------------------------------------------------------**
;** THIS SUBROUTINE IS USED TO ENABLE TRANSMISSION ON SELECTED LINES BY **
;** SETTING THE ASSOCIATED TX.ENABLE BIT ON THE OUT. **
;**
;** INPUTS: **
;** R5 - BITS SET CORRESPOND TO LINES ON WHICH TO SET TX.ENABLE. **
;** CSRA - CONTAINS THE ADDRESS OF THE OUT CSR. **
;** ISTAT - CONTAINS THE STATE OF TXIE AND AXIE BITS IN THE CSR. **
;** NUMLNS = EQUATED TO BE THE MAXIMUM NUMBER OF LINES AVAILABLE. **
;** TXAD2A - CONTAINS THE ADDRESS OF THE TBUFFAD2 REGISTER. **
;**
;** OUTPUTS: **
;** R5 - BITS SET INDICATE PREVIOUSLY DISABLED LINES. **
;** TBUFFAD2: THE STATE OF THE TX.ENABLE BIT MAY BE ALTERED. **
;** THE CONTENTS OF THE INI.ADD.REG FIELD IN THE CSR ARE DESTROYED. **
;**
;** CALLING SEQUENCE: **
;** JSR PC,TXENBL **
;**
;** COMMENTS: **
;**
;** SUBORDINATE ROUTINES CALLED: NONE. **
;**
;**---------------------------------------------------------------**

TXENBL: SAVE R5,PREG05 TO TXTENBL
          JSR PC,PREG05 ;CALL REGISTER SAVE SUBRT.
          MOV R5,R0 ;COPY BIT MAP OF LINES TO ENABLE.
          MOV @BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
          MOV TXAD2A,R2 ;GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
          MOV ISTAT,R4 ;GET THE STATES OF THE INT ENABLE BITS.
          MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER.
          MOV R5,R5 ;CLEAR TX.ENABLE BIT LOG OF DISABLED LINES.
     21: MOV R4,CSRA ;WRITE TO OUT CSR TO SELECT LINE REGISTERS.
          TSB (R2) ;CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
          BIS R1,R5 ;LOG TX ENABLE BIT CLEAR FOR SELECTED LINE.
     45: BIT R1,R0 ;CHECK STATE OF TX.ENABLE LINE BIT MAP.
          BEQ 61 ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
          JNC R4 ;PREPARE NEXT INSTRUCTION IF TX.ENABLE SET.
          ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
          INC R3 ;DECREMENT LINE NUMBER.
          BNE 21 ;LOOP TO CHECK NEXT LINE.
     60: PASS R5 ;RESTORE GPX,EXCEPT R5,RSRLOT(SP) ;PUT R5 IN STACK SLOT.
          JSR PC,PREG05 ;RETURN TO PREG05 SUBRT.
          MOV R5,SP(SP) ;RS- LINE BIT MAP CORRESPONDING TO THE
          JSR PC,PREG05 ;PREVIOUS LINES THAT WERE DISABLED.
GLOBAL SUBROUTINE

**TXIEO**

- **TRANSMITTER INTERRUPT DISABLE**
- **THIS ROUTINE IS USED TO DISABLE TRANSMITTER INTERRUPTS IN THE DMU11.**

**INPUTS:** NONE.

**OUTPUTS:**
- THE TX.INT.ENBL BIT IS CLEARED IN THE OUT CSR.
- IESTST CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT ENABLE BITS.

**CALLING SEQUENCE:** JSR PC,TXIEO

**COMMENTS:**
- THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN THE OUT CSR ARE DESTROYED.
- **SUBORDINATE ROUTINES CALLED:** NONE.

```assembly
TXIEO: MVEL RO,(SP)  ;SAVE CONTENTS OF RO ON THE STACK.
      TRAP CISPRI
      MOVL RO,(SP)  ;SAVE CURRENT PROCESSOR PRIORITY ON THE STACK.
      TRAP CISPRI
      SETPRI #PRI07  ;IGNORE ANY INTERRUPTS THAT MAY BE GENERATED.
      MOV  #PRI07,RO
      TRAP CISPRI
      BIC  #177677,IESTAT  ;CLEAR TX.INT.ENBL BIT IN IESTAT.
      MOV  IESTAT,CSRRA  ;DISABLE TX INTERRUPTS.
      SETPRI (SP)  ;ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
      MOVL (SP)++,RO
      TRAP CISPRI
      MOV  (SP)++,RO  ;RESTORE RC.
      RTS  PC
```
SBTTL GLOBAL SUBROUTINE TXIE1

*  ***************************************************************
*  * TRANSMITTER INTERRUPT ENABLE
*  * THIS ROUTINE IS USED TO ENABLE TRANSMITTER INTERRUPTS IN THE DHU11.
*  *
*  * INPUTS:   NONE.
*  *
*  * OUTPUTS:  THE TX.INT.ENABLE BIT IS SET IN THE DUT CSR.
*            IESTAT -CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT
*            ENABLE BITS.
*  *
*  * CALLING SEQUENCE:  JSR PC.TXIE1
*  *
*  * COMMENTS:  THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
*               THE DUT CSR ARE DESTROYED.
*  *
*  * SUBORDINATE ROUTINES CALLED:  NONE.
*  *
*  ***************************************************************

        017256  052767  040000  162772 TXIE1::  BIS  #BIT14.IESTAT ;SET TX.INT.ENABLE BIT IN IESTAT.
        017264  042767  137677  162764 BIC  #137677.IESTAT ;CLEAR ALL BITS EXCEPT TX RX I.E BITS.
        017272  016777  162760  162726 MOV  IESTAT.@CSRA ;ENABLE TX INTERRUPTS.
        017300  000207  RTS  PC
GLOBAL SUBROUTINE

UNSIGNED DIVIDE ROUTINE:

1. THIS SUBROUTINE IS USED TO DIVIDE A 32 BIT UNSIGNED DIVIDEND BY A 16 BIT UNSIGNED DIVISOR GIVING A 16 BIT QUOTIENT. ALL NUMBERS ARE CONSIDERED TO BE UNSIGNED. A SUCCESS FLAG IS NOT SET ON RETURN IF THE QUOTIENT WAS TOO BIG TO BE CONTAINED IN 16 BITS.

INPUTS:
- R1 - THE DIVISOR, UNSIGNED, 16 BIT.
- R2 - MOST SIGNIFICANT WORD OF THE DIVIDEND, UNSIGNED, 16 BIT.
- R3 - LEAST SIGNIFICANT WORD OF THE DIVIDEND, UNSIGNED, 16 BIT.

OUTPUTS:
- R1 - QUOTIENT, UNSIGNED, 16 BIT (177777 IF OVERFLOW).
- R4 - CARRY - SUCCESS FLAG, SET IF COMPLETE QUOTIENT FITS IN 16 BIT.

CALLING SEQUENCE:
JSR PC, UNSDIV

COMMENTS:
- IF THE DIVISOR IS 0 THE QUOTIENT IS RETURNED AS ALL ONES (177777777) AND THE CARRY IS CLEAR REGARDLESS OF THE DIVIDEND.

SUBORDINATE ROUTINES CALLED: NONE.

UNSDIV:: SAVE
JSR R5,PREGOS CALL REGISTER SAVE SUBRT.

CHECK FOR QUOTIENT GREATER THAN 16 BITS CONDITION.

MOV R2,R4 GET MSW OF DIVIDEND FOR SUBTRACT.
SUB R1,R4 SUBTRACT DIVISOR FROM MSW OF DIVIDEND.
BCS 2 IF IT DIDN'T GO, WE HAVE QUOTIENT < 16 BITS.
MOV 00000100 SET QUOTIENT TO ALL ONES (177777).
BR 601 EXIT WITH CARRY CLEAR.

SET UP COUNTERS AND VARIOUS WORKING GPRS.

2F: CLR R4 CLEAR THE LSW OF THE DIVISOR.
CLC CLEAR CARRY FOR THE SHIFT OF THE DIVISOR.
RDR R1 DIVISOR BY
RDR R4 2(UNSIGNED)
MOV 016, RD SET UP INITIAL SHIFT COUNT TO 16.

THE SUBTRACT AND SHIFT LOOP.

4F: MOV R2, SP SAVE MSWORD OF DIVIDEND.
MOV R5, SP SAVE LSWORD OF DIVIDEND.
SUB R4,R3 LSWORD DIVIDEND - LSWORD OF DIVISOR.
SBC R2 MSWORD DIVIDEND - BORROW.
BCS 6 IF BORROW FROM BORROW SUBTRACT, IT DIDN'T GO.
SUB R1,R2 MSWORD DIVIDEND - MSWORD OF DIVISOR.
BCC 8 IF NO BORROW, IT WENT, CARRY IS CLEAR.

IT DIDN'T GO, SO WE SHIFT A 1 INTO THE QUOTIENT (COMPLEMENTED LATER).
CARRY IS SET.

6F: MOV (SP)+, R3 RESTORE LSWORD OF DIVIDEND.
MOV (SP)+, R2 RESTORE MSWORD OF DIVIDEND.
BR 101  ;GOTO SHIFT 1 INTO THE QUOTIENT.

101:  ;IT WENT, SO WE RESTORE THE STACK AND SHIFT A 0 INTO QUOTIENT (WILL BE
     ;COMPLETED LATER). CARRY IS CLEAR.

81:  MOV (SP)+,(SP)+  ;POP THE SAVED DIVIDEND OFF OF THE STACK.
     ;SHIF THE RESULT OF THE SUBTRACT ATTEMPT INTO THE QUOTIENT SHIFT REG.

101: ROL R5  ;SHIFT NEXT BIT INTO THE INVERTED QUOTIENT.
           CLC  ;DIVIDE THE
           ROR R1  ;DIVISOR BY
           ROR R4  ;2 (UNSIGNED).
           DEC R0  ;COUNT THIS SHIFT AND SUBTRACT.
           BNE 41  ;LOOP FOR ANOTHER SHIFT & SUB IF NOT DONE.
           COM R5  ;GET QUOTIENT FROM INVERTED QUOTIENT.

1:  ;NOW WE EITHER ROUND UP OR LEAVE QUOTIENT ALONE.

   ;NOW WE EITHER ROUND UP OR LEAVE QUOTIENT ALONE.

;CLEAN THE CARRY FOR THE SHIFT OF THE DIVIDEND.

121: ROL R3  ;MULTIPLY LSWORD OF DIVIDEND BY 2, MSWORD IS 0.
           BCS 121  ;IF CARRY FROM SHIFT, ROUND UP.
           SUB R4,R3  ;SUBTRACT DIVISOR FROM DIVIDEND.
           BCS 141  ;IF BORROW, DON'T ROUND UP.

141:  ;ROUND UP, EXTRA SUBTRACT WENT.

1:  ;ROUND UP, EXTRA SUBTRACT WENT.

121: INC R5  ;INCREMENT THE QUOTIENT BY ONE.
           BNE 141  ;IF NO OVERFLOW, WE LEAVE THE ROUND UP.
           DEC R5  ;DON'T LET ROUNING CAUSE OVERFLOW.

141:  ;ALL DONE, PASS QUOTIENT AND EXIT.

601:  ;RESTORE GPRS, LEAVE THE FOLLOWING INTACT:
     ;R1,R1LSOT(SP)  ;PUT R1 IN STACK SLOT.
     ;RETURN TO PREGS SUBRAT.
     ;R1 - 16 BIT, UNSIGNED QUOTIENT.
     ;CARRY - SET INDICATES NO OVERFLOW (SUCCESS).
SBITL GLOBAL SUBROUTINE - WAIBIS

********************************************************************************************************************
*  ** Wait for Bit Set Routine **
*  * This subroutine waits for the specified bit to become set. If the specified bit goes to a set state within the specified time out period a success indication is returned by this routine.
*  * The last value which is read looking for the condition is returned to allow the use of this routine to look for destructive read conditions.
*  *
*  * Inputs:
*  * R1 - Time-out Value and Bit Number Indication:
*       Bits 15 thru 12 - Number of Bit to Test (Range 0 thru 15).
*       Bits 11 thru 0 - Time-out Value in Milli-Seconds (4095 Max).
*  * R2 - Address of Word Containing the Bit to Test.
*  *
*  * Outputs:
*  * R2 - The last word which was read to check for the condition.
*  * Carry - Success flag (Carry set if bit set before time-out).
*  *
*  * Calling Sequence:
*  * MOV #130040,R1 ; Pass Bit 11 (13 Octal) and 32 (40 Octal) MS Delay.
*  * MOV @LABEL,R2 ; Test bit in word at "LABEL".
*  * JSR PC,WAIBIS ; Wait 32 ms for bit 11 to set.
*  *
*  * Comments:
*  *
*  * Subroutine Routines Called: MSLGET.
*  ********************************************************************************

WAIBIS::

SAVE

MOV R2,R4
JSR R5,PREGOS ; Call Register Save Subrát.

MOV R2,R2
MOV R1,R2
BIC #170000,R1 ; Separate delay count out of passed param.
BIC #7777,R2 ; Separate line number field of passed param.
SWAB R2 ; Put line number field in BYTE.
ASR R2 ; Shift the line number field into the proper
ASR R2 ; Position to use it as a word table offset
ASR R2 ; For the table lookup of the line bit map.
MOV BIT1B(R2),R2 ; Get bit map of line to test from table.
MOV R2,R3 ; Indicate that the bit should be set.
JSR PC,MSLGET ; Wait for the bit to be set within time out.

PASS R2

MOV R0,R2

608:

MOV R2,R2 ; Put R2 in stack slot.
JSR PC,R(SP) ; Return to PREGOS Subrát.

R2 - Last value read looking for condition.

RTS PC ; Carry - Success flag (set if bit found set).
SUBTL. GLOBAL SUBROUTINE

**aits FOR TX TO FINISH

** THIS SUBROUTINE IS USED IN THE FIMAWL.TST.

** IT WAITS FOR TRANSMISSION TO COMPLETE IE TX ACTION, THEN DELAYS

** FOR 5 MILLISECONDS TO ALLOW TIME FOR THE LAST CHARACTER TO GET INTO

** THE FIFO.

** INPUTS: CSRA CONTAINS THE ADDRESS OF THE CSR.

** OUTPUTS: CARRY - SET INDICATES SUCCESS.

** CALLING SEQUENCE: JSR PC,WAITTX

** COMMENTS:

** SUBORDINATE ROUTINES CALLED: DELAY,WAIB8.

---

WAITTX:: SAVE

JSR R5,PREG0S ;CALL REGISTER SAVE SUBRT.

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 60$ ;BRANCH IF NO TX ACTION, ABORT THE TEST.

MOV #5,R4 ;PASS DELAY OF 5 MILLI SECs.

JSR PC,DELAY ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.

SEC ;SET CARRY TO INDICATE SUCCESS.

60$: PASS

JSR PC,B(SP) ;RETURN TO PREG0S SUBRT.

RTS PC ;PASS THE CARRY BIT, SET INDICATES SUCCESS.
**GLOBAL SUBROUTINE**: WTWLNC

-- Line control register setup routine

- This subroutine is used to set the device under test (DUT) line control registers (LNCTRL) to the specified state. Only the LNCTRLs 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.
LNCTRA - Contains address of the DUT LNCTRL registers.

**Outputs**: LNCTRL - Specified DUT line control registers are altered.

**Calling Sequence**: JSR PC, WTWLNC

**Comments**: 

**Subordinate Routines Called**: ALTFLD.

WTWLNC:: SAVE 
JSR R5,PREG05 ; Call register save subroutine.

SET UP THE PARAMETERS FOR THE CALL TO ALTFLD.

MOV LNCTRLA,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 0-1,R4 ; Select all register bits to be altered.

CALL THE SUBROUTINE WHICH ALTERS THE REGISTER CONTENTS.

JSR PC, ALTFLD ; Alter the register contents.

601: PASS JSR PC, @ (SP) ; Restore GPRS.
RTS PC ; Return to PREG05 subroutine.
GLOBAL SUBROUTINE

.;SBTTL GLOBAL SUBROUTINE WTWLPR

;******************************************************************************
;*                                                                 *
;*  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:                                                                  *
;*  RO - NEW LINE PARAMETERS.                                                *
;*  RS - BIT MAP OF LINES TO BE ALTERED.                                    *
;*  CSR - CONTAINS ADDRESS OF THE DUT CSR.                                   *
;*  IESTAT - CONTAINS THE CURRENT STATE OF THE TX AND RX INTERRUPT          *
;*  ENABLE BITS IN THE CSR.                                                  *
;*  LPR - SPECIFIED DUT LINE PARAMETER REGISTERS ARE ALTERED.               *
;*                                                                 *
;*  OUTPUTS:                                                                 *
;*  LPR - SPECIFIED DUT LINE PARAMETER REGISTERS ARE ALTERED.               *
;*                                                                 *
;*  CALLING SEQUENCE:                                                       *
;*  JSR PC,WTWLPR                                                           *
;*                                                                 *
;*  COMMENTS:                                                               *
;*                                                                 *
;*  SUBORDINATE ROUTINES CALLED: ALTFLD.                                    *
;******************************************************************************

WTWLPR:: SAVE
       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 RO,R2 ;SET UP THE DESIRED REGISTER CONTENTS.
       MOV R5,R3 ;SET UP THE BIT MAP OF LINES TO ALTER.
       MOV 0-1,R4 ;SELECT ALL REGISTER BITS TO BE ALTERED.

       ;CALL THE SUBROUTINE WHICH ALTERS THE REGISTER CONTENTS.
       JSR PC,ALTFLD ;ALTER THE REGISTER CONTENTS.

       608:    PASS
       JSR PC,(SP)+ ;RESTORE GPRS.
       RTS PC ;RETURN TO PREG05 SUBRT.
SBLT1L INTERRUPT SERVICE ROUTINE: CLKINT

* THIS ROUTINE IS EXECUTED CLKHZ TIMES PER SECOND. IT DECREMENTS THE
* TWO TIMER COUNTERS DOWN TO ZERO.

* INPUTS:  TIMER1: TIMER COUNTER #1.
*          TIMER2: TIMER COUNTER #2.
*          TIMER3: TIMER COUNTER FOR CALL OF BREAK MACRO.

* OUTPUTS: THE 2 TIMER COUNTERS ARE DECREMENTED IF THEY ARE NOT ZERO.

* CALLING SEQUENCE:  PUT #CLKINT IN THE CLOCK INTERRUPT VECTOR SLOT.
*                    PUT THE DESIRED TIME PERIOD (SECONDS TIMES CLKHZ) IN
*                    EITHER TIMER1 OR TIMER2 AND POLL THE RESPECTIVE TIMERS
*                    COUNTER TO DETECT ITS GOING TO 0 ON TIME-OUT.

* COMMENTS:  THE 2 COUNTERS WILL NOT WRAPAROUND BUT WILL STOP AT 0. THIS
*            ALLOWS THE DETECTION OF A TIME-OUT ANY TIME AFTER THE TIME-OUT
*            HAS OCCURRED UNTIL THE TIMER COUNTER IS SET TO ANOTHER VALUE.

* SUBORDINATE ROUTINES CALLED: NONE.

CLKINT: 3 I
        BEQ 20:  ;BRANCH TO LEAVE IT AT ZERO IF IT IS ZERO.
        DEC 15:  ;DECREMENT TIME COUNTER.
        BEQ 40:  ;BRANCH TO LEAVE IT ALONE IF IT'S ALREADY ZERO.
        DEC 18:  ;DECREMENT TIME COUNTER.
        DEC 19:  ;DECREMENT THE BREAK COUNT.
        BNE 60:  ;EXIT IF NOT TIME TO CALL BREAK.
        MOV 16:  ;SET UP TIME TILL NEXT BREAK.
        MOV (SP),16:  ;SAVE CONTENTS OF RO FROM BREAK MACRO.
        MOV (SP)+,RO  ;RESTORE CONTENTS OF RO.
        RTI        ;CHECK FOR OPERATOR CONTROL/C. TRAP CIBAK
**SBTTL**  INTERRUPT SERVICE ROUTINE  RXDECT  

- RX DETECTION ROUTINE
- THIS ROUTINE DETECTS AN RX INTERRUPT BY SETTING THE RXINTC WORD TO 1.
- THIS ROUTINE IS USED IN THE RXTIMER TESTS.

**INPUTS:**  RXINTC - STORAGE FOR THE INTERRUPT COUNT.

**OUTPUTS:**  RXINTC - SET TO 1.

**CALLING SEQUENCE:**  PUT THE ADDRESS OF THE LABEL RXDECT IN THE VECTOR LOCATION.

**COMMENTS:**  THIS ROUTINE DOES NOT READ THE RFIFO.

**SUBORDINATE ROUTINES CALLED:**  NONE.

RXDECT::  MOV  @I,RXINTC  ;INDICATE THAT AN RX-INT HAS OCCURED.
**.SBTIL GLOBAL TRAP SERVICE ROUTINE - TP4RTN

**************************************************************************************************

| | BUS TIME-OUT TRAP (004 TRAP) SERVICE ROUTINE.
| | THIS ROUTINE DETERMINES IF THE 004 TRAP WAS CAUSED BY
| | AN "EXPECTED" ERROR OR NOT BY EXAMINING THE RETURN PC VALUE ON THE
| | STACK. IF THE TRAP IS UNEXPECTED, THIS ROUTINE JUMPS TO THE NORMAL
| | DIAGNOSTIC SUPERVISOR 004 TRAP HANDLING ROUTINE.

**INPUTS: SP - POINTS TO THE CPG WHERE THE TRAP OCCURRED.
| | ADDRPT - LABEL AT THE ADDRESS WHERE "EXPECTED" TRAPS OCCUR.
| | TP4FGLG - 004 TRAP FLAGS.

**OUTPUTS: TP4FGLG - BIT 15 IS SET IF "EXPECTED" TRAP OCCURRED.

**CALLING SEQUENCE: PUT ADDRESS POINTED TO BY TP4RTN IN 004 VECTOR.
| | OCCURRENCE OF 004 TRAP VECTORS TO THIS ROUTINE.

**COMMENTS: ANY 004 TRAP WHICH OCCURS AT AN ADDRESS OTHER THAN THAT LABELED
| | ADDRPT WILL BE HANDLED BY THE NORMAL 004 TRAP SERVICE ROUTINE.

**SUBORDINATE ROUTINES CALLED: NONE.

**TP4RTN: CMP (SP),#ADDRPT  ;COMPARE EXPECTED ADDR AGAINST TRAP RET PC.
| | BEQ 2$  ;IF THEY MATCH, CONTINUE THIS ROUTINE.
| | JMP @TP4VEC  ;IF NOT, JUMP TO NORMAL 004 TRAP SERVICE RTN.
| | BIS #BIT15,TP4FGLG  ;SET THE 004 TRAP OCCURED FLAG.
| | RTI  ;ALL DONE, GO BACK TO THE TEST.
.SBTIL REPORT CODING SECTION

;"; THE REPORT CODING SECTION CONTAINS THE
; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
;"

BGNRPT

EXIT RPT

.EVEN

ENDORPT

LIRPT::

.WORD J#JMP
.WORD L10017-2.

L10017: TRAP CIRPT
.SBTTL PROTECTION TABLE

; THIS TABLE IS USED BY THE RUNTIME SERVICES
; TO PROTECT THE LOAD MEDIA.

BGNPROT

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>020036</td>
<td>177777</td>
</tr>
<tr>
<td>020040</td>
<td>177777</td>
</tr>
<tr>
<td>020042</td>
<td>177777</td>
</tr>
</tbody>
</table>

-1 OFFSET INTO P-TABLE FOR CSR ADDRESS
1 OFFSET INTO P-TABLE FOR MESSBUS ADDRESS
-1 OFFSET INTO P-TABLE FOR DRIVE NUMBER

ENDPROT
**SBTTL INITIALIZE SECTION**

* * *

This section contains the code which is performed at the beginning of each pass or after a continue command.

This code performs the following actions:

- Moves the information held in the hardware p-table into the global data area.

* * *

```assembly
BGNINIT

; See if program just started, br if yes
READEF @"F.START"

MOV @"EF.START", $0
TRAP CREFG

BCOMPLETE NEWSSTA

; See if program just restarted, br if yes
READEF @"EF.RESTART"

MOV @"EF.RESTART", $0
TRAP CREFG

BCOMPLETE NEWRES

; See if this is a new pass, br if yes
READEF @"EF.NEW"

MOV @"EF.NEW", $0
TRAP CREFG

BCOMPLETE NEWPAS

; See if program was just continued
READEF @"EF.CONTINUE"

MOV @"EF.CONTINUE", $0
TRAP CREFG

BCOMPLETE GETPRM

JMP ENDT

NEWSSTA:

RESET THE BUS TO PREVENT ILLEGAL INTERRUPTS.

TRAP CIRESET

; Set up for line time clock interrupts.

; CLOCK L, R1

; Get the clock parameters.

MOV @"L.R0", $0
TRAP CICLCLK

MOV @"RO.R1"

; Store clock CSR address.

MOV (R1)+, CLKCSR

; Store clock bus reg int level.

MOV (R1)+, CLKBRL

; Store clock interrupt vector.

MOV (R1)+, CLKVEC

; Store clock frequency.

MOV (R1)+, CLKMHZ2

; Test for 50Hz line frequency.

CMP CLKMHZ2, $050

BNE 21

; Branch if clock is not 50Hz.
```
4637 020152 012767 000024 162144  MOV  #20, .MSTICK ;INDICATE 20MS PER CLOCK TICK.
4638 020160 000403  #21:  MOV  #17, .MSTICK ;INDICATE 17 MS PER CLOCK TICK.
4639 020161 012765 000221 162134  SETVEC  CLKVEC, #0LKINT, #0PRI06 ;INITIALIZE CLOCK INTERRUPT VECTOR.
4640 020170 012746 000300  MOV  #PRI06, (SP) ;MOV
4641 020174 012746 017632  CLKINT, (SP) ;MOV
4642 020200 016746 016214  CLKVEC, (SP) ;MOV
4643 020204 012746 000003  MOV  #5, (SP) ;MOV
4644 020210 104437  TRAP  C16VEC ;TRAP
4645 020212 062706 000010  ADD  #10, SP ;ADD
4646 020216 016700 162070  MOV  CLKMRZ, RO ;INITIALIZE THE BREAK COUNT
4647 020222 006300  ASL  RO ;TO CAUSE A BREAK
4648 020224 010067 162072  MOV  RO, BCOUNT ;EVERY 2 SECONDS.
4649 020230 012700 000240  SETPRI  #0PRI05 ;ALLOW CLOCK INTERRUPTS DISABLE OTHERS.
4650 020235 104441  TRAP  C16PRI ;TRAP

4650 020236 016767 157542 162032  MOV  4, TP4VEC ;SAVE THE EXISTING 004 TRAP VECTOR.
4651 020244 012767 017712 157532  MOV  @TP4RTN, 4 ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADDR.
4652 020245  ;ENABLE LTC CHECKING FOR 004 TRAP IN CASE CSR IS NOT THERE.
4653 020252 005067 162022  CLR  TP4FLG ;CLEAR THE 004 TRAP FLAG.
4654 020255 012767 000100 162016  MOV  #BIT6, WORD1 ;SET UP TO SET BIT 6 OF THE LTC CSR.
4655 020257 000100 162016  MOV  #1, WORD1 ;SET UP WORD 1 AS? THE OXTRAP MOVE SOURCE.
4656 020267 016701 162010  MOV  CLKCSR, #1 ;SET UP LTC CSR AS DESTINATION FOR OXTRAP MOVE.
4657 020274 004767 173666  JSR  PC, OXTRAP ;MOVE AND CHECK FOR TRAP.
4658 020300 016767 161772 157476  MOV  TP4VEC, 4 ;RESTORE THE NORMAL 004 TRAP VECTOR.
4659 020306 103457 161772  BCS  81 ;IF NO TRAP, LTC IS THERE SO CONTINUE.
4660 020310 005067 161776  CLR  CLXMRZ ;CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.
4661 020314 000402  BR  81 ;BYPASS THE FOLLOWING CALIBRATION PROCEDURES.
4662 020316 004767 173230  ;CALIBRATE THE DELAY ROUTINE MILLI-SECOND DELAY COUNT VALUE.
4663 020316 004767 173230  JSR  PC, CALMCSL ;CALMCSL.
4664 020316 004767 173230  ;CHECK FOR MEMORY MANAGEMENT PRESENT ON THIS MACHINE.
4665 020316 004767 173230  HAPPY ;IF MEM MGT IS PRESENT, DISABLE IT.
4666 020322 016767 157456 161746  MOV  4, TP4VEC ;SAVE THE EXISTING 004 TRAP VECTOR.
4667 020330 012767 017712 157446  MOV  @TP4RTN, 4 ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADDR.
4668 020335 005067 161734  CLR  TP4FLG ;CLEAR THE 004 TRAP FLAG.
4669 020336 005067 161734  CLR  WORD1 ;PREPARE TO CLEAR THE MEM MGT SRO REGISTER.
4670 020342 012700 002302  MOV  #MRTD1, RO ;SELECT CLEARED WORD AS OXTRAP RTN SOURCE.
4671 020346 012700 002302  MOV  MRTSR0, R1 ;SELECT MEM MGT SRO REGISTER AS DESTINATION.
4672 020352 016701 161752  MOV  MPPMRS, R1 ;INDICATE NO MEM MGT PRESENT IN CASE IT ISN'T.
4673 020356 005067 161750  CLR  MPRNAB ;INDICATE MEM MGT IS NOT ENABLED.
4674 020362 005067 161746  CLR  MPNENAB ;INDICATE MEM MGT IS NOT ENABLED.
4675 020366 004767 173574  JSR  PC, OXTRAP ;CLEAR THE MEM MGT SRO REG AND CHECK FOR TRAP.
4676 020372 016767 161700 157404  MOV  TP4VEC, 4 ;RESTORE THE NORMAL 004 TRAP VECTOR.
4677 020380 103003 161700  BCC  101 ;SKIP INDICATING MEM MGT PRESENT IF IT ISN'T.
4678 020390 000000 000000 016172  MOV  #1, MPPRES ;INDICATE THAT MEM MGT IS PRESENT.
4679 020410 005067 161650  CLRSNAB ;CLR COUNTER USED IN REPORTING ROM VERSION 4.
4680 020414 000167 000006  JMP  NEWNPAS ;SKIP AROUND THE BUS RESET, IT'S BEEN DONE.
NEWRES: BRESET
CLR PASCNT
MOV @-1,UNITN
GETPRM:
INC PASCNT
BNE GETPRM
DEC PASCNT
GETPRM:
INC UNITN
CMP UNITN,7
UNITN
BGE NEWPAS
MOV @-1,UNITN
UNITN
; RET the hardware parameters for this unit.
; GETPRM:
; INCREMENT LOGICAL DEVICE NUMBER
; SEE IF MAXIMUM UNIT NO. EXCEEDED
; BR IF YES
; GET P-TABLE POINTER INTO R1
; MOV UNITN,RO
; TRAP CKPSHRD
; MOV RO,R1
; BR GETPRM
; SKIP THIS DEVICE
; BR GETPRM
; BR GETPRM
; BR GETPRM
; BR GETPRM
; STORE DHU-11 CSR ADDRESS IN DEV.REG ADDRESS TABLE
; STORE DHU-11 CSR ADDRESS IN DEV.REG ADDRESS TABLE
; GET THE RX INTERRUPT VECTOR ADDRESS.
; STORE RX INT VECTOR ADDRESS.
; CALCULATE RX INTERRUPT VECTOR ADDRESS.
; STORE RX INT VECTOR ADDRESS.
; STORE DHU-11 ACTIVE LINE BIT MAP
; STORE DHU-11 LOOPBACK MODE
; COPY CSR ADDRESS
; INCREMENT CSR ADDRESS
; COPY BY 2.
; SET UP REGISTER COUNT
; GET LOCATION WHERE RBUF ADDRESS GOES IN TABLE
; STORE REGISTER ADDRESS IN TABLE
; INCREMENT REGISTER ADDRESS
; BY 2, FOR THE NEXT DEVICE REGISTER.
; DECREMENT REGISTER COUNT
; LOOP IF NOT DONE
; INITIALISE THE BMP CODE QUEUE.
; GET THE START ADDRESS OF THE QUEUE.
; GET THE END ADDRESS OF THE QUEUE.
; INITIALIZE SECTION

; SET THE POINTER TO THE START OF THE QUEUE.
4738 020572 010067 161612 MOV R0,SRPCQP

; CLEAR OUT THE CONTENTS OF THE QUEUE.
4739 020576 005020 CLR (R0),

; CHECK IF END OF QUEUE HAS BEEN REACHED.
4740 020600 020001 CMP R0,R1

; LOOP IF NOT ALL DONE.
4741 020602 103775 BLQ 14$

; REPORT THE UNIT NUMBER IF THE SOFTWARE TABLE QUESTION WAS ANSWERED YES.
4742 020604 032767 000020 AND MAXIMUM UNIT NUMBER IS GREATER THAN 1.

; BIT #BIT4,OPTION CHECK IF THE QUESTION WAS ANSWERED YES.
4743 020612 001416 BEQ 16$

; SKIP REPORTING UNIT NUMBER IF IT IS DISABLED.
4744 020614 026727 161172 CMP L#UNIT,01

; CHECK MAXIMUM NUMBER OF UNITS SELECTED.
4745 020616 000001 BLE 16$

; DO NOT REPORT UNIT NUMBER IF MAX NUMBER < 1.
4746 020618 003412 PRINTF #MFUNIT,UNITN REPORT UNIT NUMBER.

4747 020622 001674 161374 MOV UNIN, (SP)

4748 020624 016746 161374 MOV @MFUNIT,-(SP)

4749 020630 012746 005231 MOV @2,-(SP)

4750 020634 012746 000002 MOV SP,R0

4751 020640 010600 TRAP C$PRINTF

4752 020642 104417 ADD #6,SP

4753 020644 062706 000006

; ENDIT: CLR CTRLCF CLR THE CTRL-C TEST ABORT FLAG.
16$

; SET THE PROCESSOR PRIORITY TO DISABLE ALL INTERRUPTS.
4754 020650 005067 161374 SETPRI @PRI07 SET PROCESSOR PRIORITY TO 7.

4755 020652 012700 000340 MOV @PRI07,RO

4756 020654 020000 TRAP C$SPRI

4757 020660 104441 ENDINIT

4758 020662 L10021: TRAP C$INIT

4759 020664 104411

4760 000000 TNUM == 0 INITIALIZE THE ASSEMBLER TEST NUMBER VARIABLE.
..SBTL AUTODROP SECTION

***
:: THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
:: THE "ADR" FLAG WAS SET. THE UNIT(S) UNDER TEST ARE CHECKED TO
:: SEE IF THEY WILL RESPOND. THOSE THAT DON'T ARE IMMEDIATELY
:: DROPPED FROM TESTING.
::--

BGNAUTO

ENDAUTO

L1AUTO::

L10022: TRAP C1AUTO
.SBTTL DROP UNIT SECTION

; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
; TO NO LONGER BE TESTED.

BGN DU

PRINTF #DROP.R0 ; REPORT UNIT THAT HAS BEEN DROPPED.

MOV R0,-(SP)
MOV #DROP,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP CIPNTRF
ADD #6,SP

BR EDROP ; BRANCH AROUND THE MESSAGE.

DROP: .ASCIZ/#A UNIT#D6A DROPPED FROM FURTHER TESTING.

.EVEN

EDROP: .EXIT DU

L10024: TRAP CIDC
.SBTIL ADD UNIT SECTION

; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
; TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
; TO THE TEST CYCLE.

BGNAU

EXIT AU

.LWOR  JUMP

.WORD L10025-2:

.EVEN

ENDAU

.L10025:

TRAP C#AU
**.SBTLT HARDWARE TEST - ADRA -**

```assembly
*** ************************************************************************************
*** - REGISTER ADDRESS TEST -
*** ************************************************************************************

** THIS TEST VERIFIES THAT THE DEVICE REGISTERS WILL RESPOND TO THE PROPER **
** UNUSUAL HANDSHAKING SIGNALS WHEN ACCESED. IF THE DMU LL DOES NOT RESPOND **
** TO THE ACCESS ATTEMPTS (IF THE DMU LL IS AT THE WRONG ADDRESS, FOR EXAMPLE) **
** THE 004 BUS TIME-OUT TRAP IS DETECTED BY THIS ROUTINE AND AN ERROR **
** IS REPORTED. THIS TEST IS PERFORMED ON LINE 0 ONLY. **

```

```assembly
** BGTNSL **

```assembly
** MOV $01022,00001 **

```assembly
** MOV 00001,012767 **

```assembly
** MOV 012767,00001 **

```assembly
** MOV 012767,161224 **

```assembly
** MOV @NUM,TSTNUM **

```assembly
** SET UP THE TEST NUMBER. **

```assembly
** MOV #0-1,CTRLCF **

```assembly
** INDICATE THAT WE ARE IN A TEST. **

```assembly
** MOV #$01,ERARNR **

```assembly
** SET THE TEST ERROR NUMBER IN THE TABLE. **

```assembly
** MOV #EM0103,ERARMS **

```assembly
** SET UP THE TEST FAILURE MESSAGE IN THE TABLE. **

```assembly
** MOV #ER0101,ERARBLK **

```assembly
** SET-UP THE ERROR ROUTINE IN THE ERROR TABLE. **

```assembly
** SET UP TO CATCH ANY 004 TRAPS WHICH OCCUR: **

```assembly
** MOV $01052,012767 **

```assembly
** MOV 012767,011430 **

```assembly
** MOV 012767,162776 **

```assembly
** MOV 00005,01074 **

```assembly
** CLR R5 **

```assembly
** CLEAR THE ERROR FLAGS. **

```assembly
** HERE BEGINS THE LOOP TO TEST THE REGISTERS FOR A LINE. **

```assembly
** FIRST TEST THE CSR AND SET THE IND.ADR.REG (I.A.R) FIELD. **

```assembly
** MOV $01076,016700 **

```assembly
** MOV 016700,161124 **

```assembly
** MOV CSR.A,RO **

```assembly
** SET UP CSR AS THE CKTRAP MOVE SOURCE. **

```assembly
** MOV $021102,012701 **

```assembly
** MOV 012701,021274 **

```assembly
** MOV PC,CKTRAP **

```assembly
** SET UP DESTINATION LOCATION FOR CKTRAP MOVE. **

```assembly
** JSR PC,CKTRAP **

```assembly
** MOVE AND CHECK FOR TRAP. **

```assembly
** MOV #521,R1 **

```assembly
** SET UP REGISTER AS THE SOURCE FOR CKTRAP MOVE. **

```assembly
** MOV #521,R1 **

```assembly
** SET UP LOCAL STORAGE AS THE DES FOR CKTRAP. **

```assembly
** JSR PC,CKTRAP **

```assembly
** PERFORM THE MOVE, CHECK FOR TRAP. **

```assembly
** BCS R1,RO **

```assembly
** IF NO TRAP, BYPASS ERROR. **

```assembly
** MOV #100001,R5 **

```assembly
** SET UP ERROR a03 FOR ERROR TIME-OUT. **

```assembly
** BCS #17521 **

```assembly
** CLEAR THE I.A.R FIELD OF THE CSR DATA. **

```assembly
** MOV #161072 **

```assembly
** MOV 0161072,161072 **

```assembly
** MOV CSR.A,R1 **

```assembly
** USE OLD DESTINATION FOR SOURCE OF CKTRAP MOVE. **

```assembly
** JSR PC,CKTRAP **

```assembly
** MOVE AND CHECK FOR TRAP. **

```assembly
** BCS #521,R1 **

```assembly
** IF NO TRAP, BYPASS ERROR. **

```assembly
** MOV #100002,R5 **

```assembly
** SET UP ERROR a02 FOR ERROR TIME-OUT. **

```assembly
** BR 401 **

```assembly
** EXIT AND REPORT ERROR. **

```assembly
** NOW, WE TEST EACH REGISTER FOR THIS LINE. **

```assembly
** INIT REGISTER COUNTER TO 8. **

```assembly
** MOV #8,R2 **

```assembly
** SET UP REGISTER AS THE SOURCE FOR CKTRAP MOVE. **

```assembly
** MOV 012702,0000100 **

```assembly
** MOV #8,R2 **

```assembly
** INITIZE THE REGISTER POINTER. **

```assembly
** MOV CSR.A,501 **

```assembly
** SET UP REGISTER AS THE SOURCE FOR CKTRAP MOVE. **

```assembly
** MOV #521,R1 **

```assembly
** SET UP LOCAL STORAGE AS THE DES FOR CKTRAP. **

```assembly
** JSR PC,CKTRAP **

```assembly
** PERFORM THE MOVE, CHECK FOR TRAP. **

```assembly
** BCS #521,R1 **

```assembly
** IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS. **

```assembly
** MOV #100001,R5 **

```assembly
** SET UP ERROR a03 FOR ERROR TIME-OUT. **

```assembly
** MOV #100002,R5 **

```assembly
** USE OLD DEST AS SRC FOR CKTRAP MOVE. **

```assembly
** MOV R1,RO **

```assembly
** SET UP ERROR a02 FOR ERROR TIME-OUT. **

```assembly
** MOV 016701,000060 **

```assembly
** MOV 501,R1 **

```assembly
** SET UP REGISTER AS THE DEST FOR CKTRAP MOVE. **
```
4955 021212 004767 172750 JSR PC,CKTRAP ;PERFORM THE MOVE, CHECK FOR TRAP.
4956 021216 103402 BCS 124 ;IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS.
4957 021220 052705 100002 BIS $100002,R5 ;SET FATAL WRITE ERROR FLAGS.
4958 021224 005267 000042 124: INC 50$ ;INCREMENT THE REGISTER.
4959 021230 005267 000036 INC 50$ ;POINTER BY 2.
4960 021234 005302 DEC R2 ;COUNT THE REGISTER.
4961 021236 001351 BNE 81 ;LOOP TO TEST THE NEXT REGISTER ADDRESS.
4962
4963 ;DONE CHECKING DEVICE REGISTER ADDRESSES.
4964 ;REPORT ANY ERRORS AND EXIT.
4965
4966
4967
4968 021240 016767 161032 156536 401: MOV TP4VEC,4 ;RESTORE THE NORMAL 004 TRAP VECTOR.
4969 021246 005705 TST R5 ;CHECK THE ERROR FLAGS.
4970 021250 100012 BPL 60$ ;EXIT ROUTINE IF NO ERRORS.
4971
4972 ;REPORT "DEVICE REGISTER ACCESS TEST FAILED"
4973
4974 021252 104460 ERROR
4975
4976
4977 021254 016700 160744 DODU UNINT ;DROP THIS UNIT FROM FUTHER TESTING.
4978 021260 104451 MOV UNINT,RO
4979 021266 005067 160762 CLR CTRLCF ;INDICATE NO CTRL-C ABORT FROM TEST.
4980 021266 104444 QDCLN ;ABORT THIS SUB PASS.
4981
4982 021270 000402 BR 60$ ;
4983
4984 021272 000000 501: ;WORD 0 ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
4985 021274 000000 521: ;WORD 0 ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
4986
4987 021276 005067 160746 601: CLR CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.
4988
4989 021302 L10026: TRAP C#ETS!
4990 021302 104401
SBTL HARDWARE TEST  - DMSTA -

; THIS TEST VERIFIES THAT THE DMA START BIT IN THE DMU'S LINE CONTROL
; REGISTERS WILL INITIATE DMA TRANSMISSION ON THE SELECTED LINE.
; THIS TEST IS PERFORMED IN INTERNAL LOOPBACK, ON ALL ACTIVE LINES.

BGNST:

SETPRI #PRI05 ; ALLOW LTC INTERRUPTS.

MOV #PRI05,RO ; INITIALIZE ERROR TYPES AS IN ERROR TABLE.

TRAP CISPRI ; SET THE CORRECT ERROR REPORTING ROUTINE.

RESET THE DMU-11 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 >>>>> 4001 <<<<<<

JSR PC,CLKST ; RESET THE DMU-11, REPORT ANY ERRORS FOUND.
BCC 501 ; RESET FAILURE, ABORT THIS TEST.

JSR PC,INDATP ; INITIALISE THE 256 BYTE DATA PATTERN.

SET INTERNAL LOOPBACK, ENABLE 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 ACTIVE LINES.

MOV ADLCN5,R5 ; PASS THE ACTIVE LINE BIT MAP.
MOV #200,RO ; PASS THE LRACTL CONTENTS.
JSR PC,UTLNRG ; INITIALISE THE LPR CONTROLLERS.
MOV #177670,RO ; PASS THE LPR CONTENTS.
JSR PC,UTLNRG ; INITIALISE THE LPR REGISTERS ON ALL LINES.
JSR PC,SEXBL ; ENABLE TRANSMITTERS ON ALL LINES.

SET-UP OUTER LOOP TO TEST THE DMA START BIT ON ALL ACTIVE LINES.

MOV ADLCN5,R5 ; GET THE ACTIVE LINE BIT MAP.
CLR R1 ; CLEAR THE LINE NUMBER COUNTER.
CLR #4002,ERRNB ; CLEAR THE CARRY BIT PRIOR TO SHIFTING BIT MAP.
CLC ; SHIFT THE BIT MAP INTO THE CARRY BIT.
ROR R5 ; DO NOT TEST THE LINE IF IT IS INACTIVE.
JMP 14 ; PURGE THE FIFO.
JMP 301 ; GO REPORT ERROR IF FIFO WILL NOT CLEAR.

PERFORM DMA_START BIT TESTING ON EACH LINE INDIVIDUALLY.
TEST EACH DMA START BIT BEFORE TX'ING DATA PATTERN, REPORT ERROR IF SET.
SET DMA START BIT ON LUT, VERIFY IT IS SET, REPORT ERROR IF CLEAR.
Wait FOR DMA TO COMPLETE.
5044  021540  005267  162336
5045  021544  012702  002650
5046  021545  012703  000144
5047  021546  004767  172710
5048  021547  103067
5049  021548  05267  162336
5050  021549  010177  160524
5051  021550  105777  160534
5052  021551  100060
5053  021552  005267  162354
5054  021553  010177  160524
5055  021554  105777  160534
5056  021555  100060
5057  021556  005267  162336
5058  021557  010177  160524
5059  021558  105777  160534
5060  021559  100060
5061  021560  101013
5062  021561  170226
5063  021562  16702  160500
5064  021563  004767  175704
5065  021564  103045
5066  021565  012704  000005
5067  021566  004767  172516
5068  021567  010301
5069  021568  005267  162300
5070  021569  010177  160450
5071  021570  105777  164640
5072  021571  100432
5073  021572  005267  162300
5074  021573  010177  160450
5075  021574  105777  164640
5076  021575  100432
5077  021576  005267  162336
5078  021577  010177  160450
5079  021578  105777  164640
5080  021579  100432
5081  021580  005003
5082  021581  012704  002000
5083  021582  012767  007647  162252
5084  021583  004767  174652
5085  021584  040200
5086  021585  011007
5087  021586  005267  162230
5088  021587  017702  160424
5089  021588  100021
5090  021589  170301
5091  021590  040200
5092  021591  011007
5093  021592  005267  162230
5094  021593  004767  174652
5095  021594  040200
5096  021595  005003
5097  021596  014222
5098  021597  007577
5099  021598  005203
5100  021599  001444
5101  021600  002753
5102  021601  005003
5103  021602  012704  002000
5104  021603  012767  007647  162252
5105  021604  004767  174652
5106  021605  040200
5107  021606  011007
5108  021607  005267  162230
5109  021608  004767  174652
5110  021609  040200
5111  021610  005003
5112  021611  014222
5113  021612  007577
5114  021613  005203
5115  021614  001444
5116  021615  002753
5117  021616  005003
5118  021617  012704  002000
5119  021618  012767  007647  162252
5120  021619  004767  174652
5121  02161A  040200
5122  02161B  011007
5123  02161C  005267  162230
5124  02161D  004767  174652
5125  02161E  040200
5126  02161F  005003
5127  021620  014222
5128  021621  007577
5129  021622  005203
5130  021623  001444
5131  021624  002753
BR 141: ;SKIP AROUND THE ERROR REPORT.

; REPORT ERROR, SKIP FURTHER TESTING ON THIS LINE.
MOV R3,R1 ;RESTORE THE CURRENT LINE NUMBER.

MOV #EM4002,R2 ;PASS THE ERROR MESSAGE TO BE REPORTED.

ERROR ;"DM1A,START BIT BAD ON LINE NN".

TRAP CERROR ;>>>>> ERROR <<<<

;EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
BEQ 601: ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

INC R1 ;INCREMENT THE LINE NUMBER COUNTER.

TST R5 ;ARE THERE ANY MORE ACTIVE LINES TO TEST?.

BNE 21: ;YES; BRANCH TO TEST THE NEXT LINE.

BR 601: ;NO; EXIT THIS TEST.

JSR PC,TSABRT ;REPORT TEST ABORTED. NON-TEST RELATED ERROR.

CLR CTRALCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDSTST

L10027: TRAP CIETST
..SBTIL  HARDWARE TEST   .DMABRT..

******************************************************************************
| ** DMABRT TEST **
| ** THIS TEST VERIFIES THAT EACH DMA_ABORT_BIT WILL CORRECTLY HALT A DMA **
| ** TRANSMISSION, AND RETURN A TX_ACTION. **
| ** IT WILL ALSO VERIFY THAT THE ABORTED DMA TRANSMISSION CAN BE RESUMED, **
| ** AND THAT A TX_ACTION IS RETURNED UPON COMPLETION. **
| ** THIS TEST IS PERFORMED IN INTERNAL LOOPBACK, ON ALL ACTIVE LINES. **

******************************************************************************

BGNST

SETPRI @PRI05

; ALLOW LTC INTERRUPTS.

MOV @PRI05,RO

TRAP C15PMT

T3:

TNUM := TNUM + 1

; INCREMENT THE ASSEMBLY TIME TEST COUNTER.

MOV @TNUM,TSTNUM

; SET UP THE TEST NUMBER.

(41)

MOV 0,-CTRCLCF

; INDICATE THE ERROR TYPE FROM THE FIFO.

MOV $1,ERROR

; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

MOV @ERROR,ERROR

; SELECT THE CORRECT ERROR REPORTING ROUTINE.

TNUM := 0

; CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.

BCC 60

; THIS SUBROUTINE REPORTS ERROR 401.

60:

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.

JSR PC,CLNRT

; CLEAR THE DMABRT TEST.

JSR PC,INATP

; INITIALISE 256 BYTE DATA PATTERN.
INC EARNBR ;INCREMENT ERROR NUMBER TO 4103.
MOV R1,CSRA ;SELECT THE LINE CURRENTLY UNDER TEST.
BIT @B100,DLNCTRA ;TEST THE STATE OF THE DMA_ABORT BIT.
BNE 68 ;GO REPORT ERROR IF BIT IS SET.

;ENABLE DMA TX ON SELECTED LINE, WAIT FOR DMA TO TX APPROX 1/4 OF DATA.
;ABORT THE DMA TRANSMISSION, WAIT FOR TX_ACTION TO BE RETURNED.

INC EARNBR ;SET ERROR NUMBER TO 4104.
MOV @BUBAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
MOV @B56,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
JSR PC,DMCH ;TRANSMIT THE DATA PATTERN.
BCC 50 ;GO REPORT ERROR IF THERE ARE TX PROBLEMS.

;WAIT FOR DMA TO TRANSMIT 1/4 OF THE DATA BEFORE ABORTING.

MOV R1,CSRA ;SELECT THE LINE CURRENTLY UNDER TEST.
MOV @B40,R4 ;PASS THE DELAY TIME OF 40 MILLI SECONDS.
JSR PC,DEL ;WAIT FOR APPROX 1/4 OF DATA TO BE TX'D.
BIS @B3I0,DLNCTRA ;ABORT THE DMA TRANSMISSION.

;WAIT FOR TX_ACTION TO BE RETURNED, REPORT ERROR IF TIME-OUT OCCURS.

INC EARNBR ;INCREMENT ERROR NUMBER TO 4105.
MOV R1,R3 ;SAVE THE LINE NUMBER.
MOV @I90012,R1 ;TEST BIT IS, TIMEOUT OF 10 MILLI SECS.
MOV CSRA,R2 ;PASS THE ADDRESS OF THE REGISTER TO TEST.
JSR PC,WAIB ;WAIT FOR DMA TO COMPLETE.
BCC 41 ;GO REPORT ERROR IF TIMEOUT OCCURRED.

;VERIFY DMA.START BIT CLEAR, REPORT ERROR IF SET.

INC EARNBR ;INCREMENT ERROR NUMBER TO 4106.
MOV @EM4105,R2 ;SELECT MESSAGE TO BE REPORTED.

;"DMA.START BIT FOUND SET AFTER DMA ABORTED".
MOV R1,CSRA ;SELECT THE LINE CURRENTLY UNDER TEST.
TSTB @TXAD2A ;TEST THE STATE OF THE DMA.START BIT.
BHI 81 ;GO REPORT ERROR IF IT IS SET.

;RESUME DMA TRANSMISSION BY CLEARING DMA_ABORT AND SETTING DMA.START.

BIC @B100,DLNCTRA ;CLEAR THE DMA_ABORT BIT.
BIS @B117,DTXAD2A ;SET THE DMA.START BIT.

;WAIT FOR DMA TRANSMISSION TO COMPLETE.

INC EARNBR ;INCREMENT ERROR NUMBER TO 4107.
MOV R1,R3 ;SAVE THE LINE NUMBER.
MOV @I70536,R1 ;TEST BIT IS, TIMEOUT OF 350 MILLI SECS.
MOV CSRA,R2 ;PASS THE ADDRESS OF THE REGISTER TO TEST.
JSR PC,WAIB ;WAIT FOR DMA TO COMPLETE.
BCC 41 ;GO REPORT ERROR IF TIMEOUT OCCURRED.
MOV @B2,R4 ;PASS TIME-OUT OF 2 MILLI SECS.
JSR PC,DEL ;WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.
MOV R3,R1 ;RESTORE THE CURRENT LINE NUMBER.
5239  TEST THE STATE OF THE DMA_ABORT BIT ON THE LINE UNDER TEST.
5240  REPORT ERROR IF DMA_ABORT BIT IS SET.
5241  INC ERRNBR  ; INCREMENT ERROR NUMBER TO 4108.
5242  MOV R1,DECRA  ; SELECT THE LINE CURRENTLY UNDER TEST.
5243  BIT #BIT0,SLNCTRA  ; TEST THE STATE OF THE DMA_ABORT BIT.
5244  BNE 6$  ; GO REPORT ERROR IF BIT IS SET.
5245  BR 10$  ; BRANCH TO CHECK FOR ANY MORE LINES TO TEST.
5246  4$:
5247  ; REPORT ERROR, SKIP FURTHER TESTING ON THIS LINE.
5248  MOV R3,R1  ; RESTORE THE CURRENT LINE NUMBER.
5249  6$:
5250  MOV #EM4102,R2  ; PASS THE ERROR MESSAGE TO BE REPORTED.
5251  "DMA_ABORT BIT BAD ON LINE NN".
5252  ERROR  ; "" >>>>>> ERROR <<<<<
5253  TRAP C$ERROR
5254  10$:
5255  EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
5256  EXIT WITH TEST FAILURE MESSAGE IF
5257  BIT #BIT06,OPTION  ; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
5258  BEQ 60$  ; DURING THE SOFTWARE QUESTIONS.
5259  60$:
5260  VERIFY ALL ACTIVE LINES HAVE BEEN TESTED.
5261  INCR R1  ; INCREMENT THE LINE NUMBER COUNTER.
5262  TST R5  ; ARE THERE ANY MORE ACTIVE LINES TO TEST?.
5263  BNE 2$  ; YES: BRANCH TO TEST THE NEXT LINE.
5264  AR 60$  ; NO: EXIT THIS TEST.
5265  2$:
5266  JSR PC,TSTABRT  ; REPORT TEST ABORTED: NON-TEST RELATED ERROR.
5267  CLR CTRLCF  ; INDICATE THAT WE ARE NOT WITHIN A TEST.
5268  50$:
5269  60$:
5270  ENDST
5271  10030:  TRAP C$TEST
**SBTTL HARDWARE TEST - DMAERR**

**DMA ERROR BIT TEST**

- This test verifies that the TX_DMA_ERROR bit in the CSR is functioning correctly. The DMA error is forced by making the DUT attempt to perform a DMA transfer from the address of its own CSR since the device cannot be both a bus master and slave at the same time. Timeout will occur waiting for the appropriate handshake signal.

- This test is performed in internal loopback.

```assembly
BGNST

SETPRI #PRI05

; ALLOW LTC INTERRUPTS.

MOV #PRI05,0

TRAP 0C8H

TA:;

TAU = TNUM ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.

MOV @TNUM,TNUM

; SET UP THE TEST NUMBER.

MOV #1,CTRLCF

; SET ERROR TYPE AS FATAL IN ERROR TABLE.

MOV #1,ERRTY

; SET THE FIRST ERROR NUMBER IN ERROR TABLE.

MOV #ERRA01.ERRRBR

; SET ERROR MESSAGE ADDRESS IN ERRBL.

MOV @ERRA01.ERRRBL

; SELECT THE CORRECT ERROR REPORTING ROUTINE.

; RESET THE DUT TO A KNOWN STATE. REMOVE THE STATUS CODES FROM THE IP.

; CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.

; THIS SUBROUTINE REPORTS ERROR 4201 <<<<<.

; JSR PC.CLNRST ; RESET THE DHU-11, REPORT ANY ERRORS FOUND.

; BCC 601 ; RESET FAILURE?, ABORT THIS TEST.

; SET INTERNAL LOOPBACK, ENABLE READER FUNCTIONS ON ALL LINES.

; SET LPR ON ALL LINES TO 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY.

; 2 STOP BITS.

; ENABLE TRANSMITTERS ON ALL LINES.

; JSR PC.INACT ; FIND AN ACTIVE LINE.

; BCC 601 ; EXIT THE TEST IF NO ACTIVE LINES.

; MOV R1,R0

; SAVE THE LINES NUMBER.

; MOV #004,RO

; PASS THE LNCRL CONTENTS.

; JSR PC,WTLC

; INITIALISE THE LNCRL REGISTERS.

; MOV #177676,RO

; PASS THE LPR CONTENTS.

; JSR PC,WTLP

; INITIALISE THE LPR REGISTERS ON THE ACTIVE LINE.

; JSR PC,XTNBL

; ENABLE TRANSMITTERS ON THE ACTIVE LINE.

; VERIFY THAT THE DMA START BIT IS CLEAR BEFORE ATTEMPTING THE DMA TRANSFER.

; INC ERRBR ; SET THE ERROR NUMBER TO #202.

; BIT #0T07,BTXAD2A ; TEST THE DMA START BIT.

; BNE 501 ; BRANCH TO REPORT THE ERROR IF THE BIT IS SET.

; SET UP THE DMA REGISTERS TO PERFORM THE TRANSFER FROM THE ADDRESS OF THE CSR.
```
MOV CSRA, @IXA01A ; SET UP THE LOW 16 BITS OF THE DMA ADDR.
MOV @IXA02A, CSRA ; UP TO DMA ONE CHARACTER.
MOV #203, @IXA02A ; SET UP THE 2 MSB'S AND INITIATE THE DMA.
MOV #170012, R1 ; TEST BIT 15, TIME OUT OF 10 MS.
MOV CSRA, R2 ; INDICATE TO TEST THE CSR.
INC ERRNBR ; SET THE ERROR NUMBER TO 4203.
JSR PC, WAIBIS ; WAIT FOR A TX-ACTION.
BCC $50 ; REPORT THE ERROR IF NO TX-ACTION.

; VERIFY THAT THE DMA ERROR BIT IS SET AND THE DMA START BIT IS CLEAR.
INC ERRNBR ; SET THE ERROR NUMBER TO 4204.
BIT #BIT12, @CSRA ; TEST THE DMA-ERROR BIT.
BEQ $24 ; REPORT THE ERROR IF BIT IS CLEAR.
INC ERRNBR ; SET THE ERROR NUMBER TO 4205.
BIT #BIT07, @IXA02A ; TEST THE DMA-START BIT.
BNE $50 ; REPORT THE ERROR IF THE BIT IS SET.

; VERIFY THAT THE DMA ERROR BIT CLEARS WHEN A "GOOD" DMA TRANSFER IS PERFORMED.
MOV R4, R1 ; SET UP THE ACTIVE LINE NUMBER.
MOV @BUFBA0. R2 ; SET UP THE START ADDRESS OF THE DMA BUFFER.
INC ERRNBR ; SET THE ERROR NUMBER TO 4206.
JSR PC, DODMA ; START THE DMA.
BCC $50 ; REPORT THE ERROR IF ONE OCCURRED.
MOV #170036, R1 ; SET UP TO TEST BIT 15 WITH TIMEOUT OF 30 MS.
MOV CSRA, R2 ; INDICATE TO TEST THE CSR.
INC ERRNBR ; SET THE ERROR NUMBER TO 4207.
JSR PC, WAIBIS ; WAIT FOR A TX-ACTION.
BCC $50 ; REPORT THE ERROR IF NO TX ACTION OCCURED.
INC ERRNBR ; SET THE ERROR NUMBER TO 4208.
BIT #BIT12, R2 ; TEST THE DMA ERROR BIT OF THE LAST CSR WORD READ.
BEQ $60 ; EXIT THE TEST IF THE BIT IS CLEAR.

; REPORT THE ERROR, DMA ERROR BIT BAD.
MOV @EM4202, R1 ; SET THE MESSAGE
"DMA ERROR BIT BAD",

; TRAP C$ERROR
BR $60 ; EXIT THE TEST.
JSR PC, TSABRT ; REPORT THE NON-RELATED TEST ERROR.
CLR CTRLCF ; INDICATE THAT WE ARE NOT \_\_\_\_ IN A TEST.
ENDST

L10031: TRAP C$TEST
**SBTTL HARDWARE TEST**

**SBIYL HARDWARE TEST**

**D8UTOI D0 AUTO BIT INACTIVE TEST**

* THIS TEST VERIFIES THAT THE DUT'S D8UTOI FUNCTION BEHAVES CORRECTLY.
* WHEN INACTIVE, IE D8UTOI BIT CLEAR.
* THIS TEST WILL ONLY EXECUTE IF STAGGERED LOOPBACK MODE IS SELECTED.
* THE SPECIAL STAGGERED LOOPBACK CONNECTOR MUST BE FITTED.

**BONSTT**

CMPI B LOPBACK,#02
JMP 601 ;DO NOT EXIT IF STAGGERED LOOPBACK MODE SELECTED.
SETPRI 09105 ;EXIT THIS TEST.
MOV @PRI05,RO
TRAP C68PR

TNUM ** TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV @TNUM,1STNUM ;SET UP THE TEST NUMBER.
MOV @#1,CTRCLF ;INDICATE THAT WE ARE IN A TEST.
MOV 0,#1,ERRTP ;SET ERROR TYPE AS FATAL IF ERR TABLE.
MOV #4901,ERRBKR ;SET ERROR NUMBER TO 4901.
MOV #EM4901,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERR TABLE.
MOV #ERR4911,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.

RESET THE OUT TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO.
CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.

**SUBROUTINE REPORTS ERROR ******* 4901 *******.

JSR PC,CLNRST ;RESET THE DMU-11, REPORT ANY ERRORS FOUND.
JSR PC,ASNLST ;INITIALISE THE ASSOCIATED TX/RX TABLES.

SET-UP THE ASSOCIATED TX/RX LINE NUMBER TABLES.

JSR PC,ASNLST ;INITIALISE THE ASSOCIATED TX/RX TABLES.

SET EXTERNAL LOOPBACK, DISABLE D8UTOI AND ENABLE RECEIVER ON ALL ACTIVE LINES.
SET LPR ON ALL LINES TO 38.4K BAUD, 8 BITS PER CHARACTER, 000 PARITY, 2 STOP BITS.
ENABLE TRANSMITTERS ON ALL LINES.

MOV AQLNS,R5 ;PASS THE ACTIVE LINE BIT MAP.
MOV #4,RO ;PASS THE LMCTRL CONTENTS.
JSR PC,LMNCR ;INITIALISE THE LMCTRL REGISTERS.
MOV @QALNS,R5 ;PASS BIT MAP OF ALL LINES.
MOV #177670,RO ;PASS THE LPR CONTENTS.
JSR PC,LMNLPR ;INITIALISE THE LPR REGISTERS ON ALL LINES.
JSR PC,TTXMRL ;ENABLE TRANSMITTER ON ALL LINES.

SET UP OUTER LOOP FOR TESTING ACTIVE LINES IN BOTH LINE GROUPS.
MOV #1000000,R3 ;SET UP LOOP CONTROL FLAG.
MOV AQLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
BIC LQRP2M,R5 ;REMOVE LINES IN GROUP 2.
5430 023050 010567 0000404 21: MOV R5,451 'SAVE THE CURRENT LINE GROUP.
5431 023054 005067 000076 'CLEAR THE LINE NUMBER COUNTER.
5432 023060 016701 000372 48: CLR R01 'CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
5433 023064 000241 'COPY THE LINE NUMBER.
5434 023066 006005 'CLEAR ACTIVE LINE BIT MAP INTO CARRY BIT.
5435 023070 100364 BCC 81 'SKIP TESTING THIS LINE IF IT IS INACTIVE.

5437 'TEST THE STATE OF THE QAUTO BIT ON THE LINE UNDER TEST.
5438 'REPORT ERROR IF IT IS FOUND SET, AND SKIP FURTHER TESTING OF THAT LINE.

5440 023072 012767 011446 160752 MOV #4902,ERRORB 'SET THE ERROR NUMBER TO 4902.
5441 023100 010117 157122 'SELECT THE LINE TO BE TESTED.
5442 023104 032777 000020 157124 BIT #BIT4,BL,CTRLA 'TEST THE STATE OF THE QAUTO BIT.
5443 023112 001410 BEQ 61 'SKIP ERROR REPORT IF QAUTO BIT IS CLEAR.
5444 023114 012702 005760 MOV #E4902,R2 'PASS THE ERROR MESSAGE.
ERROR 61: "QAUTO BIT BAD ON LINE "

5446 023120 104460 ERRORB '>>ERROR 4902 <<<<<<
TRAP 'CERROR
5447
5448
5449
5450
5451 023122 032767 000100 157060 'EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED.
5452 023130 001556 'EXIT WITH TEST FAILURE MESSAGE IF NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED.
5453
5454
5455 023132 000443 'DURING THE SOFTWARE QUESTIONS.
5456
5457
5458
5459 023134 116177 004010 157064 61: MOV B,ASCII DCS 'SELECT THE ASSOCIATED ASCII XOFF (ASCII DCS) ON THE ASSOCIATED LINE.
5460 023142 112777 000023 157064 MOV B,DATA 'TRANSMIT THE XOFF CHARACTER TO THE LUT.
5461
5462
5463
5464 023150 005267 160676 'WAIT FOR TRANSMISSION TO COMPLETE.
5465 023154 012701 170012 'INCREMENT ERROR NUMBER TO 4903.
5466 023160 016702 157042 MOV R1,CSR,AR 'PASS THE ADDRESS OF THE REGISTER TO TEST.
5467 023164 004767 174246 MOV CSR,AR 'PASS THE ADDRESS OF THE REGISTER TO TEST.
5468 023170 103134 JSR PC,habis 'WAIT FOR TRANSITION TO COMPLETE.
5469 023172 002704 000005 'ABORT TEST IF TIMEOUT OCCURRED.
5470 023176 004767 171060 MOV #5,R4 'PASS TIME-OUT OF 5 MILLISECS.
5471 JSR PC,DELAT 'WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.

5472
5473
5474
5475 023202 005267 160644 'TEST THE STATE OF THE TX_ENABLE BIT ON THE LINE UNDER TEST.
5476 023206 016701 000024 'REPORT ERROR IF TX_ENABLE BIT IS CLEAR.
5477 023212 010377 157010 'INCREMENT ERROR NUMBER TO 4904.
5478 023216 005777 157020 MOV R1,CSR,AR 'GET THE NUMBER OF THE LINE TEST.
5479 023222 100407 'SELECT THE LINE CURRENTLY UNDER TEST.
5480 023224 012702 005760 JSR BITXADR2 'TEST THE STATE OF THE TX EN JLE BIT.
5481 023230 104460 BMI 81 'SKIP ERROR REPORT IF BIT IS SET.

5483
5484
EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION
EXIT WITH TEST FAILURE MESSAGE IF NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED DURING THE SOFTWARE QUESTIONS.

INC 401
INCREMENT THE LINE NUMBER.

TST R5
CHECK IF THERE ARE ANY MORE LINES TO TEST.

BNE 41

DISABLE TRANSMITTERS ON THE SELECTED LINES IN THE CURRENT LINE GROUP.

MOV 454,R5
RESTORE THE CURRENT LINE ACTIVE LINE GROUP.

JSR PC,TXOSBL
DISABLE TRANSMITTERS ON THE SELECTED LINES.

MOV 454,R5
GET THE CURRENT ACTIVE LINE GROUP AGAIN.

CLR 401
CLEAR THE LINE NUMBER.

MOV #4905,,ERRNBR
SET ERROR NUMBER TO 4905.

MOV 401,R1
COPY THE LINE NUMBER.

CLC CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.

RDR R5 SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.

BCC 124
SKIP TESTING THIS LINE IF IT IS INACTIVE.

TRANSMIT THE XON (ASCII DC1) ON THE ASSOCIATED LINE.

MOVX TXLN,($R1),BCSR
SELECT THE ASSOCIATED TX LINE.

MOVB $21,OFDMATA
TRANSMIT THE XON CHARACTER TO THE LUT.

WAIT FOR TRANSMISSION TO COMPLETE.

MOV #170012,R1 TEST BIT 15, TIMEOUT OF 10 MILLISECS.

MOV CXR,RA2
PASS THE ADDRESS OF THE REGISTER TO TEST.

JSR PC,WAIHIS WAIT FOR TRANSMISSION TO COMPLETE.

BCC 501
ABORT TEST IF TIMEOUT OCCURRED.

MOV #5,R4 PASS TIME-OUT OF 5 MILLISECS.

JSR PC,DELAY
WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.

TEST THE STATE OF THE TX ENABLE BIT ON THE LINE UNDER TEST.

REPORT ERROR IF TX_ENABLE BIT IS SET.

INC ERRNBR INCREMENT ERROR NUMBER TO 4906.

MOV 401,R1
GET THE NUMBER OF THE LINE UNDER TEST.

MOV R1,BCSR
GIVE THE LINE CURRENTLY UNDER TEST.

TST @TXAD2A TEST THE STATE OF THE TX_ENABLE BIT.

BPL 121 SKIP ERROR REPORT IF BIT IS CLEAR.

MOV #EN4902,R2 PASS THE MESSAGE TO BE REPORTED.

"AUTO BIT BAD ON LINE NN" TRAP CERROR

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION EXIT WITH TEST FAILURE MESSAGE IF NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED DURING THE SOFTWARE QUESTIONS.

INC 401 INCREMENT THE LINE NUMBER.
5541 023420 005705  TST    R5  ;CHECK IF THERE ARE ANY MORE LINES TO TEST.
5542 023422 001523  BNE  101  
5543
5544  ;CHECK LOOP CONTROL FLAG TO DETERMINE IF BOTH SETS OF LINES HAVE BEEN TESTED
5545  ;IF THIS IS THE FIRST TIME AROUND, RE-ENABLE TX ON ALL LINES, GENERATE ACTIVE
5546  ;BIT MAP FOR SECOND LINE GROUP.
5547
5548 023424 005703  TST    R3  ;HAVE BOTH LINE GROUPS BEEN TESTED?.
5549 023426 001417  BEQ  601  ;YES; THEN EXIT THIS TEST.
5550 023430 005003  CLR    R5  ;NO; CLEAR THE LOOP CONTROL FLAG.
5551 023432 012705 177777  MOV    MAPLSN,R5  ;PASS THE BIT MAP OF ALL AVAILABLE LINES.
5552 023436 004767 173460  JSR    PC,TXENBL  ;RE-ENABLE TRANSMISSION ON ALL LINES.
5553 023442 016705 156552  MOV    ACILNS,R5  ;GET THE ACTIVE LINE BIT MAP.
5554 023446 046705 156606  BIC    LGRP1M,R5  ;REMOVE ALL ACTIVE LINES IN GROUP 1.
5555 023452 000167 177372  JMP    21  ;ONCE MORE AROUND AND WE ARE DONE.
5556
5557 023456 000000 401:  .WORD    0  ;STORAGE FOR CURRENT LINE NUMBER.
5558 023460 000000 451:  .WORD    0  ;STORAGE FOR CURRENT ACTIVE LINE BIT MAP.
5559 023462 004767 173204 501:  JSR    PC,TSAVRT  ;REPORT TEST ABDORTED. NON TEST RELATED ERROR.
5560 023466 005067 156856 601:  CLR    CTRLCF  ;INDICATE THAT WE ARE NOT WITHIN A TEST.
5561
5562 023472 000000  L10032:  TRAP  C$E7ST
.SBTTL  HARDWARE TEST     -  OAUTOA -

*  OAUTO BIT ACTIVE TEST  *

*  THIS TEST VERIFIES THAT THE DUT'S OAUTO FUNCTION BEHAVES CORRECTLY *
*  WHEN ACTIVE.  IF OAUTO BIT ASSERTED HIGH, *
*  THIS TEST WILL ONLY EXECUTE IF THE STAGGERED LOOPBACK MODE IS SELECTED. *
*  THE SPECIAL STAGGERED LOOPBACK CONNECTOR MUST BE FITTED. *

*  BGNST  *

023474  CMPB  LOPBCK,#2  ; CHECK MODE SELECTED.
023477  DPL   #6  ; DO NOT EXIT IF STAGGERED LOOPBACK MODE SELECTED.
02347A  JMP   601  ; EXIT THIS TEST.
02347D  SETPRI  #PRIO5  ; ALLOW LTC INTERRUPTS.

02347F  MOV  #PRIO5,RO  ; TRAP  CISPRI

023481  TGI:

023483  TNUM = TNUM + 1  ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.

023485  MOV  #TNUM,TISNUM  ; SET UP THE TEST NUMBER.  (50)

023488  MOV  #1,CTRLCF  ; INDICATE THAT WE ARE IN A TEST.

02348B  MOV  #1,ERRTYP  ; SET ERROR TYPE AS FATAL IN ERROR TABLE.

02348E  MOV  #5001,ERRNR  ; SET ERROR NUMBER TO 5001.

023491  MOV  #EM5001,ERRMSG  ; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

023494  MOV  #ERR9101,ERRBLK  ; SELECT THE CORRECT ERROR REPORTING ROUTINE.

023497  ; SET THE DUT TO A KNOWN STATE.  REMOVE THE STATUS CODES FROM THE FIFO.

02349A  ; CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.

02349D  ; THIS SUBROUTINE REPORTS ERROR >>>>> 5001 <<<<<<.

02349F  JSR  PC,CLNRST  ; REFORMAT THE DMA-11, REPORT ANY ERRORS FOUND.

0234A2  BCS   #6  ; DO NOT EXIT IF REFORMAT WAS SUCCESSFUL.

0234A5  JMP   601  ; EXIT THIS TEST.

0234A8  ; SET UP THE ASSOCIATED TX/RX LINE NUMBER TABLES.

0234AB  JSR  PC,ASLNTL  ; INITIALISE THE ASSOCIATED TX/RX TABLES.

0234AE  ; SET EXTERNAL LOOPBACK, ENABLE OAUTO AND RECEIVER FUNCTIONS ON ALL ACTIVE LINES.

0234B1  ; SET LPR ON ALL LINES TO 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY,
0234B4  ; 2 STOP BITS.

0234B7  ; ENABLE TRANSMITTERS ON ALL LINES.

0234BB  MOV  ACTLNS,R5  ; PASS THE ACTIVE LINE BIT MAP.

0234BD  MOV  #24,RO  ; PASS THE LCLNTL CONTENTS.

0234BF  JSR  PC,WTLNC  ; INITIALISE THE LCLNTL REGISTERS.

0234C2  MOV  #MAPLNS,R5  ; PASS BIT MAP OF ALL LINES.

0234C5  MOV  #177670,RO  ; PASS THE LPR CONTENTS.

0234C8  JSR  PC,WTLPL  ; INITIALISE THE LPR REGISTERS ON ALL LINES.

0234CB  JSR  PC,TKENG  ; ENABLE TRANSMITTERS ON ALL LINES.

0234CD  ; SET UP OUTER LOOP FOR TESTING ACTIVE LINES IN BOTH LINE GROUPS.

0234D0  MOV  #1000000,R3  ; SET-UP LOOP CONTROL FLAG.

0234D3  MOV  ACTLNS,R5  ; GET THE ACTIVE LINE BIT MAP.
```assembly
BIC LGRP2M1, R5 ; REMOVE LINES IN GROUP 2.
MOV R5, R5+45 ; SAVE THE CURRENT LINE GROUP.
CLR R4+1 ; CLEAR THE LINE NUMBER COUNTER.
MOVS &R0, R1+000241 ; COPY THE LINE NUMBER.
CLC ; CLEAR CARRY BIT PRIOR TO SHIFTING BIT MAP.
ROR R5 ; SHIFT ACTIVELY LINE BIT MAP INTO CARRY BIT.
BCC 81 ; SKIP TESTING THIS LINE IF IT IS INACTIVE.

* TEST THE STATE OF THE DAUTO BIT ON THE LINE UNDER TEST.*
* REPORT ERROR IF IT IS FOUND CLEAR, AND SKIP FURTHER TESTING OF THAT LINE.

MOV @5002, ERRNBR ; SET THE ERROR NUMBER TO #5002.
MOV R1, BCSRA ; SELECT THE LINE TO BE TESTED.
BIT @BIT4,1,000241 ; TEST THE STATE OF THE DAUTO BIT.
BNE 61 ; SKIP ERROR REPORT IF DAUTO BIT IS SET.
MOV #E84002, R2 ; PASS THE ERROR MESSAGE.

"DAUTO BIT BAD ON LINE MN" ; ERROR

>>> ERROR #5002 <<<< ; TRAP CERROR

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06, OPTION ; EXIT WITH TEST FAILURE MESSAGE IF
BEQ 601 ; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
DURING THE SOFTWARE QUESTIONS.

BR 81 ; SKIP FURTHER TESTING OF THIS LINE.

TRANSMIT THE XOFF (ASCII DC3) ON THE ASSOCIATED LINE.

MOV #23, TDATA ; TRANSMIT THE XOFF CHARACTER TO THE LUT.

WAIT FOR TRANSMISSION TO COMPLETE.

INC ERRNBR ; INCREMENT ERROR NUMBER TO #5003.
MOV #17001, R1 ; TEST BIT IS, TIMEOUT OF 10 MILLI SECS.
MOVS &R0, R1+05777 ; PASS THE ADDRESS OF THE REGISTER TO TEST.
JSR PC, WAIT100005 ; WAIT FOR TRANSMISSION TO COMPLETE.
BCC 501 ; ABORT TEST IF TIMEOUT OCCURRED.
MOV #R4, R4 ; PASS TIME-OUT OF 5 MILLI SECS.
JSR PC, DELAY ; WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.

* TEST THE STATE OF THE TX_ENABLE BIT ON THE LINE UNDER TEST.*
* REPORT ERROR IF TX_ENABLE BIT IS SET.

INC ERRNBR ; INCREMENT ERROR NUMBER TO #5004.
MOV #170001, R1 ; GET THE ADDRESS OF THE LINE TEST.
MOVS &R0, R1+05777 ; SELECT THE LINE CURRENTLY UNDER TEST.
TST @IXAD2A ; TEST THE STATE OF THE TX_ENABLE BIT.
BPL 81 ; SKIP ERROR REPORT IF BIT IS CLEAR.
MOV #E84002, R2 ; PASS THE MESSAGE TO BE REPORTED.

"DAUTO BIT BAD ON LINE MN" ; ERROR

>>> ERROR #5004 <<<< ; TRAP CERROR
```
EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06, OPTION

EXIT WITH TEST FAILURE MESSAGE IF

NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

DURING THE SOFTWARE QUESTIONS.

INC 401

INCREMENT THE LINE NUMBER.

TST R5

CHECK IF THERE ARE ANY MORE LINES TO TEST.

ANE 41

DISABLE TRANSMITTERS ON THE SELECTED LINES IN THE CURRENT LINE GROUP.

MOV 451, R5

RESTORE THE CURRENT LINE ACTIVE LINE GROUP.

JSR PC, TXDSBL

DISABLE TRANSMITTERS ON THE SELECTED LINES.

MOV 451, R5

GET THE CURRENT LINE ACTIVE LINE GROUP AGAIN.

CLR 401

CLEAR THE LINE COUNTER.

MOV #5005, ERRMBR

SET ERROR NUMBER TO 5005.

MOV 401, R1

COPY THE LINE NUMBER.

CLC

CLEAR CARRY BIT PRIOR TO SHIFITNG BIT MAP.

ROR R5

SHIFT ACTIVE LINE BIT MAP INTO CARRY BIT.

BCC 121

SKIP TESTING THIS LINE IF IT IS INACTIVE.

TRANSMIT THE XON (ASCII D1) ON THE ASSOCIATED LINE.

MOVB TXRN+B, R1, BCSRA

SELECT THE ASSOCIATED TX LINE.

MOV #21, BFDATA

TRANSMIT THE XON CHARACTER TO THE LUT.

WAIT FOR TRANSMISSION TO COMPLETE.

MOV #170012, R1

TEST BIT 15, TIMEOUT OF 10 MILLI SECS.

MOV CSRA, R2

PASS THE ADDRESS OF THE REGISTER TO TEST.

JSR PC, WBIBS

WAIT FOR DMA TO COMPLETE.

BCC 501

ABORT TEST IF TIMEOUT OCCURRED.

MOV #5, R4

PASS TIME-OUT OF 5 MILLI SECS.

JSR PC, DELAY

WAIT FOR CHAR TO BE RECEIVED AND PROCESSED.

TEST THE STATE OF THE TX_ENABLE BIT ON THE LINE UNDER TEST.

REPORT ERROR IF TX_ENABLE BIT IS CLEAR.

INC ERRMBR

INCREMENT ERROR NUMBER TO 5006.

MOV R1, BCRA

SELECT THE LINE CURRENTLY UNDER TEST.

TST #TXD02A

TEST THE STATE OF THE TX_ENABLE BIT.

BHI 121

SKIP ERROR REPORT IF BIT IS SET.

MOV #EMA002, R2

PASS THE MESSAGE TO BE REPORTED.

"DATUO BIT BAD ON LINE #nn"

ERROR

TRAP CERROR

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06, OPTION

EXIT WITH TEST FAILURE MESSAGE IF

NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

DURING THE SOFTWARE QUESTIONS.
5729 024214 0005267 000036
5730 024230 0005705
5731 024222 0001323
5732
5733 ; CHECK LOOP CONTROL FLAG TO DETERMINE IF BOTH SETS OF LINES HAVE BEEN TESTED
5734 ; IF THIS IS THE FIRST TIME AROUND, RE-ENABLE TX ON ALL LINES, GENERATE ACTIVE
5735 ; BIT MAP FOR SECOND LINE GROUP.
5736
5737 024224 0005703
5738 024226 0001417
5739 024230 0005003
5740 024232 012705 177777
5741 024236 004767 172660
5742 024242 016705 155752
5743 024246 046705 156006
5744 024252 000167 177372
5745
5746 024256 000000
5747 024260 000000
5748 024262 004767 172404
5749 024266 0005067 155756
5750 024272
5751 024272
5752 024272 104401

INC 40
TST R5
BNE 10

; INCREMENT THE LINE NUMBER.
; CHECK IF THERE ARE ANY MORE LINES TO TEST.

L10033:
TRAP CIETST
5753 .SBITL HARDWARE TEST - IAUTO -
5755 - IAUTO BIT INACTIVE TEST
5756
5757
5758" THIS TEST VERIFIES THAT THE DUT'S IAUTO FUNCTION BEHAVES CORRECTLY
5759 WHEN INACTIVE, I.E. IAUTO BIT CLEAR.
5760 ALL ACTIVE LINES ARE TESTED INDIVIDUALLY BY FILLING THE FIFO
5761 THEN READING THE RECEIVED DATA CHECKING FOR THE PRESENCE OF
5762 XOFF(ASCII DC3) OR XON (ASCII DC1) CHARACTERS.
5763 IF ANY ARE FOUND THEN APPROPRIATE ERRORS ARE REPORTED.
5764 ANY BMP CODES THAT ARE FOUND WILL BE PLACED ON THE BMP CODE QUEUE.
5765 TO BE REPORTED LATER.
5766 THE CHARACTERS ARE TRANSMITTED ON ALL ACTIVE LINES, IN INTERNAL
5767 LOOPBACK MODE.
5768
5769
5770 8GNTS!
5771 SETPRI @PRI05 ;ALLOW LTC INTERUPTS.
5772
5773 MOV @PRI05,RC
5774 TRAP C7'PRI
5775
5776 TNUM = TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
5777 MOV @TNUM,TNUM ;SET UP THE TEST NUMBER.
5778 MOV @1,CTRCLF ;INDICATE THAT WE ARE IN A TEST.
5779 MOV @1,ERRORTP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
5780 MOV @5101,E字第 ;SET ERROR NUMBER TO 5101.
5781 MOV @MS101,ERRORSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
5782 MOV @ER9101,ERRORBK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
5783
5784 RESIT THE DUT TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO.
5785 CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.
5786 THIS SUBROUTINE REPORTS ERROR >>>> 5101 <<<<.
5787
5788 JSR PC,CLRNTS ;RESET THE DMU-11, REPORT ANY ERRORS FOUND.
5789 BCC 601 ;EXIT TEST IF FATAL ERROR FOUND.
5790
5791 INITIALIZE THE 256 BYTE DATA PATTERN.
5792 ENSURE THE DATA PATTERN IS FREE FROM XON'S OR XOFF'S TO PREVENT ERRORS.
5793 NOTE: THE FIRST TWO CHARACTERS AND THE LAST TWO CHARACTERS WILL BE THE SAME.
5794 JSR PC,INITDPX ;INITIALISE DATA PATTERN.
5795
5796 SET INTERNAL LOOPBACK, DISABLE IAUTO, ENABLE RECEIVER ON THE SELECTED LINE.
5797 SET LPR TO 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY, 2 STOP BITS.
5798 MOV #ACTLRG,AS ;PASS THE ACTIVE LINE BIT MAP.
5799 MOV #204,RO ;PASS INT'L LOOP, ENABLE RX, DISABLE IAUTO.
5800 JSR PC,WTMNC ;INITIALISE THE LINE CONTROL REGISTER.
5801 MOV @177670,RO ;PASS THE LPR CONTENTS.
5802 JSR PC,WTMLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
5803 MOV @10,R4 ;PASS DELAY TIME OF 10 MILLLI SECONDS.
5804 JSR PC,DELAY ;WAIT FOR LNTCL AND LPR REGS TO BE UPDATED.
5805
5806
SET UP LOOP FOR ALL ACTIVE LINES.

TEST THE STATE OF THE IAUTO BIT PRIOR TO TRANSMITTING THE DATA PATTERN.

IF THE BIT IS SET, THEN REPORT THE ERROR AND SKIP TRANSMITTING

THE DATA PATTERN ON THE SELECTED LINE.

TRANSPORT A 256 CHARACTER DATA PATTERN USING DMA, ON A SINGLE CHANNEL

EMPTY THE FIFO, AND VERIFY NO XOFF OR XON CHARMS WERE FOUND.

CLR R1, CLEAR THE LINE NUMBER COUNTER.

CLR R5, CLEAR STORAGE FOR LINE NUMBER.

MOV @5102, ERRNBR, SET THE ERROR NUMBER TO 5102.

JSR PC, PUFIFO, PURGE THE FIFO.

BCC 50$, GO REPORT ERROR IF FIFO DID NOT PURGE.

CLC, CLEAR CARRY PRIOR TO ROTATING BIT MAP.

ROR R5, ROTATE THE BIT MAP INTO THE CARRY BIT.

BCC 12$, BRANCH IF LINE IS INACTIVE.

TEST THE IAUTO BIT ON THE SELECTED ACTIVE LINE.

REPORT ERROR IF IT IS SET.

DO NOT TRANSMIT THE DATA PATTERN ON THE SELECTED LINE.

INC ERRNBR, SET ERROR NUMBER TO 5103.

MOV R1, BCXRA, SELECT LINE TO TEST.

BIT @BIT1, BLNCTRA, TEST THE STATE OF THE IAUTO BIT ON THIS LINE.

BEQ 44, SKIP ERROR IF IAUTO BIT CLEAR

MOV #5102, R2, PASS THE CORRECT ERROR MESSAGE.

ERROR, "\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\n\nEXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT @BIT06, OPTION, EXIT WITH TEST FAILURE MESSAGE IF

BEQ 60$, NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

DURING THE SOFTWARE QUESTIONS.

BR 12$, SKIP TRANSMITTING DATA PATTERN.

TRANSMIT DATA PATTERN OF 256 CHAR.

INC ERRNBR, SET ERROR NUMBER TO 5104.

MOV #UBF08, R2, PASS THE START OF THE DATA PATTERN TO TX.

MOV #256, R3, PASS THE LENGTH OF THE DATA PATTERN.

JSR PC, DDMMA, TRANSMIT THE DATA PATTERN.

BCC 50$, ABORT THE TEST IF ERROR FOUND DURING DMA TX.

WAIT FOR DMA TO COMPLETE, THEN WAIT FOR THE LAST CHARACTER PLUS XOFF

TO ARRIVE IN THE FIFO.

INC ERRNBR, SET ERROR NUMBER TO 5105.

MOV #170536, R1, PASS TIME-OUT VALUE OF 350 MILLI SECS.

MOV CSRA, R2, PASS THE ADDRESS OF THE CSR.

JSR PC, WABIS, WAIT FOR DMA TO COMPLETE, TX.ACTION SET.

BCC 50$, IF NO TX.ACTION WAS RECEIVED, ABORT THE TEST.

MOV 810, R4, PASS DELAY OF 10 MILLI SECS.

JSR PC, DELAY, WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
READ 256 CHARs FROM THE FIFO. REPORT ERROR IF ANY XOFF's OR XON's ARE FOUND.

INC ERNRBR
INC ERNRBR
MOV #0256,R1
INITIALISE THE READ COUNTER.

MOV @RBUFFA,R2
READ CHAR FROM THE FIFO.

BPL 50:
GO REPORT ERROR IF FIFO EMPTY.

CHECK FOR BMP CODE IN THE FIFO. SAVE ANY FOUND ON THE QUEUE.

MOV @170301,R0
SET UP BMP BIT MASK.

BIC R2,R0
TRY TO CLEAR ALL THE BMP BITS.

BNE 8:
SKIP BMPSAV IF NOT A BMP CODE.

JSR PC,SAVBMP
SAVE THE BMP CODE ON THE QUEUE.

CHECK FOR XOFF AND XON CHARACTERS.

CMPB R2,#23
IS IT AN XOFF CHARACTER?

BEQ 10:
YES; GO REPORT ERROR.

CMPB R2,#01
NO; IS IT AN XON CHARACTER?

BEQ 10:
YES; GO REPORT ERROR.

DEC R1
DECREMENT THE READ COUNT.

BNE 61:
LOOP TO READ THE NEXT CHAR.

BR 12:
GO CHECK FOR ANY UNTESTED ACTIVE LINES.

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED.

BIT #BIT06,OPTION
EXIT WITH TEST FAILURE MESSAGE IF NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED DURING THE SOFTWARE QUESTIONS.

CHECK IF ALL ACTIVE LINES HAVE BEEN TESTED.

INC 55:
INCREMENT LINE NUMBER.

MOV 551,R1
GET NUMBER OF THE NEXT LINE TO TEST.

TST R5
ARE THERE ANY MORE ACTIVE LINES TO TEST?

BNE 21:
LOOP TO CHECK NEXT LINE.

BR 60:
EXIT TEST.

REPORT TEST ABORTED. NON-TEST RELATED ERROR.

EXIT THIS TEST.

STORAGE FOR LINE NUMBER.

INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDSTST

L10034:
TRAP CIESTST
BGN6T

SETPRI  #PRIOS   ; ALLOW LTC INTERRUPTS.
MOVE  #PRIOS,RO
TRAP  C4SPRI

INUM = INUM + 1  ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.

MOV  #INUM,TSTNUM  ; SET UP THE TEST NUMBER. (52)

MOV  #01,CTRLCF  ; INDICATE THAT WE ARE IN A TEST.

MOV  #01,ERRTP  ; SET ERROR TYPE AS FATAL IN ERROR TABLE.

MOV  #5201,ERRRND  ; SET ERROR NUMBER TO 5201.

MOV  #5201,ERRRMSG  ; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

MOV  #5201,ERRRLBK  ; SELECT THE CORRECT ERROR REPORTING ROUTINE.

; RESET THE OUT 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 >>>>>> 5201 <<<<..

JSR  PC,CLNRST  ; RESET THE DMU-11, REPORT ANY ERRORS FOUND.

BNS  6

JMP  60  ; EXIT TEST IF FATAL ERROR FOUND.

; 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  ; INITIALIZE DATA PATTERN.

JSR  PC,INDTPX  ; INITIALIZE DATA PATTERN.

SET INTERNAL LOOPBACK, ENABLE IAUTO AND RECEIVER ON THE SELECTED LINE.

SET LPR TO 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY, 2 STOP BITS.

MOV  ACTLNS,R5  ; PASS THE ACTIVE LINE BIT MAP.

MOV  #0206,RO  ; PASS INTERNAL LOOPBACK, ENABLE RX AND IAUTO.

JSR PC,WTNLNC  ; INITIALIZE THE LINE CONTROL REGISTER.

MOV  #177670,RO  ; PASS THE LPR CONTENTS.

JSR  PC,WTLCPR  ; SET THE LPR CONTENTS TO 38.4K BAUD.

JSR  PC,DELAY  ; WAIT FOR LCTRL AND LPR REGS TO BE UPDATED.

SET UP LOOP FOR ALL ACTIVE LINES.
; TEST THE STATE OF THE OUTBIT PRIOR TO TRANSMITTING THE DATA PATTERN.
; IF THE BIT IS CLEAR, THEN REPORT THE ERROR AND SKIP TRANSMITTING
; THE DATA PATTERN ON THE SELECTED LINE.
; TRANSMIT A 224 CHARACTER DATA PATTERN USING DMA, ON A SINGLE CHANNEL
; EMPTY THE FIFO, AND COUNT THE XOFF AND AN XON CHARS FOUND.
; TEST THE IAUTO BIT ON THE SELECTED ACTIVE LINE.
; REPORT ERROR IF IT IS CLEAR.
; DO NOT TRANSMIT THE DATA PATTERN ON THE SELECTED LINE.
; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
; TRANSMIT DATA PATTERN TO FILL THE FIFO, 223 CHAR'S * 32 XOFF'S * XON.
; WAIT FOR DMA TO COMPLETE. THEN WAIT FOR THE LAST CHARACTER PLUS XOFF
; TO ARRIVE IN THE FIFO.

; CLR R1 ; CLEAR THE LINE NUMBER COUNTER.
; CLR 55 ; CLEAR STORAGE FOR LINE NUMBER.
; MOV #5202, ERRNBR ; SET THE ERROR NUMBER TO 5202.
; JSR PC,PURGE ; PURGE THE FIFO.
; BCC 501 ; GO REPORT ERROR IF FIFO DID NOT PURGE.
; CLC ; CLEAR CARRY PRIOR TO ROTATING BIT MAP.
; ROR R5 ; ROTATE THE BIT MAP INTO THE CARRY BIT.
; BCC 16# ; BRANCH IF LINE IS INACTIVE.

; INC ERRNBR ; SET ERROR NUMBER TO 5203.
; MOV R1,BCSRA ; SELECT LINE TO TEST.
; BIT #BITI, BLNCTRA ; TEST THE STATE OF THE IAUTO BIT ON THIS LINE.
; BNE 4# ; SKIP ERROR IF IAUTO BIT SET.
; MOV #EMS202,R2 ; PASS THE CORRECT ERROR MESSAGE.
; "IAUTO BIT FOUND CLEAR ON LINE NN"
; TRAP CERROR ; >>>>> ERROR <<<<<

; BIT #BITI06,OPTION ; EXIT WITH TEST FAILURE MESSAGE IF
; BEQ 60# ; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
; ; DURING THE SOFTWARE QUESTIONS.
; BR 16# ; SKIP TRANSMITTING DATA PATTERN.

; INC ERRNBR ; SET ERROR NUMBER TO 5204.
; MOV #BUTFAS,R2 ; PASS THE START OF THE DATA PATTERN TO TX.
; MOV #223,R3 ; PASS THE LENGTH OF THE DATA PATTERN.
; JSR PC,000MA ; TRANSMIT THE DATA PATTERN.
; BCC 50# ; ABRORT THE TEST IF ERROR FOUND DURING DMA TX.

; WAIT FOR DMA TO COMPLETE. THEN WAIT FOR THE LAST CHARACTER PLUS XOFF
; TO ARRIVE IN THE FIFO.

; INC ERRNBR ; SET ERROR NUMBER TO 5205.
; MOV #170454,R1 ; PASS TIME-OUT VALUE OF 300 MILLI SECS.
; MOV #170454,R1 ; PASS TIME-OUT VALUE OF 300 MILLI SECS.
; MOV CSRA,R2 ; PASS THE ADDRESS OF THE CSR.
; JSR PWAITIS ; WAIT FOR DMA TO COMPLETE. TX ACTION SET.
; BCC 50# ; IF NO TX ACTION WAS RECEIVED, ABRORT THE TEST.
; MOV #10,R4 ; PASS DELAY OF 10 MILLI SECS.
; JSR PC,DELAY ; WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
I READ 256 CHARs FROM THE FIFO. COUNT ANY XOFF OR XON CHARs FOUND.

I CLR R3 ; CLEAR XOFF COUNTER.
I CLR R4 ; CLEAR XON COUNTER.
I INC ERRBR ; INCREMENT ERROR NUMBER TO 5206.
I MOV #256, R1 ; INITIALIZE THE READ COUNTER.
I MOV BRBUF, R2 ; READ CHAR FROM THE FIFO.
I BPL 501 ; GO REPORT ERROR IF FIFO EMPTY.

I CHECK FOR BMP CODE IN THE FIFO. SAVE ANY FOUND ON THE QUEUE.

I MOV #170301, R0 ; SET UP BMP BIT MASK.
I BIC R2, R0 ; TRY TO CLEAR ALL THE BMP BITS.
I BNE 86 ; SKIP BMP BY IF NOT A BMP CODE.
I JSR PC, SAVBMP ; SAVE THE BMP CODE ON THE QUEUE.

I CHECK FOR XOFF AND XON CHARACTERS.

I CMPB R2, #023 ; IS IT AN XOFF CHARACTER?
I BNE 101 ; NO, BRANCH TO SEE IF IT IS AN XON.
I INC R3 ; COUNT THE XOFF CHAR.
I INC R4 ; COUNT THE XON.
I INC R5 ; COUNT THE XOFF CHAR.
I BNE 120 ; DECREMENT THE READ COUNT.
I BRG 131 ; BRANCH IF ALL CHARACTERS READ.

I CHECK IF THE FIFO HAS BEEN EMPTIED BELOW THE HALF LEVEL, IF IT
I HAS DELAY FOR IMS TO ALLOW THE XON TO BE GENERATED.

I CMP R1, #126. ; IS THE FIFO LEVEL = 126?
I BNE 61 ; LOOP TO READ THE NEXT CHARACTER IF NOT.
I MOV R4, R0 ; SAVE THE XON COUNT, ALTHOUGH THERE SHOULDN'T
I BE ANY.
I MOV R1, R4 ; SET THE DELAY TO IMS.
I JSR PC, DELAY ; PERFORM THE DELAY.
I MOV R0, R4 ; RESTORE THE XON COUNT.
I BR 61 ; LOOP TO READ THE NEXT CHAR.

I VERIFY THAT AT LEAST 1 XOFF AND 1 XON WAS FOUND IN THE FIFO.
I REPORT ERROR IF NONE WERE FOUND.

I TST R3 ; CHECK XOFF COUNT.
I BNE 141 ; GO REPORT ERROR IF NONE FOUND.
I CMP R4, #1 ; CHECK XON COUNT = 1.
I BNE 161 ; SKIP THE ERROR REPORT IF ONE XON WAS FOUND.
I INC ERRBR ; SET ERROR NUMBER TO 5207.
I MOV #553, R1 ; PASS THE LINE NUMBER TO BE REPORTED.
I MOV #EMS105, R2 ; PASS THE ERROR MESSAGE TO BE REPORTED.
I "IAUTO BIT BAD ON LINE "N.
I >>>>>> ERROR "<<<.
I TRAP CERROR

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED.
6083 6084 025344 032767 000100 154636 BIT BIT0C.OPTION EXIT WITH TEST FAILURE MESSAGE IF
6085 025352 001413 BEQ 601 NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
6086 DURING THE SOFTWARE QUESTIONS.
6087 "CHECK IF ALL ACTIVE LINES HAVE BEEN TESTED.
6088 6089
6090 025354 005267 000020 161: INC 551 INCREMENT LINE NUMBER.
6091 025360 016701 000014 MOV 551,R1 GET NUMBER OF THE NEXT LINE TO TEST.
6092 025364 005705 TST R5 ARE THERE ANY MORE ACTIVE LINES TO TEST?.
6093 025366 001230 BNE 21 LOOP TO CHECK NEXT LINE.
6094 025370 000404 BR 601 EXIT TEST.
6095 6096 025372 004767 171274 501: JSR PC,T5ABRT REPORT TEST ABORTED, NON-TEST RELATED ERROR.
6097 025376 000401 BR 601 EXIT THIS TEST.
6098 025400 000000 551: 'WORD 0 STORAGE FOR LINE NUMBER.
6099 025402 005067 154642 601: 'CLR CTRL.CF INDICATE THAT WE ARE NOT WITHIN A TEST.
6100 6101 025406 ENDS1
025406 025406 104401 L10035; TRAP CHTEST
THIS TEST VERIFIES THAT THE DUT IS CAPABLE OF HOLDING 256 VALID CHARACTERS IN ITS FIFO. THE CHARACTERS ARE TRANSMITTED ON THE FIRST AVAILABLE ACTIVE LINE, IN INTERNAL LOOPBACK MODE. THE DATA FOUND IN THE FIFO IS COMPARED WITH THE EXPECTED DATA, AND ANY DISCREPANCIES ARE REPORTED. 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.

BGNST

SETPRI @PRIO5

MOV @PRIO5,RO

BUD*: TRAP CISPRI

BUD

T9:

INCR NUM,* TNUM, I: INCREASE THE ASSEMBLY TIME TEST COUNTER.

MOV #1,CTRLCF: INDICATE THAT WE ARE IN A TEST.

MOV #1,ERRTPY: SET ERROR TYPE AS FATAL IN ERROR TABLE.

MOV #5301,ERRMGR: SET ERROR NUMBER TO 5301.

MOV #EH5301,ERRMSG: SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

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 5301, 5301.

JSR PC,CLNRST: RESET THE DMU-11, REPORT ANY ERRORS FOUND.

BCC 601: EXIT TEST IF FATAL ERROR FOUND.

JSR PC,FNACT: FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.

INITIALISE 256 BYTE DATA PATTERN.

JSR PC,INDAT: INITIALISE THE DATA PATTERN.

TRANSmits A 256 CHARACTER DATA PATTERN USING DMA, ON A SINGLE CHANNEL.

AT 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY, 2 STOP BITS.

JSR PC,HLEEP: SET INTERNAL LOOPBACK ON THE SELECTED LINE.

JSR PC,ADAT: TRANSMIT THE DATA PATTERN ON THE FIRST AVAILABLE ACTIVE LINE.

MOV #204,RO: PASS PARAMETER FOR INTERNAL LOPBACK ENABLE RX.

JSR PC,WTWLC: INITIALISE THE LINE CONTROL REGISTER.

MOV #177670,RO: PASS THE LPR CONTENTS.

JSR PC,WTWPR: SET THE LPR CONTENTS TO 38.4K BAUD.

MOV #10.4: PASS DELAY TIME OF 10 MILLI SECONDS.

JSR PC,DEL: HINT FOR LCTRL AND LPR REGS TO BE UPDATED.

MOV #255,RO: PASS THE START OF THE DATA PATTERN TO TX.

MOV #255,RO: PASS THE LENGTH OF THE DATA PATTERN.
; SET ERROR NUMBER TO 5302.
INC ERRNBR

; TRANSMIT THE DATA PATTERN.
JSR PC,DOOMA

; AVOID TEST IF ERROR FOUND DURING DMA TX.
BCC 501

; WAIT FOR DMA TO COMPLETE, THEN WAIT FOR THE LAST CHARACTER TO ARRIVE IN
; THE FIFO.

; SET ERROR NUMBER TO 5303.
INC ERRNBR

; SAVE THE NUMBER OF THE SELECTED ACTIVE LINE.
MOV R1,R3

; PASS TIME-OUT VALUE OF 350 MILLI SECS.
MOV @170536,R1

; PASS THE ADDRESS OF THE CSR.
MOV CSRA,R2

; WAIT FOR DMA TO COMPLETE, TxAction Set.
JSR PC,WAITIS

; BRANCH IF FIFO EMPTY, AVOID THE TEST.
BCC 501

; PASS DELAY OF 5 MILLI SECS.
MOV $5,R4

; WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
JSR PC,DELAY

; READ THE FIFO CHECKING FOR DATA CORRUPTION, REPORT ANY ERRORS FOUND.

; AVOID THE TEST IF A BMP CODE WAS FOUND IN THE FIFO.

; MULTIPLE BY 2.
ASL R3

; INITIALISE THE EXPECTED DATA.

; GET THE ADDRESS OF THE RECEIVER BUFFER REG.
MOV RBUFA,R5

; SET UP ERROR NUMBER EACH TIME AROUND LOOP.
MOV $5304,ERRNBR

; GET THE ACTUAL DATA FROM THE FIFO.
MOV (R5),R2

; AVOID THE TEST IF THE FIFO IS EMPTY.
BPL 501

; CHECK IF THE READ CHARACTER IS A BMP CODE.

; IF IT IS A BMP CODE SAVE IT ON THE QUEUE TO BE REPORTED LATER, AND
; AVOID THE TEST.

; SET ERROR NUMBER TO 5305.
INC ERRNBR

; CHECK IF CHARACTER IS A BMP CODE.
JSR PC,CHKBMP

; BRANCH IF NOT A BMP CODE.
BCC 41

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.

; REPORT THE ERROR "FIFO BAD, DATA FIELD CORRUPTED"

; AVOID THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
BIF

; BIT #BIT06.OPTION
BIT #BIT06.OPTION

; EXIT WITH TEST FAILURE MESSAGE IF
BED 601

; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED

; DURING THE SOFTWARE QUESTIONS.
025704 004767 170762 503: JSR PC,TSA ABORT THE TEST, REASON SHOWN BY ERROR NUMBER.
025710 005067 154334 604: CLR CTL CFC
; INDICATE THAT WE ARE NOT WITHIN A TEST.
025714 END TST
025714 104401

L10036: TRAP C4 TST
6218 6219 6220 6221 6222 6223 6224 6225 6226 6227 6228 6229 6230 6231 6232 6233 6234 6235 025716 025716 012700 000240 025722 104441 6236 6237 025724 012767 000012 154322 6238 025732 12767 17777 154310 6239 025740 025767 000001 156102 6240 025746 012767 012431 156076 6241 025754 012767 046452 156072 6242 025762 012767 011762 156066 6243 6244 6245 6246 6247 6248 025770 004767 166222 6249 025774 103111 6250 6251 6252 6253 025776 004767 166466 6254 026002 103106 6255 6256 6257 6258 6259 6260 6261 026004 004767 166570 6262 6263 6264 6265 6266 6267 026010 012700 000206 6268 6269 6270 026014 004767 171532

SBITL HARDWARE TEST ---- FISQLI ----

; 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.

BGNSTS

SETPRI #PRIOR  ; ALLOW LTC INTERRUPTS.

TRAP C'&SPR.

T10: ;

MOV #PRIOR,RO

.INCREMENT THE ASSEMBLY TIME TEST COUNTER.

MOV #TNUM,TSTNUM  ; SET UP THE TEST NUMBER.

INC TNUM  ; (54)

MOV #1,CTRLCF  ; INDICATE THAT WE ARE IN A TEST.

MOV #1,EERNTYP  ; SET FATAL ERROR TYPE IN ERROR TABLE.

MOV #401,EERNBR  ; SET ERROR NUMBER TO 5401.

MOV #EM5401,EERRMSG  ; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

MOV @EER503,EERBLK  ; SELECT THE CORRECT ERROR REPORTING ROUTINE.

; RESET THE OUT 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 FAILED.

JSR PC,CLNRM

; RESET THE DUT-11. 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 30.4K BAUD, 8 BITS PER CHARACTER, ODC 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,RO  ; PASS INTERNAL LOOPBK, ENABLE RX AND IAUTO.

JSR PC,WTWNC  ; INITIALISE THE LINE CONTROL REGISTER.
MOV #177670,R0 ;PASS THE LPR CONTENTS
JSR PC,WTLPR ;SET THE LPR CONTENTS TO 38.4K BAUD.
MOV #10,R4 ;PASS DELAY TIME OF 10 MILI SECONDS.
JSR PC,DELAY ;WAIT FOR LNRCTR AND LPR REGS TO BE UPDATED.
MOV #BUBFAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
MOV #191,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
JSR PC,DOOMA ;TRANSMIT THE DATA PATTERN.
BCC 504 ;IF ERROR FOUND DURING DMA THEN ABORT TEST.

;WAIT FOR DMA TO COMPLETE, THEN WAIT FOR THE LAST CHARACTER TO ARRIVE IN
;THE FIFO.

INC ERRNBR ;SET ERROR NUMBER TO 5402.
MOV #170454,R1 ;PASS TIME-OUT VALUE OF 300 MILLI SECS.
MOV CSR,R2 ;PASS THE ADDRESS OF THE CSR.
JSR PC,WAIT1S ;WAIT FOR DMA TO COMPLETE, TX_ACTION SET.
BCC 504 ;IF FIFO EMPTY, REPORT ERROR, ABORT THE TEST.
MOV #5,R4 ;PASS DELAY OF 5 MILI SECS.
JSR PC,DELAY ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.

READ THE CONTENTS OF THE FIFO. IF ANY OF THE FOLLOWING CONDITIONS OCCUR
REPORT THE ERROR AND ABORT THE TEST;
FIFO EMPTY TOO SOON.
BMP CODE FOUND.
XOFF CODE FOUND.
EXTRA (192) CHARACTER FOUND IN FIFO.

CLR R4 ;CLEAR THE CHARACTER COUNT.
MOV RBUFA,R5 ;GET THE ADDRESS OF THE RECEIVER BUFFER REG.
MOV #3403,ERRNBR ;SET ERROR NUMBER TO 5403.
MOV (RS),R2 ;GET THE ACTUAL DATA FROM THE FIFO.
BPL 504 ;IF FIFO EMPTY, ABORT TEST.
INC R4 ;COUNT THE CHARACTER.

CHECK IF THE READ CHARACTER IS A BMP CODE.
IF IT IS A BMP CODE SAVE IT ON THE QUEUE TO BE REPORTED LATER, AND
ABORT THE TEST.

INC ERRNBR ;SET ERROR NUMBER TO 5404.
JR C,CHKBMP ;CHECK IF CHARACTER IS A BMP CODE.
BCC 41 ;BRANCH IF NOT A BMP CODE.
REPORT ERROR "BMP CODE FOUND IN FIFO, TEST INVALIGATED".
BR 81 ;REPORT THE ERROR AND ABORT THE TEST.

CHECK IF THE CHARACTER IS AN XOFF. REPORT THE ERROR IF ONE IS FOUND.

INC ERRNBR ;SET ERROR NUMBER TO 5405.
CMP R3,R2 ;CHECK IF THE READ DATA IS AN XOFF.
BNE 61 ;BRANCH IF NOT AN XOFF.
MOV #EMS402,R1 ;PASS THE MESSAGE TO BE REPORTED.
REPORT THE ERROR "FIFO BAD, ALARM SIGNAL DEFECTIVE".
BR 81 ;GO REPORT THE ERROR AND ABORT THE TEST.

INC ERRNBR ;SET ERROR NUMBER TO 5406.
CMP R4,R191 ;CHECK IF WE HAVE READ ALL THE CHARACTERS.
BNE 2$ ; LOOP BACK TO GET THE NEXT CHARACTER.

MOV (R5),R2 ; TRY TO READ AN EXTRA CHARACTER FROM THE FIFO.

BPL 601 ; EXIT IF NOT FOUND.

MOV #EMS402,R1 ; PASS THE MESSAGE TO BE REPORTED.

; REPORT THE ERROR "FIFO BAD, ALARM SIGNAL DEFECTIVE".

8$: ERROR ; >>>>>> ERRORS 5304 THRU 5306 <<<<<<.

BR 601 ; EXIT THE TEST.

50$: JSR PC,TSA8RT ; REPORT TEST ABOATED, NON-TEST RELATED ERROR.

60$: CLR CTRLCF ; INDICATE THAT WE ARE NOT WITHIN A TEST.
THIS TEST VERIFIES THAT THE DUT'S FIFO 3/4 LEVEL ALARM SYSTEM.
BECOMES ACTIVE WHEN THE FIFO CONTAINS > 192 CHARACTERS.
THE TEST COMPARES THE ACTUAL NUMBER OF XOFF (ASCII DC3)
CHARACTERS THAT ARE FOUND IN THE FIFO WITH THE EXPECTED NUMBER.
AN ERROR WILL BE REPORTED IF THE COUNTS ARE FOUND TO DIFFER.
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.

SETPRI,T11::

SETPRI @PRIONS
ALLOW LTC INTERRUPTS.
MOV @PRIONS, 0

TRAP C1SPRI

T11::

TNUM == TNUM + 1 ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV @TNUM, ISTNUM ; SET UP THE TEST NUMBER. (55)
MOV #5-1, CLRCH ; INDICATE THAT WE ARE IN A TEST.
MOV @1, ERRTP ; SET ERROR TYPE AS FATAL IN ERROR TABLE.
MOV @5501..ERRNBR ; SET ERROR NUMBER TO 5501.
MOV @EM5501..ERRMSG ; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

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 5501.

JSR PC.CLRNST ; RESET THE DHU-11. REPORT ANY ERRORS FOUND.
JMP 60 ; EXIT TEST FATAL ERROR FOUND.

JSR PC.FINACI ; FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.
JMP 60 ; EXIT TEST.

PC.CLNSM ; INITIALIZE THE 256 BYTE DATA PATTERN.
PC.CMDTP ; ENSURE THE DATA PATTERN IS FREE FROM XOFF'S OR XON'S TO PREVENT ERRORS.
(NOTE: THE FIRST TWO CHARACTERS AND THE LAST TWO CHARACTERS WILL BE THE SAME.)
JSR PC.INITP ; INITIALISE 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 RECEIVER ON THE SELECTED LINE.
TRANSMIT THE FIRST 191 CHARACTERS ON THE FIRST AVAILABLE ACTIVE LINE.
INC ERRNBR ; SET ERROR NUMBER TO 5502.
6398 026326 012700 000206
6399 026332 004767 171214
6400 026336 012700 177670
6401 026342 004767 171234
6402 026346 012704 000012
6403 026352 004767 165704
6404 026356 010105
6405 026360 012702 002650
6406 026364 012703 000277
6407 026370 004767 166004
6408 026374 103147

6410
6411
6412
6413
6414 026376 005267 155450
6415 026402 012701 170454
6416 026406 016702 153614
6417 026412 004767 171020
6418 026416 103136
6419 026420 012704 000005
6420 026424 004767 165632

6421
6422
6423
6424 026430 005267 155416
6425 026434 010501
6426 026436 012702 002650
6427 026440 004767 170012
6428 026444 004767 165726
6429 026452 103120

6430
6431
6432 026454 005267 155372
6434 026460 012701 170012
6435 026464 016702 153536
6436 026470 004767 170742
6437 026474 103107
6438 026476 012704 000005
6439 026502 004767 165554

6440
6441
6442 026506 012702 002650
6443 026512 105022
6444 026514 020227 003250
6446 026520 103774

6447
6448
6449
6450
6451 026522 005267 155324
6452 026526 010501
6453 026530 012702 002650

6454
6455 026534 012703 000037
6456 026540 004767 165634
6457 026544 103063
6458
6459 1
6460
6461
6462 026546 005267 155300
6463 026552 012701 170454
6464 026556 016702 153444
6465 026562 004767 170650
6466 026566 103052
6467 026570 012704 000005
6468 026574 004767 165462
6469
6470
6471
6472
6473 026600 005004
6474 026602 005003
6475 026604 012701 170001
6476 026610 012767 012604 155234 64: MOV #170001.R1
6477 026616 016702 153406
6478 026622 004767 170610
6479 026626 103032
6480 026630 005204
6481
6482
6483
6484 026632 005267 155214
6485 026636 004767 165134
6486 026642 103422
6487
6488
6489
6490 026644 122702 000023 68: CMPB #23,R2
6491 026650 001001
6492 026652 005203
6493
6494
6495
6496
6497 026654 020427 000400 10: CMP R4,#256
6498 026660 002753
6499
6500
6501
6502 026662 012767 012606 155162
6503
6504 026670 002703 000040
6505 026674 001411
6506 026676 012767 011762 155152
6507 026704 012701 006520
6508
6509
6510 026710 010440
BR 601:  ;ABORT THE TEST.

JSR PC,TSABRT  ;REPORT TEST ABORTED. ERROR 0 SHOWS REASON.

CLR CR,LCF  ;INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDST

L10040:  TRAP C#ETST
6518 .SBTTL HARDWARE TEST
6519 .FI3QAI
6520 
6521 ****************************
6522 | * - FIFO 3/4 ALARM LEVEL ACTIVE/INACTIVE TEST - *
6523 | *
6524 | * THIS TEST VERIFIES THAT THE DUT'S FIFO 3/4 LEVEL ALARM SYSTEM *
6525 | * BECOMES ACTIVE AND INACTIVE AT THE CORRECT LEVELS. *
6526 | * ANY BMP CODE FOUND WILL INVALIDATE THE TEST AND CAUSE IT TO BE ABORTED. *
6527 | * HOWEVER THE BMP CODE WILL BE PLACED ON THE BMP CODE QUEUE, TO BE *
6528 | * REPORTED LATER. *
6529 | * THE CHARACTERS ARE TRANSMITTED ON THE FIRST AVAILABLE ACTI
6530 | * VME LOOPBACK MODE.
6531 |
6532 BGNST
6533 026726
6534 026726
6535 026726 012700 000240
6536 026726 10441
6537 000014
6538 026734 012767 000014 153312
6539 026742 012767 17777 153300
6540 026750 012767 000001 155072
6541 026756 012767 012741 155066
6542 026764 012767 006625 155062
6543
6544 T12::
6545 SETPRI #PRI05 ; ALLOW LTC INTERRUPTS.
6546 MOV #PRI05,RO
6547 TRAP CISPRI
6548
6549 TNUM == TNUM + 1 ; INCREMENT THE ASSEMBLY TIME TEST COUNTER
6550 MOV #TNUM,ISTNUM ; SET UP THE TEST NUMBER.
6551 (56)
6552 MOV #1,CTRLCF ; INDICATE THAT WE ARE IN A TEST.
6553 MOV #1,ERRTP ; SET ERROR TYPE AS FATAL IN ERROR TABLE.
6554 MOV #5601,ERRNBR ; SET ERROR NUMBER TO 5601.
6555 MOV #EM5601,ERRMSG ; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
6556
6557 ; CLEAR TX AND RX INTERRUPT ENABLE Bits IN THE CSR.
6558 ; THIS SUBROUTINE REPORTS ERROR >>>> 5601 <<<<.
6559 ; -
6560 JSR PC,CNRSST ; RESET THE DUT TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO.
6561 ; - CLEAR TX AND RX INTERRUPT ENABLE Bits IN THE CSR.
6562 ; -
6563 ; -
6564 BCS 24 ; SKIPEXIT TEST IF SUCCESSFUL RESET.
6565 JMP 604 ; EXIT THIS TEST.
6566
6567 24:
6568 ; FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.
6569 ; -
6570 JSR PC,FNUMST ; FIND AN ACTIVE LINE.
6571 BCS *.6 ; SKIP EXIT OF TEST IF ACTIVE LINE FOUND.
6572 JMP 604 ; EXIT TEST.
6573 *.6
6574 ; INITIALIZE THE 256 BYTE DATA PATTERN.
6575 ; ENSURE THE DATA PATTERN IS FREE FROM XON'S OR XOFF'S TO PREVENT ERRORS.
6576 ; NOTE: THE FIRST TWO CHARACTERS AND THE LAST TWO CHARACTERS WILL BE THE SAME.
6577 ; -
6578 JSR PC,INDTPX ; INITIALISE THE DATA PATTERN.
6579 ; -
6580 TRANSMIT A 256 CHARACTER DATA PATTERN USING DMA, ON A SINGLE CHANNEL
6581 ; AT 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY, 2 STOP BITS.
6582 ; -
6583 ; -
6584 ; -
6585 ; -
6586 ; -
6587 ; -
6588 ; -
6589 INC ERRNBR ; SET ERROR NUMBER TO 5602.
6590 MOV #206,RO ; PASS INTERNAL LOOPBACK, ENABLE RX AND IAUTO.
6591 JSR PC,VTLINC ; INITIALISE THE LINE CONTROL REGISTER.
6572 027036 012700 177670
6573 027042 004767 170534
6574 027048 012702 002650
6575 027052 004767 165204
6576 027056 004767 170534
6577 027060 004767 165304
6578 027064 004767 165304
6579 027070 004767 165304
6580 027074 103146

MOV #177670,R0
PASS THE LPR CONTENTS.

MOV #170534,R0
PASS THE LPR CONTENTS TO 38.4K BAUD.

JSR PC,DELAY
SET THE LPR CONTENTS TO 38.4K BAUD.

MOV @01, R4
PASS DELAY TIME OF 10 MILLI SECONDS.

JSR PC,DELAY
WAIT FOR LCNTRL AND LPR REGS TO BE UPDATED.

MOV R1,R5
COPY THE LINE NUMBER.

MOV RBUFAS,R2
PASS THE START OF THE DATA PATTERN TO TX.

MOV @91,R3
PASS THE LENGTH OF THE DATA PATTERN.

JSR PC,DODMA
TRANSMIT THE DATA PATTERN.

BCC 501
EXIT IF ERROR FOUND DURING DMA TX.

6581

6582

6583

6584

INC ERRNR
WAIT FOR DMA TO COMPLETE, THEN WAIT FOR THE LAST CHARACTER TO ARRIVE IN THE FIFO.

MOV #170454,R1
SET ERROR NUMBER TO 5403.

MOV CSRA,R2
PASS TIME-OUT VALUE OF 300 MILLI SECS.

JSR PC,WAIBIS
WAIT FOR DMA TO COMPLETE, TX ACTION SET.

BCC 501
BRANCH IF FIFO EMPTY, ABORT THE TEST.

MOV @5,R4
PASS DELAY OF 5 MILLI SECS.

J.R PC,DELAY
WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.

6586

6587

INC ERRNR
TRANSMIT A NULL CHARACTER WHICH WILL CAUSE AN XOFF TO BE GENERATED.

MOV R5,R1
SET ERROR NUMBER TO 5604.

MOV RBUFAS,R2
PASS THE LINE NUMBER.

MOV #1,R3
PASS THE START OF THE DATA PATTERN TO TX.

MOV @1,R3
PASS THE NUMBER OF

JSR PC,DODMA
TX A NULL CHARACTER TO CAUSE AN XOFF.

BCC 501
ABORT THE TEST IF ERROR FOUND DURING DMA TX.

6589

6590

6591

6592

6593

6594

6595

6596

6597

INC ERRNR
WAIT FOR THE XOFF TO BE RECEIVED BEFORE CONTINUING THE TEST.

MOV #170012,R1
SET ERROR NUMBER TO 5605.

MOV CSRA,R2
PASS TIME-OUT VALUE OF 10 MILLI SECS.

JSR PC,WAIBIS
WAIT FOR DMA TO COMPLETE, TX ACTION SET.

BCC 501
IF NO TX ACTION WAS RECEIVED, ABORT THE TEST.

MOV @5,R4
PASS DELAY OF 5 MILLI SECS.

JSR PC,DELAY
WAIT FOR XOFF TO GET INTO THE FIFO.

MOV R5,CSRA
SELECT THE LINE READY FOR TRANSMISSION.

6405

6406

6407

6408

6409

6410

6411

6412

6413

6414

6415

6416

6417

6418

6419

6420

6421

6422

6423

6424

6425

6426

6427

6428

CLA R5
CLEAR THE TX FLAG.

CLA R3
CLEAR THE CHARACTER COUNTER.

MOV #192,R3
SET UP READ COUNTER FOR THE FIRST 192 CHAR.

MOV #170005,R1
SET READ COUNTER.

MOV RBUFAS,R2
INDICATE TO TEST DATA VALID BIT, TIME-OUT 5MS.

MOV RBUFAS,R2
INDICATE TO CHECK RECEIVE REGISTER.

JSR PC,WAIBIS
WAIT FOR RECEIVED CHAR OR TIME-OUT.

BCC 146
EXIT LOOP IF TIME OUT, FIFO EMPTY.
027244 005300 DEC R0 ;DECREMENT READ COUNTER.
027246 005303 DEC R3 ;DECREMENT CHAR COUNTER.
027248 003002 BGT 81 ;SKIP DISBL.C TX IF FIRST 192 CHARS NOT READ.
027250 052709 100000 BIS #BIT15,R5 ;DISABLE ANY FURTHER TRANSMISSIONS.
027252 012746 154566 81 MDV #5M608,ERRNR ;SET ERROR NUMBER EACH TIME AROUND LOOP.
027254 004767 164506 JSR PC,CHRBMP ;CHECK IF CHARACTER IS A BMP CODE.
027256 103466 BCS 161 ;GO REPORT ERROR AND ABORT TEST IF BMP FOUND.
027258 112777 000000 152714 MOV #0,BFDATA ;TX A NULL CHARACTER.
02725A 012702 000025 CMPR #23,R2 ;CHECK IF THE RECEIVED CHARACTER WAS AN XOFF.
02725C 010011 BNE 121 ;BRANCH IF CHARACTER WAS NOT AN XOFF.
02725E 052040 INC R4 ;INCREASE THE XOFF CHAR COUNT.
027260 005700 TST R0 ;CHECK READ COUNT. TO SEE IF A CHAR CAN BE TX.
027262 005705 BNE 61 ;BRANCH IF 3 CHAR HAVE NOT YET BEEN READ.
027264 010744 TST R5 ;CHECK THE TRANSMISSION ENABLED FLAG.
027266 011350 BMI 41 ;SKIP TRANSMITTING A CHARACTER IF TX DISABLED.
027268 112777 000000 152714 MOV #0,BFDATA ;TX A NULL CHARACTER.
02726A 010446 MOV R4,(-SP) ;SAVE THE XOFF COUNT ON THE STACK.
02726C 005267 154524 INC ERRNR ;SET ERROR NUMBER TO 5607.
02726E 012701 170012 MDV #170012.R1 ;PASS TIME-OUT VALUE OF 10 MILLI SECS.
027270 016702 152670 MOV CSRA,R2 ;PASS THE ADDRESS OF THE CSR.
027272 004767 170747 JSR PE,NAIBIS ;WAIT FOR DNA TO COMPLETE. TX.ANCTION SET.
027274 010205 BCC #501 ;IF NO TX.ANCTION WAS RECEIVED, ABORT THE TEST.
027276 000005 MOV #5,R4 ;PASS DELAY OF 5 MILLI SECS.
027278 012704 164706 JSR PC,DELA1 ;WAIT FOR XOFF TO GET INTO THE FIFO.
02727A 012604 MOV (#SP)+,R4 ;RESTORE THE XOFF COUNT.
02727C 000721 BR 41 ;GO RESET THE READ COUNT AND GET NEXT CHAR.

; CHECK IF THE CORRECT NUMBER OF XOFF'S WERE FOUND IN THE FIFO
02727E 012767 012750 154464 141 MDV #5M608,ERRNR ;SET ERROR NUMBER TO 5608.
027280 020427 000077 CMPR R4,#63. ;COMPARE THE EXPECTED AND ACTUAL XOFF COUNTS.
027282 011411 BEQ 604 ;EXIT TEST IF SUCCESS.
027284 012767 011762 154454 MOV #ER0503,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
027286 006520 MOV #ERS402.R1 ;PASS THE MESSAGE TO BE REPORTED.
027288 005402 R1 JSR PC,TAUR ;REPORT THE ERROR "FIFO BAD, ALARM SIGNAL DEFECTIVE".
02728A 004640 161 ERROR WS1 ERROR <---- TRAP C#ERROR
02728C 000402 BR 601 ;EXIT THIS TEST.
02728E 012712 004767 167254 501 JSR PC,TSABRT ;REPORT TEST ABORTED. ERROR # INDICATES FAULT.
027290 005067 152626 601 CLR CTRLF ;INDICATE "HAVE WE NOT WITHIN A TEST."
6686 \texttt{SBITT} \texttt{HARDWARE TEST} \texttt{FIMAVL} \texttt{FIFO HALF LEVEL ACTIVE/INACTIVE TEST}

\texttt{THIS TEST CHECKS THAT THE DUT'S FIFO HALF LEVEL ALARM SYSTEM BECOMES ACTIVE AND INACTIVE AT THE CORRECT LEVELS. 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.}

\begin{verbatim}
6700 \texttt{027424}
\texttt{027424} \texttt{027424} \texttt{027424} \texttt{012700} \texttt{000240} \texttt{027430} \texttt{104441}
\texttt{027432} \texttt{000015} \texttt{027432} \texttt{012767} \texttt{000015} \texttt{152614} \texttt{027440} \texttt{012767} \texttt{177777} \texttt{152602}
\texttt{027446} \texttt{012767} \texttt{000001} \texttt{154374} \texttt{027454} \texttt{012767} \texttt{013105} \texttt{154370}
\texttt{027462} \texttt{012767} \texttt{006702} \texttt{154364} \texttt{027470} \texttt{012767} \texttt{011762} \texttt{154360}
\texttt{027476} \texttt{004767} \texttt{164514} \texttt{027502} \texttt{103402} \texttt{027504} \texttt{000167} \texttt{000364}
\texttt{027510} \texttt{027510} \texttt{027510} \texttt{027510} \texttt{004767} \texttt{164754} \texttt{027514} \texttt{103167}
\texttt{027516} \texttt{000000} \texttt{027516} \texttt{004767} \texttt{165056} \texttt{027518} \texttt{004767} \texttt{165056}
\texttt{027522} \texttt{005267} \texttt{154324} \texttt{027526} \texttt{004767} \texttt{167014}
\texttt{BGNST}

\texttt{SETPRI} \texttt{PRI05} \texttt{ALLOW LTC INTERRUPTS.}
\texttt{MOV} \texttt{PRI05,RO}
\texttt{TRAP} \texttt{C1SPRI}
\texttt{INUM ++ INUM + 1 INC 1 THE ASSEMBLY TIME TEST COUNTER.}
\texttt{MOV} \texttt{INUM,TSTNUM} \texttt{SET UP THE TEST NUMBER.}
\texttt{MOV} \texttt{01,CTRLF} \texttt{INDICATE THAT WE ARE IN A TEST.}
\texttt{MOV} \texttt{#1,ERRTP} \texttt{SET ERROR TYPE AS FATAL IN ERROR TABLE.}
\texttt{MOV} \texttt{#5701,ERRBR} \texttt{SET ERROR NUMBER TO 5701.}
\texttt{MOV} \texttt{#5701,ERMMSG} \texttt{SET ERROR MESSAGE ADDRESS IN ERROR TABLE.}
\texttt{MOV} \texttt{#30503,ERLBLK} \texttt{SELECT THE ERROR REPORTING ROUTINE.}
\texttt{RESET THE DUT TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO.}
\texttt{CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.}
\texttt{THIS SUBROUTINE REPORTS ERROR >>5701<<.}
\texttt{JSR} \texttt{PC,CLRST} \texttt{RESET THE DUT, REPORT ANY ERRORS FOUND.}
\texttt{JSR} \texttt{PC,FINACT} \texttt{FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.}
\texttt{JSR} \texttt{PC,INTPA} \texttt{INITIALISE THE 256 BYTE DATA PATTERN.}
\texttt{FILL THE FIFO AND THE UART'S 3 UART BUFFER BY TRANSMITTING 225 CHARACTERS. (IE 225 + 3 XOFF'S). TRANSMIT A PATTERN USING DMA ON A SINGLE CHANNEL.}
\texttt{EXIT IF NO ACTIVE LINES AVAILABLE.}
\texttt{EXIT IF IF NO ACTIVE LINES AVAILABLE.}
\texttt{JSR} \texttt{PC,INDTPX} \texttt{INITIALISE THE DATA PATTERN.}
\texttt{EXIT IF IF NO ACTIVE LINES AVAILABLE.}
\texttt{JSR} \texttt{PC,FINACT} \texttt{FIND AN ACTIVE LINE.}
\texttt{JSR} \texttt{PC,FINACT} \texttt{FIND AN ACTIVE LINE.}
\texttt{EXIT IF IF NO ACTIVE LINES AVAILABLE.}
\texttt{EXIT IF IF NO ACTIVE LINES AVAILABLE.}
\texttt{INCR ERRBR} \texttt{SET ERROR NUMBER TO 5702.}
\texttt{JSR} \texttt{PC,SETPAR} \texttt{SET UP PARAMETERS FOR TRANSMISSION.}
\end{verbatim}
MOV @225, R0  ; Pass length of data pattern.
JSR PC.TXDATP ; Transmit data pattern.
BCC #01 ; Exit if error found during tx.
MOV R1,R5 ; Copy the line number.

** Wait for DMA to complete, then wait for the last character to arrive in the FIFO. **

** Inc ERRNRBR ; Set error number to 5703. **
** JSR PC.WAITTX ; Wait for transmission to complete. **
** BCC #01 ; Go report error if tx failed to complete. **
** Read the first 130 characters from the FIFO, if any XON's are found **
** Report the error, if any BMP codes are found then save them on the queue **
** And abort the test. **

** Inc ERRNRBR ; Set error number to 5704. **
** MOV #130, R0 ; Pass the number of chars to read. **
** JSR PC.READBx ; Read the first 130 chars from the FIFO. **
** BCC #01 ; Go report error if BMP code found. **
** INC ERRNRBR ; Set error number to 5705. **
** TST R1 ; Check if an XON was found. **
** BNE #100 ; Go report error if an XON was found. **

** Transmit a NULL character (which causes an XOFF to be generated). **
** MOV R5, BCRA ; Select the line ready for transmission. **
** MOVB #0, #DATA ; Transmit a NULL character. **
** INC ERRNRBR ; Set error number to 5706. **
** JSR PC.WAITTX ; Wait for tx to complete. **
** BCC #01 ; Go report error if tx did not complete. **

** Read three characters, to cause the XON to be generated. **
** INC ERRNRBR ; Set error number to 5707. **
** MOV R3, R0 ; Set the read count to 3. **
** JSR PC.READBx ; Read 3 characters from the FIFO. **
** BCC #01 ; Go report error if FIFO empty. **
** INC ERRNRBR ; Set error number to 5708. **
** TST R1 ; Check if an XON was found. **
** BNE #401 ; Go report error if an XON was found. **

** Transmit 62 characters to bracket the XON and fill the FIFO with 191 chars. **

** MOV #62, R0 ; Pass length of data pattern. **
** MOV R5, R1 ; Pass the line number. **
** INC ERRNRBR ; Set error number to 5709. **
** JSR PC.TXDATP ; Transmit data pattern. **
** BCC #01 ; Exit if error found during tx. **

** Wait for DMA to complete, then wait for the last character to arrive in the FIFO. **

** INC ERRNRBR ; Set error number to 5710. **
** JSR PC.WAITTX ; Wait for tx to complete. **
BCC 501
GO REPORT ERROR IF TX FAILED TO COMPLETE.

6979 READ THE FIRST 126 CHARACTERS.
6980 READ THE NEXT 4 CHARACTERS AND CHECK IF THEY ARE IN THE FOLLOWING ORDER
6981 NULL, XOFF, XON, NULL.

6982 INC ERRNB
6983 MOV @126.,R0
6984 SET UP READ COUNTER.
6985 JSR PC, READBX
6986 READ THE FIRST 126 CHAR.
6987 BCC 501
GO REPORT THE ERROR IF FIFO EMPTY.
6988 INC ERRNB
6989 SET ERROR NUMBER TO 5712.
6990 TST R1
6991 CHECK IF AN XON WAS FOUND.
6992 BNE 401
6993 SET ERROR NUMBER TO 5713.
6994 INC ERRNB
6995 GET THE RECEIVED BUFFER ADDRESS.
6996 MOV R6, @15254
6997 READ THE NULL CHARACTER FROM THE FIFO.
6998 CMPB R2, #0000
6999 CHECK IF IT IS A NULL CHARACTER.
7000 BNE 401
7001 GO REPORT THE ERROR IF THE SAME.
7002 MOV R5, R3
7003 GO REPORT THE ERROR IF NOT THE SAME.
7004 MOV (R5), R2
7005 READ THE XOFF FROM THE FIFO.
7006 CMPB R2, #00
7007 CHECK IF THE READ CHAR IS AN XOFF.
7008 BNE 401
7009 GO REPORT THE ERROR IF NOT THE SAME.
7010 MOV (R5), R2
7011 READ THE XON FROM THE FIFO.
7012 INC ERRNB
7013 SET ERROR NUMBER TO 5715.
7014 CMPB R2, #00
7015 CHECK IF THE READ CHARACTER IS AN XON.
7016 BNE 401
7017 GO REPORT THE ERROR IF NOT THE SAME.
7018 MOV (R5), R2
7019 READ THE NULL CHARACTER FROM THE FIFO.
7020 CMPB R2, #0000
7021 CHECK IF IT IS A NULL CHARACTER.
7022 BNE 401
7023 GO REPORT THE ERROR IF NOT THE SAME.

6983 READ THE REMAINING CHARACTERS FROM THE FIFO.

6984 MOV @61.,R0
6985 SET UP READ COUNTER.
6986 JSR PC, READBX
6987 READ THE FIRST 61 CHAR.
6988 BCC 501
GO REPORT THE ERROR IF FIFO EMPTY.
6989 INC ERRNB
6990 SET ERROR NUMBER TO 5718.
6991 TST R1
6992 CHECK IF AN XON WAS FOUND.
7000 INC ERRNB
7001 SET ERROR NUMBER TO 5719.
7002 BNE 401
7003 EXIT THE TEST.
7004 MOV @EM5402,R1
7005 "FIFO ALARM SIGNAL DEFECTIVE".
7006 "FIFO ALARM SIGNAL DEFECTIVE"
7007 TRAP CERROR
7008 "FIFO ALARM SIGNAL DEFECTIVE"
7009 TRAP CERROR
7010 "FIFO ALARM SIGNAL DEFECTIVE"
7011 TRAP CERROR
7012 "FIFO ALARM SIGNAL DEFECTIVE"
7013 TRAP CERROR
7014 "FIFO ALARM SIGNAL DEFECTIVE"
7015 TRAP CERROR
7016 "FIFO ALARM SIGNAL DEFECTIVE"
7017 TRAP CERROR
7018 "FIFO ALARM SIGNAL DEFECTIVE"
7019 TRAP CERROR
7020 "FIFO ALARM SIGNAL DEFECTIVE"
7021 TRAP CERROR
7022 "FIFO ALARM SIGNAL DEFECTIVE"
7023 TRAP CERROR
7024 "FIFO ALARM SIGNAL DEFECTIVE"
7025 TRAP CERROR
7026 "FIFO ALARM SIGNAL DEFECTIVE"
7027 TRAP CERROR
7028 "FIFO ALARM SIGNAL DEFECTIVE"
7029 TRAP CERROR
7030 "FIFO ALARM SIGNAL DEFECTIVE"
7031 TRAP CERROR
7032 "FIFO ALARM SIGNAL DEFECTIVE"
7033 TRAP CERROR
7034 "FIFO ALARM SIGNAL DEFECTIVE"
7035 TRAP CERROR
7036 "FIFO ALARM SIGNAL DEFECTIVE"
7037 TRAP CERROR
7038 "FIFO ALARM SIGNAL DEFECTIVE"
7039 TRAP CERROR
7040 "FIFO ALARM SIGNAL DEFECTIVE"
7041 TRAP CERROR
7042 "FIFO ALARM SIGNAL DEFECTIVE"
7043 TRAP CERROR
7044 "FIFO ALARM SIGNAL DEFECTIVE"
7045 TRAP CERROR
7046 "FIFO ALARM SIGNAL DEFECTIVE"
7047 TRAP CERROR
; SET INTERNAL LOOPBACK ON THE FIRST ACTIVE LINE AND ENABLE RECEIVERS. SET UP
; THE LPR'S TO 38.4K BAUD, 8 BITS/CHAR, ODD PARITY, 2 STOP BITS.
;jsr pc,finact

; FIND AN ACTIVE LINE FOR THIS TEST.
; continue if a line has been found.
;jmp 601

; EXIT THE TEST IF NO LINES ACTIVE.
; pass parameter for internal loopback,
; enable receivers.
; initialise the line control reg.
; set the LPR'S TO 38.4K BAUD.
; set delay time of 10 milli secs.
; wait for INCTRL and LPR REGS TO BE UPDATED.
; set up the loop 10 test the RX TIMER WITH DELAYS OF 15.31, 63.127 AND 255 MS.
; DMA 19 CHARACTERS INTO THE FIFO AND THEN ENABLE INTERRUPTS, VERIFY THAT
; THE INTERRUPT OCCURS WITHIN +/- 20% OF THE RX TIMER VALUE.
; mov r1,701

; save the line number.
; set the first (RX TIMER VALUE + 1).
; mov $803, errnbr

; set the error number to $803.
;jsr pc,pufifo

; purge the RX FIFO.
; continue if successful.
;jmp 501

; report the error if fifo failed to purge.
; pass the line number.
; pass the number of chars to DMA.
; set the error number to $804.
; perform the DMA from addr #BUFFAS, I:IS SUBR
; produces errors >>>>> $804 thru $805 <<<<<
; continue if successful.
; jmp 501

; report the error if one occurred.
; calculate the time-out value for the RX_INT, SET UP THE RX TIMER, AND
; WAIT FOR THE RX_INT.
; mov r5,r1

; copy the RX TIMER VALUE + 1.
; mov r5,ro

; copy the RX TIMER VALUE.
; asl r1

; multiply by 4 to obtain.
; asl r1

; time-out for the RX_INT.
; clr r0, csrc

; clear the ind addr reg bits of the CSR ready,
; for the write to the RX TIMER REG.
; move ro,rxth

; load the RX TIMER reg.
; mov #2,r4

; set delay of 2 ms.
; mov #2,r4

; delay to allow the RX TIMER VALUE TO UPDATE.
; mov #0, r2,r3

; indicate to test bit 0.
; mov $8110, r2

; indicate to test for a "1".
; mov $8110, r2

; pass addr of word to test.
; mov $8110, r2

; clear the RX_INT COUNT.
; clr rxintc

; set the RX_INT ENABLE BIT IN IESTAT.
; enable RX-INTS.
; mov #80, r4

; wait for the int to occur.
; bcs 4f

; avoid error report if the interrupt occurred.
; report the time-out error. >>>>> 5806 <<<<.
; mov r5,r2

; pass the RX TIMER VALUE to,
DEC R2 ; THE ERROR REPORTING ROUTINE.
MOV @EM5803,R1 ; PASS THE MESSAGE,
MOV @EM5805,R1 ; "RXTIMER BAD, TIME-OUT OCCURRED WAITING FOR
; THE RX-INT".

ERRDF 5806,EM5801,ER5801 ; REPORT ERROR 5806.

TRAP 5806
.WORD 5806
.WORD EM5801
.WORD ER5801

BIT #BIT06,OPTION ; HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
BEQ 601 ; EXIT THE TEST IF IT HASN'T
BR 61 ; BRANCH TO TEST ANOTHER RXTIMER VALUE.

6963 ; CHECK THAT THE INTERRUPT OCCURRED WITHIN +/- 20% OF THE RXTIMER VALUE.

6965 ; THIS SUBROUTINE REPORTS ERROR >>>>> 5807 <<<<<.

6967 MOV @EM507,.ERRNBR ; SET THE ERROR NUMBER TO 5807.
6968 MOV R5,R2 ; PASS THE RXTIMER VALUE TO,
6969 MOV C,R2 ; THE "CHECK TIME" SUBR.
6970 JSR PC,CRKRIX ; CHECK THE TIME TAKEN AND REPORT ANY ERROR.
6971 JSR PC,EXEQR ; TEST THE "EXIT ON ERROR" FLAG.
6972 JSR PC,RAIEO ; DISABLE RX-INTS.
6973 JSR PC,DXIEX ; DMA ANOTHER CHARACTER TO FILL THE FIFO TO THE 75% LEVEL, AND CHECK THAT THE
; RX-INT OCCURS IMMEDIATELY.

6978 MOV R0,R1 ; PASS THE LINE NUMBER.
6979 MOV #1,R3 ; PASS THE NUMBER OF CHAR TO DMA.
6980 INC ERRNBR ; SET THE ERROR NUMBER TO 5808.
6981 JSR PC,DMABUF ; PERFORM THE DMA FROM ADDR @xBF8A.THIS SUBR
; PRODUCES ERRORS >>>>> 5808 THRU 5809 <<<<<.
6982 BCC 580 ; REPORT THE ERROR IF OCCURRED.
6983 CLR RXINTC ; CLEAR THE RX-INT COUNT.
6984 MOV #5,R1 ; SET THE TIME-OUT TO 5 MS.
6985 MOV @RXINTC,R2 ; PASS ADDR OF WORD TO TEST.
6986 JSR PC,RAIE1 ; ENABLE INTERRUPTS.
6987 JSR PC,WAIIBS ; WAIT FOR THE INT TO OCCUR.
6988 BCS 601 ; AVOID THE ERROR IF AN INTERRUPT OCCURRED.

6990 ; REPORT THE ERROR, RX-INT DID NOT OCCUR IMMEDIATELY >>>>> 5810 <<<<<.

6993 MOV R5,R2 ; PASS THE RXTIMER VALUE.
6994 DEC R2 ; "RXTIMER BAD, RX-INT DID NOT OCCUR
6995 MOV @EM5803,R1 ; IMMEDIATELY WHEN RXFIFO 3/4 FULL".

6996 
6997 
6998 ERRDF 5810,EM5801,ER5801 ; REPORT ERROR 5810.

TRAP 5810
.WORD 5810
.WORD EM5801
.WORD ER5801

7000 BIT #BIT06,OPTION ; HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
7001 030646 032767 000100 151334
7002 03065A 001520 BEQ 60: ;EXIT THE TEST IF IT HASN'T.
7004
7006
7007 030656 004767 165332 64: JSR PC.RXIEO ;DISABLE INTERRUPTS.
7008 030662 006505 ASL H5 ;MULTIPLY (RX_TIMER VALUE) BY 2.
7009 030664 020527 000400 MOV R5,.#256. ;HAVE ALL VALUES BEEN TESTED?
7010 030670 003621 CMPL R5, R1 ;TEST RX_TIMER WITH INDEFINATE DELAY SET.
7011
7012 030672 012767 013263 153152 MOV $5811,.ERRNBR ;VERIFY THAT WHEN RX_TIMER VALUE IS 0 THE INTERRUPT IS DELAYED INDEFINITELY.
7013 030676 004767 164464 JSR PC.PUFIFO ;UNLESS THE PXIFO IS 75% FULL OR MORE.
7014
7015 030700 004767 164464 MOV $5807,.ERRNBR ;PURGE THE PXIFO.
7016 030704 103102 BCC 50: ;REPORT THE ERROR IF THE FIFO FAILED TO PURGE.
7017 030706 016701 000234 MOV 704,.R1 ;REPORT THE ERROR IF THE FIFO FAILED TO PURGE.
7018 030712 012703 000277 MOV #191,.R3 ;PASS THE NUMBER OF CHARS TO DMA.
7019 030716 005267 153130 INC ERRNBR ;PASS THE ERROR NUMBER 5812.
7020 030722 004767 163374 JSR PC.DMABUF ;PERFORM THE DMA FROM ADDR #BUFFAS THIS SUBR
7021
7022 030726 103071 BCC 50: ;PERFORM ERRORS >>> 5812 THRU 5813 <<<<.
7023
7024 030730 012701 001750 MOV #1750,.R1 ;REPORT THE ERROR IF ONE OCCURRED.
7025 030734 012702 002266 MOV #RXINTC,.R2 ;INDICATE TO TEST BITO WITH TIME OUT OF 1 SEC.
7026 030740 005067 151322 MOV #RXINTC,.R2 ;PASS THE ADDR OF THE WORD TO TEST.
7027 030744 105077 151256 CLR RXINTC ;CLEAR THE RX-INT COUNT.
7028 030750 105077 151254 CLRB #CSRA ;CLEAR THE IND.ADDR.REG BITS OF THE CSR READY.
7029 030754 012704 000002 CLR #RXTHA ;FOR THE WRITE TO THE RX_TIMER REG.
7030 030760 004767 163276 MOV #2,.R4 ;SET THE VALUE 0 IN THE RX_TIMER.
7031 030764 004767 163444 MOV #2,.R4 ;SET THE DELAY OF 2 MS.
7032 030768 004767 164442 JSR PC.DELAY ;DELAY TO ALLOW THE RX_TIMER VALUE TO UPDATE.
7033 030770 004767 164442 JSR PC.RXIE1 ;ENABLE RX-INTS.
7034 030774 004767 164442 JSR PC.WAITBUS ;WAIT FOR THE INTERRUPT TO OCCUR.
7035 030778 010307 BCC 51: ;AVOID THE ERROR IF NO INTERRUPT.
7036
7037 030776 012701 007201 MOV @EMS804,.R1 ;REPORT THE ERROR, RX-INT OCCURED WITH RX_TIMER VALUE ZERO.>>>> 5814 <<<<.
7038
7039 030786 104455 MOV $5814,.R1 ;PASS THE MESSAGE.
7040
7041 031002 011266 "RX_TIMER BAD, RX-INT OCCURED WITH RX_TIMER VALUE ZERO".
7042
7043 031007 002018 ERROF 5814,EMS801,EROS03 ;REPORT ERROR 5814.
7044
7045 031012 000441 BR 60: ;EXIT THE TEST.
7046
7047 031014 004767 165374 64: JSR PC.RXIEO ;VERIFY THAT WHEN THE FIFO IS 75% FULL THE INTERRUPT OCCURS IMMEDIATELY.
7048 031020 012767 013267 153024 MOV $5815,.ERRNBR ;DISABLE RX-INTS.
7049 031026 016701 000114 MOV 704,.R1 ;SET THE ERROR NUMBER.
7050 031032 012703 000001 MOV #1,.R3 ;PASS THE NUMBER OF CHARS TO DMA.
7051 031036 004767 163260 JSR PC.DMABUF ;PERFORM THE DMA FROM ADDR #BUFFAS THIS SUBR
7052
7053 031042 103023 BCC 50: ;PRODUCES ERRORS >>> 5815 THRU 5816 <<<<.
7054
CLR RXINTC ; CLEAR THE RX-INT COUNT.
MOV $5,R1 ; SET THE TIME-OUT TO 5 MS.
MOV @RXINTC,R2 ; PASS ADDR OF WORD TO TEST.
JSR PC,RAEIEI ; ENABLE INTERRUPTS.
JSR PC,WAITIS ; WAIT FOR THE INT TO OCCUR.
BCS 60$ ; EXIT THE TEST IF AN INTERRUPT OCCURED.

; REPORT THE ERROR, RX-INT DID NOT OCCUR IMMEDIATELY.>>>>> 5817 <<<<<.
CLR R2 ; PASS THE RXTIMER VALUE.
MOV @EMS803,R1 ; PASS THE MESSAGE.

ERRDF 5817,EM801,ER5801 ; REPORT ERROR 5817.

TRAP CIERDF .WORD 5817
TRAP .WORD 5817
TRAP .WORD EM801
TRAP .WORD EP5801

BR 60$ ; EXIT THE TEST.

JSR PC,TSABR ; REPORT NON-RELATED TEST ERROR.
SETPRI @PRIO7 ; DISABLE ALL INTERRUPTS.

MOV @PRIO7,RO ; TRAP CISPRI
MOV CISPRI

JSR PC,RXIEO ; DISABLE DEVICE RX-INTS.
CLRVECA RXVECA ; CLEAR DOWN THE RX VECTOR.

MOV RXVECA,RO ; TRAP CISCVE
MOV CISCVE

CLR CTRLCF ; INDICATE THAT WE ARE NOT WITHIN A TEST.
EXIT TST ; TRAP CIEXIT

.HEAD 0 ; LOCAL STORAGE FOR LINE NUMBER USED IN THE TEST.

ENDTST ; TRAP CIETST
**SBTTL** HARDWARE TEST

**ITXTCTF**

* * *

**THIS TEST VERIFIES THAT THE DUT'S TX-ACTION FIFO CAN CORRECTLY**

**HOLD 16 TX-ACTIONS. ONE CHARACTER IS TRANSMITTED ON EACH LINE**

**USING DMA, THE TX-ACTIONS ARE THEN READ FROM THE FIFO, VERIFYING**

**THAT THEY ARE IN THE CORRECT ORDER AND THAT THERE ARE 16 OF THEM.**

**THE TEST ALSO VERIFIES THAT THE OUT WILL NOT SEND TX-INTS AFTER**

**THE TX-ACTION FIFO HAS BEEN EMPTIED.**

**THIS TEST IS PERFORMED IN INTERNAL LOOPBACK ON ALL LINES.**

* * *

**BGNTST**

**T15:**

**SETPRI #PRI05**

**HALLOW LTC INTERRUPTS.**

**MOV #PRI05, RO**

**TRAP C+SPRI**

**INUM = INUM + 1**

**INCREMENT ASSEMBLY TIME TEST COUNTER**

**MOV INUM, ISTNUM**

**SET UP THE TEST NUMBER.**

**MOV 0-1, CTRLOF**

**INDICATE THAT WE ARE IN A TEST.**

**MOV 01, ERRYP**

**SET ERROR TYPE AS FATAL IN ERROR TABLE.**

**MOV #5901, ERERERB**

**SET ERROR MESSAGE ADDRESS IN ERROR TABLE.**

**MOV #ER5901, ERERRBLK**

**SET ERROR REPORTING ROUTINE.**

**RESET THE OUT TO A KNOWN STATE, REMOVE ANY STATUS CODES IN THE FIFO.**

**CLEAR THE RX AND TX ENABLE BITS IN THE CSR.**

**THIS SUBROUTINE REPORTS ERRORS >>>>5901<<<<.**

**JSR PC, CMRST**

**RESET THE DHU-11 REPORT ANY ERRORS FOUND.**

**BCS 36**

**SLEEP EXIT OF TEST IF NO FATAL ERROR FOUND.**

**JMP 60**

**EXIT THE TEST IF FATAL ERROR FOUND.**

**SET INTERNAL LOOPBACK ON ALL LINES AND ENABLE RECEIVERS. SET UP THE**

**LPR'S TO 38.4K BAUD, 8 BITS/CHARA, ODD PARITY, 2 STOP BITS.**

**MOV #MAPLN8R8, R5**

**INDICATE TO SET UP ALL LINES.**

**MOV #0240, RO**

**PASS PARAMETER FOR INTERNAL LOOPBACK.**

**DISABLE RECEIVERS.**

**JSR PC, NULMN**

**INITIALISE THE LINE CONTROL REGS.**

**MOV #177670, RO**

**PASS THE LPR CONTENTS.**

**JSR PC, NULMFR**

**SET THE LPR'S TO 38.4K BAUD.**

**MOV #01, R4**

**PASS DELAY TIME OF 10 MILLI SECS.**

**JSR PC, ODEL**

**WAIT FOR LINCTRL AND LPR REGS TO BE UPDATED.**

**INITIATE A DMA ON EACH LINE AND WAIT FOR ALL DMA'S TO COMPLETE.**

**CLR R1**

**PASS THE FIRST LINE NUMBER.**

**MOV #0UFBAS, R2**

**PASS THE START OF THE DATA PATTERN TO TX.**

**MOV #1, R3**

**PASS THE LENGTH OF THE DATA PATTERN.
7134  031304  012704  000005  MOV  #5,R4  ;PAS THE DELAY TIME OF 5 MILLI SECS.
7139  031310  005267  152536  INC ERRNBR  ;SET THE ERROR NUMBER TO 5902.
7143  031354  022701  000020  CMP #16,R1  ;BRANCH TO INITIATE ANOTHER DMA IF
7148  031340  001365  000000  BNE 2#  ;NOT ALL LINES SERVED.
7153  031352  005001  007402  INC ERRNBR  ;READ THE TX-ACTIONS FROM THE TX-ACTION FIFO AND VERIFY THAT THEY OCCURED
7158  031346  012703  152504  MOV #EMS902,R3  ;IN THE CORRECT ORDER.
7163  031354  017702  150646  CLR R1  ;SET THE ERROR NUMBER TO 5903.
7168  031360  100150  150646  MOV @CSR,R2  ;SET THE ERROR MESSAGE TO.
7173  031354  003502  177760  BNE 2#  ;"TX-ACTION RECEIVED FROM THE WRONG LINE".
7178  031364  042702  177760  BIT #BIT0,OPTION  ;CLEAR THE LINE NUMBER.
7183  031370  020201  000000  CMP R2,R1  ;EXIT IF EXTENDED ERROR REPORTING HAS NOT
7188  031376  032767  001000  150604  TRAP CSRERROR  ;BEEN REQUESTED.
7193  031344  001552  060#  ;INCREMENT THE EXPECTED LINE NUMBER.
7198  031346  005201  000020  CMP #16,,R1  ;BRANCH AND READ ANOTHER TX-ACTION IF,
7203  031354  013577  000000  BNE 4#  ;NOT ALL LINES HAVE BEEN SERVICED.
7208  031316  005777  150604  BIT #BIT0,OPTION  ;16 TX-ACTIONS HAVE BEEN READ, THE TX-ACTION BIT SHOULD NOW BE CLEAR.
7213  031422  100024  150646  TEST THE TX-INTERRUPTS.
7218  031424  012767  013421  152420  MOV #5905,,ERRNBR  ;CHECK THAT IT IS CLEAR, IF IT ISN'T THEN COUNT THE NUMBER OF EXTRA
7223  031432  012702  000021  150664  MOV #17,,R2  ;TX-ACTIONS RECEIVED AND REPORT THE ERROR.
7228  031436  005777  150646  8#  ;BRANCH IF THE TX-ACTION BIT IS CLEAR, TO
7233  031442  100123  150646  BPL 10#  ;TEST THE TX-INTERUPTS.
7238  031444  005202  00145  ;SET THE ERROR NUMBER TO 5905.
7243  031446  022702  000000  CMP #101,,R2  ;SET R2 TO BE THE NUMBER OF TX ACTIONS FOUND.
7248  031452  001371  000000  CMP R2,#101  ;READ THE CSR.
7253  031454  005267  152372  8#  ;BRANCH AND REPORT ERROR IF THE TX-ACTION FIFO,
7258  031460  012701  007570  150646  BPL 16#  ;FINALLY CLEARED.
7263  031444  005202  00145  ;INCREMENT THE NUMBER OF TX-ACTIONS FOUND.
7268  031446  022702  000000  CMP #101,,R2  ;IF LESS THAN 100 TX-ACTIONS HAVE BEEN READ.
7273  031452  001371  000000  CMP R2,#101  ;THEN CONTINUE READING THE CSR.
7278  031454  005267  152372  BNE 8#  ;OTHERWISE REPORT THE ERROR.
7283  031444  005202  00145  ;REPORT THE ERROR "TX-ACTION FIFO WOULD NOT EMPTY"
7288  031454  005267  152372  INC ERRNBR  ;SET THE ERROR NUMBER TO 5906.
7293  031460  012701  007570  MOV #EMS904,R1  ;SET THE ERROR MESSAGE TO,
08: CLR R1 ; PASS THE NUMBER OF THE FIRST LINE

119: INITIATE A DMA ON ALL LINES AND WAIT FOR ALL DMA'S TO COMPLETE.

122: MOV #5907, ERRBR ; SET THE ERROR NUMBER TO 5907.
MOV #BUFNB, R2 ; PASS THE START OF THE DMA PATTERN TO TX.
MOV 03A, R3 ; PASS THE LENGTH OF THE DATA PATTERN.
JSR PC, DMA ; TRANSIT THE DATA PATTERNS.
BCC 50F ; AVOID THE TEST IF ERROR FOUND DURING DMA TX.
INC R1 ; INCREMENT THE LINE NUMBER.
CMP #0001, R1 ; BRANCH TO INITIATE ANOTHER DMA IF.
BNE 128 ; ALL LINES NOT SERVED.
MOV #100, R4 ; SET THE DELAY OF 100 MILLISECS.
JSR PC, DELAY ; WAIT FOR THE DMA'S TO COMPLETE.

178: SET UP THE INTERRUPT SERVICE ROUTINE THAT WILL READ THE TX ACTION FIFO

181: UNTIL EMPTY AND CHECK FOR ANY SUBSEQUENT INTERRUPTS WITH NO TX_ACTION. CAN ENABLE TX INTERRUPTS.

185: CLR TXINIC ; CLEAR THE TX INT COUNTER.
CLR TAINF ; CLEAR THE TX INT FLAGS.
SETVIC TXVECA, TXAINT, #Prio6
MOV #Prio6, (SP)
MOV #TXAINT, (SP)
MOV TXVECA, (SP)
MOV #81, (SP)
TRAP C15VECA
ADD #10, SP

192: SETPRI #Prio4 ; ALLOW DEVICE INTERRUPTS.
MOV #Prio4, (SP)
TRAP C15PRI

200: JSR PC, TIIEI ; ENABLE TX INTERRUPTS.

205: WAIT FOR THE INTERRUPTS TO OCCUR

209: MOV #5, R4 ; SET THE DELAY FOR 5 MILLISECS.
JSR PC, DELAY ; DELAY FOR 5 MS.

215: DISABLE INTERRUPTS AND CLEAR DOWN THE INTERRUPT SERVICE ROUTINE.

219: SETPRI #Prio7 ; DISABLE ALL INTERRUPTS.
MOV #Prio7, (SP)
TRAP C15PRI

225: JSR PC, TIIEO ; DISABLE DUT TX INTERRUPTS
CLRVECA TXVECA ; CLEAR THE TX INT VECTOR
MOV TXVECA, (SP)
TRAP C15VECA
7235  * VERIFY THAT A TX INTERRUPT OCCURRED, AND THAT NO TX INTERRUPT OCCURRED WHEN 7236  * THE TX-ACTION FIFO WAS EMPTY.
7237
7238  * SET THE ERROR NUMBER TO 5908.
7239  INC ERANNBR
7240  TST TXINTC
7241  BEQ 50:
7242  INC ERANNBR
7243  TST TXINTF
7244  BPL 60:
7245  MOV @EMS905.R1
7246  MOV @ER0503.ERRBLK
7247  BR 18:
7248  "TX INT OCCURRED AFTER TX FIFO HAD BEEN EMPTYED".
7249  "ERROR REPORTING.
7250  "ERROR REPORTING.
7251  "ERROR REPORTING.
7252  "ERROR REPORTING.
7253  "ERROR REPORTING.
7254  "ERROR REPORTING.
7255  031702 010102
7256  148: MOV R1,R2
7257  031704 012767 013420 152140
7258  164: MOV @5904,ERANNBR
7259  031716 012701 000020
7260  007476
7261  031722 104640
7262  031724 000402
7263  188: ERROR
7264  BR 60:
7265  "REPORT THE ERROR.
7266  "EXIT THE TEST.
7267  "REPORT THE ERROR.
7268  "EXIT THE TEST.
7269
7270  "REPORT THE ERROR.
7271  "EXIT THE TEST.
7272
7273  "REPORT THE ERROR.
7274  "EXIT THE TEST.
7275
7276  "REPORT THE ERROR.
7277  "EXIT THE TEST.
7278
7279  "REPORT THE ERROR.
7280  "EXIT THE TEST.
7281
7282  "REPORT THE ERROR.
7283  "EXIT THE TEST.
7284
7285  "REPORT THE ERROR.
7286  "EXIT THE TEST.
7287
7288  "REPORT THE ERROR.
7289  "EXIT THE TEST.
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7269  SBTLT  HARDWARE TEST  TXFIFO  
7270  *                    TXFIFO TEST  *
7271  *  THIS TEST IS USED TO VERIFY THAT THE DUT'S TRANSMIT FIFO'S CAN HOLD  *
7272  *  64 CHARACTERS AND THAT ONLY ONE TX INTERRUPT OCCURS FOR ALL 64  *
7273  *  CHARACTERS. THE TEST ALSO EXERCISES THE BYTE SWAPPER BY USING  *
7274  *  ALTERNATE WORD AND BYTE WRITES TO THE TX FIFO, AS WELL AS CHECKING  *
7275  *  THE FIFO SIZE DRIVER REPORTS THE NUMBER OF CHARACTERS IN THE FIFO  *
7276  *  CORRECTLY. ANY BMP CODES FOUND ARE SAVED ON THE QUE TO BE REPORTED  *
7277  *  LATER.  *
7278  *  THE TEST IS PERFORMED IN INTERNAL LOOPBACK ON ALL ACTIVE LINES.  *
7279  *                    
7280  BGNSTI
7281
7282  031740
7283  031740
7284  031740  012700  000240
7285  031744  104441
7286  031746  012767  000020  150300
7287  031754  012767  177777  150266
7288  031762  012767  000001  152060
7289  031770  012767  013561  152054
7290  031776  012767  007754  152050
7291  032004  005067  150602
7292  032010  005067  150256
7293  032014  012700  002614
7294  032020  004767  162214
7295
7296  SETPRI  #PRI05  IALLOW LIC INTERRUPTS.
7297  T6::  MOV  #PRI05.R0
7298  TRAP  C1SPRI
7299
7300  TNUM  =  TNUM  +  1  ;INCREMENT ASSEMBLY TIME TEST COUNTER
7301  MOV  TNUM,TNUM  ;SET UP THE TEST NUMBER.
7302  MOV  #1,CTRCLCF  ;INDICATE THAT WE ARE IN A TEST.
7303  MOV  #1,ERRTYP  ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
7304  MOV  #6001,ERRNR  ;SET THE ERROR NUMBER TO 6001.
7305  MOV  #8001,ERRMSG  ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
7306  CLR  ERRSHF  ;CLEAR THE ERROR SUMMARY FLAGS.
7307  CLR  EXDERR  ;CLEAR THE "EXIT ON ERROR" FLAG.
7308  MOV  #8000,RO  ;SET UP THE START ADDRESS FOR THE BLOCK.
7309  JSR  PC,CLR16W  ;CLEAR THE BLOCK OF 16 WORDS.
7310
7311  ;RESET THE DUT TO A KNOWN STATE, REMOVE ANY STATUS CODES IN THE FIFO.
7312  ;CLEAR THE RX AND TX ENABLE BITS IN THE CSR.
7313  ;THIS SUBROUTINE REPORTS ERROR >>> 6001 <<<.
7314  ;
7315  J6::  JSR  PC,CLR6ST  ;RESET THE DMU-11 REPORT ANY ERRORS FOUND.
7316  BCS  .+6  ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
7317  JMP  621  ;EXIT THE TEST IF FATAL ERROR FOUND.
7318
7319  ;SET INTERNAL LOOPBACK ON ALL ACTIVE LINES AND ENABLE RECEIVERS. SET UP THE
7320  ;LPR'S TO 38.4K BAUD, 8 BITS/CHARA, ODD PARITY, 2 STOP BITS. DISABLE
7321  ;TRANSMITTERS ON ALL LINES.
7322
7323  T0::  MOV  #MAPLS.R5  ;INDICATE TO DISABLE ALL LINES.
7324  J5::  JSR  PC,TX5DBL  ;DISABLE TX ON ALL LINES.
7325  MOV  ACTLNS.R5  ;INDICATE TO SET UP ACTIVE LINES ONLY.
7326  MOV  #204.RD  ;PASS PARAMETER FOR INTERNAL LOOPBACK.
7327  JSR  PC,WTMLNC  ;INITIALISE THE LINE CONTROL REGS.
7328  MOV  #177660.R0  ;PASS THE LPR CONTENTS.
7329  JSR  PC,WTMLPR  ;SET THE LPR'S TO 38.4K BAUD.
7330  MOV  #610.R4  ;PASS DELAY TIME OF 10 MILLI SEC.
7331  JSR  PC,DLY5  ;WAIT FOR LMSCTR AND LPR REGS TO BE UPDATED.
7332
7333  JSR  PC,DLY5  ;SET UP THE TX INTERRUPT SERVICE ROUTINE AND VECTOR. THE ROUTINE COUNTS
7334  ;THE NUMBER OF INTERRUPTS AND CHECKS FOR ANY INTERRUPTS OCCURING WITH
7335  ;NO TX-ACTION.
7323  032102  MOVE  LENGTH, R5
032102  012746  000300
032106  012746  017734
032110  016746  150100
032114  03166   000003
032118  104437   000001
032122  062706   000010
032126  000476  165122

7324  032130  JSR   PC, TXIE1 ; SET THE TX INT ENABLE BIT IN THE CSR.

7325  032134  MOV   ACLRNS, R5 ; SET UP THE ACTIVE LINE BIT MAP.
7326  032140  CLR   R1 ; SET UP THE FIRST LINE NUMBER.

7328  032142  000241
7329  032144  006005
7330  032146  105007
7331  032150  110177  150052
7332  032154  010103
7333  032156  000303

7334  032160  01270C   000003
7335  032164  016377  150044
7336  032166  150203
7337  032172  110377  150303
7338  032176  062703   000401
7339  032202  005300
7340  032204  001367
7341  032206  000502
7342  032210  117702  150020
7343  032214  122702  000667
7344  032220  001423

7345  032222  010104

7346  032224  012701  000667
7347  032230  010346
7348  032232  012767  013562  151612
7349  032240  012703  010000
7350  032244  012767  012767  151604
7351  032252  010401
7352  032254  104460
7353  032256  012603
7354  032260  032767   001000  147722
7355  032266  001002
7356  032270  000167  000444

7357  032272

7358  032274  MOV   R1, R4 ; PASS THE LINE NUMBER TO THE ERROR ROUTINE.
7359  032278  MOV   R5, R1 ; PASS THE EXPECTED FIFO SIZE.
7360  032282  MOV   R3, (SP) ; SAVE THE DATA PATTERN.
7361  032286  MOV   #6002, R4 ; SET THE ERROR NUMBER TO 6002.
7362  032290  MOV   #EH6002, R3 ; PASS THE MESSAGE.
7363  032294  MOV   #ER6001, R6 ; "INCORRECT VALUE IN FIFO SIZE REGISTERS".
7364  032298  MOV   #ER6001, R6 ; "INCORRECT VALUE IN FIFO REGISTER".
7365  032302  MOV   (SP), R3 ; RESTORE THE PATTERN.
7366  032306  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7367  032310  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7368  032314  MOV   (SP), R3 ; RESTORE THE PATTERN.
7369  032318  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7370  032322  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7371  032326  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7372  032330  MOV   R4, R1 ; RESTORE THE LINE NUMBER.
7373 032274 012700 000022  61:  MOV 018.,RO  
7375 032300 010377 147750  81:  MOV R5,0F0DATA  
7376 032304 105203  
7377 032306 110377 147722  
7378 032312 062703 000401  
7379 032316 005300  
7380 032320 01367  
7381 032322 110377 147706  
7382  
7383  
7384  
7385 032336 005705  101:  TST R5  
7386 032330 001400  121:  AND R5,0001  
7387 032332 005201  
7388 032334 000700  
7389  
7390  
7391  
7392  
7393  
7394  
7395 032336 016705 147656 121:  MOV ACILNS,R5  
7396 032342 005001  141:  CLR R1  
7397  
7398 032344 000241  141:  CLC  
7399 032346 060005  
7400 032350 103157  
7401 032352 110177 147650  
7402 032354 110177 147660  
7403 032360 000003  
7404 032362 012767 013563 151462  
7405 032370 004767 162774  
7406 032374 103151  
7407 032376 147670  
7408 032402 005067 147666  
7409 032406 052777 100000 147626  
7410 032414  
7411 032414 012700 000200  
7412 032420 104441  
7413 032422 012704 000144  
7414 032426 004767 161630  
7415 032432 100513  
7416 032440 042777 100000 147574  
7417  
7418 032446 005267 151400  
7419 032452 005767 147614  
7420 032456 001520  
7421 032460 005267 151366  
7422 032464 005767 147604  
7423 032470 100513  
7424 032472 022767 000001 147572  

; SET UP THE MASTER LOOP.
; MOVE A WORD OF DATA INTO THE FIFO.
; INCREMENT THE LOW BYTE OF THE DATA PATTERN.
; MOVE A BYTE OF DATA INTO THE FIFO.
; INCREMENT THE HIGH AND LOW BYTE OF THE DATA.
; DECREMENT THE COUNT.
; BRANCH IF NOT ALL 54 CHARACTERS WRITTEN.
; LOAD THE LAST CHARACTER.

; LOOP UNTIL THE TX FIFO'S ON ALL ACTIVE LINES HAVE BEEN FILLED.
; HAVE ALL FIFO'S BEEN FILLED.
; YES, THEN GO AND TRANSMIT THE CHARACTERS.
; OTHERWISE SELECT THE NEXT LINE.
; BRANCH TO TEST IF ITS ACTIVE.

; ENABLE TRANSMISSION ON EACH LINE IN TURN AND WAIT FOR 100 MS TO ALLOW
; ANY INTERRUPTS TO OCCUR BEFORE DISABLING TRANSMISSION. VERIFY THAT ONLY
; INTERRUPT OCCURRED AND CHECK THAT THE CHARACTERS WERE TRANSMITTED UNCORRUPTED
; AND ON THE CORRECT LINE.

; SET UP THE ACTIVE LINE BIT MAP.
; SET THE ACTIVE LINE NUMBER.

; CLEAR THE CARRY BIT READY FOR THE ROTATION.
; ROTATE THE ACTIVE LINE BIT MAP INTO THE CARRY.
; AVOID TESTING THIS LINE IF ITS INACTIVE.
; LOAD THE LINE NUMBER OF THE DUT INTO THE CSR.
; INITIALISE THE "EXPECTED" DATA PATTERN FOR
; THIS LINE.

; MOVE @0003,ERRNR  
; SET THE ERROR NUMBER TO 6003.
; JSR PC,PURRFN  
; PURGE THE RX FIFO.
; MOV R1,RO  
; REPORT THE ERROR IF THE FIFO WOULD NOT PURGE.
; MOV R1,R3  
; CLEAR THE INTERRUPT COUNT.
; CLR TXINTC  
; CLEAR THE INTERRUPT FLAGS.
; BIS @BIT15,BTXAD2A  
; ENABLE TRANSMISSION ON THIS LINE.
; SETPRI @PRI04  
; ALLOW DEVICE INTERRUPTS.

; MOV #PRI04,RO  
; TRAP CISPRI

; BIC @BIT15,BTXAD2A  
; DISABLE TRANSMISSION ON THIS LINE.

; VERIFY THAT ONLY ONE INTERRUPT OCCURRED AND THAT IT WAS ACCOMPANIED BY
; A CORRESPONDING TX-ACTION.

; MOV #PRI05,RO  
; TRAP CISPRI

; MOV #PRI10,RO  
; TRAP CISPRI

; INC ERRNR  
; SET THE ERROR NUMBER TO 6004.
; TST TXINTC  
; HAS AN INTERRUPT OCCURRED?
; BEQ 501  
; REPORT THE ERROR IF NO INTERRUPT OCCURRED.
; INC ERRNR  
; SET THE ERROR NUMBER TO 6005.
; TST TXINTF  
; HAS AN INTERRUPT OCCURRED WITHOUT A TX-ACTION?
; BNE 501  
; REPORT THE ERROR IF IT HAS.
; CMP #1,TXINTC  
; DID ONLY ONE INT OCCUR?
7426 032500 001424 BEQ 161: AVOID THE ERROR IF ONLY ONE INTERRUPT.
7427
7428
7429
7430 032502 016702 147564 MOV IXINTC.R2 ; PASS THE ACTUAL NUMBER OF INTERRUPTS.
7431 032506 010104 MOV R1,R4 ; PASS THE LINE NUMBER.
7432 032510 012701 000001 MOV @1.R1 ; PASS THE EXPECTED NUMBER OF INTS.
7433 032514 010346 MOV R3,(SP) ; SAVE THE DATA PATTERN.
7434 032516 012703 010040 MOV @EM6003.R3 ; PASS THE MESSAGE.
7435
7436
7437 032522 005267 151324 INC ERRBR ; SET THE ERROR NUMBER TO 6006.
7438 032526 012767 012376 151322 MOV #ER6001,ERRBLK ; SET UP THE ERROR BLOCK.
7439 032534 02533 104460 ERROR TRAP CERROR
7440 032536 012603 MOV (SP),R3 ; RESTORE THE DATA PATTERN.
7441 032540 010401 MOV R4,R1 ; RESTORE THE LINE NUMBER.
7442 032542 032767 000100 147440 BIT @BIT06,OPTION ; HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
7443 032550 001473 BEQ 601: ; EXIT THE TEST IF IT HAS.
7444
7445
7446
7447 ; READ THE CHARACTERS FROM THE RXFIFO AND VERIFY THEY ARE CORRECT AND WERE
7448 ; RECEIVED ON THE CORRECT LINE.
7449 ; THIS SUBROUTINE REPORTS ERRORS. >>>>> 6007 THRU 6008 <<<.<
7450
7451 032552 012767 012367 151272 161: MOV #6007.,ERRBR ; SET UP THE ERROR NUMBER TO 6007.
7452 032560 005000 CLR R0 ; INITIALISE THE NUMBER OF CHARs READ COUNT.
7453 032562 012704 000025 MOV #21,R4 ; SET UP THE OUTER LOOP COUNT.
7454 032566 012702 000003 MOV #2,R2 ; SET UP THE INNER LOOP COUNT.
7455 032572 004767 163140 201: JSR PC,REPDER ; READ A CHARACTER FROM THE RXFIFO, VERIFY THAT
7456 ; IT IS CORRECT AND CAME FROM THE UUT. REPORT ANY ERRORS. >>>>> 6007 THRU 6008 <<<.<
7457
7458 032576 103022 BCC 224: ; BRANCH TO REPORT THE ERROR IF THE FIFO EMPTY.
7459 032600 005767 147446 TST EXERR ; TEST THE "EXIT ON ERROR" FLAG.
7460 032604 001055 BNE 604: ; EXIT THE TEST IF SET, I.E., AN ERROR OCCURRED.
7461 ; AND NO EXTENDED ERROR REPORTING WAS REQUESTED.
7462
7463 032606 005200 INC R0 ; INCREMENT THE READ CHAR COUNT.
7464 032610 105203 INCBR R3 ; INCREMENT THE LOBYTE OF THE DATA PATTERN.
7465 032612 000303 SMAB R3 ; SWAP BYTES TO PLACE EXPECTED CHAR IN LOBYTE.
7466 032614 003502 DEC R2 ; DECREMENT THE INNER LOOP COUNTER.
7467 032616 001565 BNE 201: ; BRANCH TO READ ANOTHER CHAR IF 3 CHARs HAVE NOT BEEN READ.
7468 032620 000303 SMAB R3 ; RESTORE THE DATA BYTES TO THE CORRECT POSITION
7469 032622 005304 DEC R4 ; DECREMENT THE OUTER LOOP COUNTER.
7470 032624 001360 BNE 181: ; BRANCH TO READ ANOTHER 3 CHARACTERS IF NOT ALL 63 HAVE BEEN READ.
7471
7472 032626 004767 163104 JSR PC,REPDER ; READ AND CHECK THE LAST CHARACTER.
7473 032632 103004 BCC 224: ; BRANCH TO REPORT THE ERROR IF THE FIFO EMPTY.
7474 032634 005767 147412 TST EXERR ; TEST THE "EXIT ON ERROR" FLAG.
7475 032640 001057 BNE 604: ; EXIT THE TEST IF SET.
7476 032642 000422 BR 241: ; OTHERWISE GO AND TEST ANOTHER LINE.
7477
7478 ; REPORT THE ERROR, NOT ALL CHARACTERS TRANSMITTED. >>>>> 6009 <<<.<
7479
7480 032644 012767 013571 151200 221: MOV #6009.,ERRBR ; SET THE ERROR NUMBER TO 6009.
7481 032652 012703 010243 MOV #EM6006.R3 ; PASS THE MESSAGE.
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>7482</td>
<td>MOV  R1,R4</td>
<td>; TX FIFO BAD, TOO FEW CHAR'S RECEIVED</td>
</tr>
<tr>
<td>7483</td>
<td>MOV  #64,R1</td>
<td>; PASS THE LINE NUMBER OF THE UT</td>
</tr>
<tr>
<td>7484</td>
<td>MOV  R0,R2</td>
<td>; PASS THE EXPECTED NUMBER OF CHARACTERS</td>
</tr>
<tr>
<td>7485</td>
<td>MOV  #ER6001,ERRLCK</td>
<td>; SET UP THE ERROR REPORTING ROUTINE</td>
</tr>
<tr>
<td>7486</td>
<td>ERROR</td>
<td>; TRAP C1ERROR</td>
</tr>
<tr>
<td>7487</td>
<td>MOV  R4,R1</td>
<td>; RESTORE THE LINE NUMBER</td>
</tr>
<tr>
<td>7488</td>
<td>BIT  #BIT06,OPTION</td>
<td>; HAS EXTENDED ERROR REPORTING BEEN REQUESTED</td>
</tr>
<tr>
<td>7489</td>
<td>BEQ  60$</td>
<td>; EXIT THIS TEST IF IT HAS</td>
</tr>
<tr>
<td>7490</td>
<td>; IF ALL ACTIVE LINES HAVE BEEN TESTED THEN EXIT THE TEST, OTHERWISE BRANCH TO</td>
<td></td>
</tr>
<tr>
<td>7491</td>
<td>TEST ANOTHER LINE</td>
<td></td>
</tr>
<tr>
<td>7492</td>
<td>TST  R5</td>
<td>; HAVE ALL LINES BEEN TESTED</td>
</tr>
<tr>
<td>7493</td>
<td>BEQ  52$</td>
<td>; YES, THEN GO AND REPORT ERROR SUMMARIES IF</td>
</tr>
<tr>
<td>7494</td>
<td>INC  R1</td>
<td>; NECESSARY</td>
</tr>
<tr>
<td>7495</td>
<td>BR  14$</td>
<td>; OTHERWISE SELECT THE NEXT LINE</td>
</tr>
<tr>
<td>7496</td>
<td>; AND BRANCH TO TEST IF IT IS ACTIVE</td>
<td></td>
</tr>
<tr>
<td>7497</td>
<td>14$</td>
<td></td>
</tr>
<tr>
<td>7498</td>
<td>50$</td>
<td></td>
</tr>
<tr>
<td>7499</td>
<td>JSR  PC,TSABRT</td>
<td>; REPORT NON-RELATED TEST ERROR</td>
</tr>
<tr>
<td>7500</td>
<td>; ABDOT THE TEST</td>
<td></td>
</tr>
<tr>
<td>7501</td>
<td>; REPORT ERROR NUMBER TO 6010</td>
<td></td>
</tr>
<tr>
<td>7502</td>
<td>JSR  PC,REPSMR</td>
<td>; REPORT ERROR SUMMARIES IF NECESSARY</td>
</tr>
<tr>
<td>7503</td>
<td>; DISABLE ALL INTERRUPTS</td>
<td></td>
</tr>
<tr>
<td>7504</td>
<td>MOV  #PRI07,RO</td>
<td></td>
</tr>
<tr>
<td>7505</td>
<td>MOV  PRI07,RO</td>
<td></td>
</tr>
<tr>
<td>7506</td>
<td>TRAP  C1SPIRI</td>
<td></td>
</tr>
<tr>
<td>7507</td>
<td>CLR  TXVECA</td>
<td>; CLEAR DOWN THE TX-INT VECTOR</td>
</tr>
<tr>
<td>7508</td>
<td>CLRCF</td>
<td></td>
</tr>
<tr>
<td>7509</td>
<td>ENDST</td>
<td></td>
</tr>
<tr>
<td>7509</td>
<td>L10045:</td>
<td></td>
</tr>
<tr>
<td>7510</td>
<td>TRAP  C1ETST</td>
<td></td>
</tr>
</tbody>
</table>
.SBTL2 HARDWARE TEST

* BREKB *

* THIS TEST VERIFIES THAT ALL SERIAL TRANSMIT LINES CAN GENERATE A BREAK *
* BY SETTING THE BK BIT IN THE ASSOCIATED LNTLR REGISTER. *
* USE OF THE INTERNAL LOOPBACK FEATURE OF THE DUARTS IS MADE TO MINIMISE *
* ANY EXTERNAL EFFECTS CAUSED ON THE SERIAL LINES BY THIS TEST. *
* FRAMING ERROR DETECTION IS USED TO INDICATE THE PRESENCE OF A BREAK. *
* BY SETTING THE APPROPRIATE BIT IN THE RBUF REGISTER. *

BGNTST

T17:

MOV -1,CTRCF ;INDICATE THAT WE ARE IN A TEST. 

TNUM = TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER. 

MOV TNUM,TSNUM ;SET UP THE TEST NUMBER. (64) 

MOV 1,ERR;YP ;SET ERROR TYPE AS FATAL IN ERROR TABLE. 

MOV #6401.,ERRBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE. 

MOV #6401.,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRBL. 

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 >>>>> 6401 <<<<. 

JSR PC,CLNST ;RESET THE DHU-11, REPORT ANY ERRORS FOUND. 

BCC 604 ;EXIT TEST IF FATAL ERROR FOUND. 

SET UP DEVICE UNDER TEST (DUT) TO: 

DISABLE TRANSMISSION AND RECEPTION INTERRUPTS. 

DELAY FOR 10 MILLI-SECONDS TO ALLOW TIME TO CLEAR ANY BREAKS. 

MOV #MARLS,RS ;PASS ACTIVE LINE BIT MAP. 

MOV #200,RO ;PASS INTERNAL LOOPBACK MODE. 

JSR PC,WTWLNC ;SELECT INTERNAL LOOPBACK,DISABLE DMA. 

MOV #10, R4 ;PASS DELAY TIME OF 10 MILLI SECONDS. 

JSR PC,DELAY ;DELAY TO ALLOW ANY BREAKS TO BE CLEARED. 

SET TRANSMISSION AN RECEPTION PARAMETERS FOR ALL LINES. 

9600 BAUD, 8 CHAR, 1 STOPBIT, NO PARITY. 

MOV #156430,RO ;SET UP BAUD RATE, ETC. 

JSR PC,WTWLPR ;SET COMMUNICATION PARAMETERS ON ALL LINES. 

ENABLE TRANSMITERS ON ALL ACTIVE LINES. 

MOV ACTLNS,RS ;PASS ACTIVE LINE BIT MAP. 

JSR PC,TXENBL ;ENABLE TRANSMISSIONS ON ALL LINES. 

PURGE THE FIFO OF ANY UNWANTED CHARACTERS. 

THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>>> 6402 THRU 6404 <<<<. 

INC ERRBR ;SET ERROR NUMBER TO 6402. 

JSR PC,PURIFR ;PURGE FIFO. 

9CC 601 ;ABORT TEST IF FIFO WILL NOT CLEAR. 

VERIFY BREAK GENERATION ON INDIVIDUAL LINES.
7567  CLEAR BREAKS ON ALL LINES.
7568  DELAY FOR 10 MILLI-SECONDS TO ALLOW TIME FOR ANY BREAKS TO BE CLEARED.
7569  SELECT LINE, SET BREAK BIT IN LNCTRL REGISTER.
7570  TEST FOR A CHARACTER IN THE FIFO WITH FRAME ERROR.

24:  CLR R2 ; CLEAR LINE COUNTER.
7571  MOV #1,R3 ; SET UP ACTIVE LINE BIT MASK.
7572  MOV #1,R5 ; COPY ACTIVE LINE BIT MASK.
7573  MOV #1,R6 ; SET UP ACTIVE LINE BIT MASK.
7574  MOV #1,R7 ; SET UP ACTIVE LINE BIT MASK.
7575  MOV #1,R8 ; SET UP ACTIVE LINE BIT MASK.
7576  MOV #1,R9 ; SET UP ACTIVE LINE BIT MASK.
7577  MOV #1,R10 ; SET UP ACTIVE LINE BIT MASK.
7578  MOV #1,R11 ; SET UP ACTIVE LINE BIT MASK.
7579  MOV #1,R12 ; SET UP ACTIVE LINE BIT MASK.
7580  MOV #1,R13 ; SET UP ACTIVE LINE BIT MASK.
7581  MOV #1,R14 ; SET UP ACTIVE LINE BIT MASK.
7582  MOV #1,R15 ; SET UP ACTIVE LINE BIT MASK.
7583  MOV #1,R16 ; SET UP ACTIVE LINE BIT MASK.
7584  MOV #1,R17 ; SET UP ACTIVE LINE BIT MASK.
7585  MOV #1,R18 ; SET UP ACTIVE LINE BIT MASK.
7586  MOV #1,R19 ; SET UP ACTIVE LINE BIT MASK.
7587  MOV #1,R20 ; SET UP ACTIVE LINE BIT MASK.
7588  MOV #1,R21 ; SET UP ACTIVE LINE BIT MASK.
7589  MOV #1,R22 ; SET UP ACTIVE LINE BIT MASK.
7590  MOV #1,R23 ; SET UP ACTIVE LINE BIT MASK.
7591  MOV #1,R24 ; SET UP ACTIVE LINE BIT MASK.
7592  MOV #1,R25 ; SET UP ACTIVE LINE BIT MASK.
7593  MOV #1,R26 ; SET UP ACTIVE LINE BIT MASK.
7594  MOV #1,R27 ; SET UP ACTIVE LINE BIT MASK.
7595  MOV #1,R28 ; SET UP ACTIVE LINE BIT MASK.
7596  MOV #1,R29 ; SET UP ACTIVE LINE BIT MASK.
7597  MOV #1,R30 ; SET UP ACTIVE LINE BIT MASK.
7598  MOV #1,R31 ; SET UP ACTIVE LINE BIT MASK.
7599  MOV #1,R32 ; SET UP ACTIVE LINE BIT MASK.
7600  MOV #1,R33 ; SET UP ACTIVE LINE BIT MASK.
7601  MOV #1,R34 ; SET UP ACTIVE LINE BIT MASK.
7602  MOV #1,R35 ; SET UP ACTIVE LINE BIT MASK.
7603  MOV #1,R36 ; SET UP ACTIVE LINE BIT MASK.
7604  MOV #1,R37 ; SET UP ACTIVE LINE BIT MASK.
7605  MOV #1,R38 ; SET UP ACTIVE LINE BIT MASK.
7606  MOV #1,R39 ; SET UP ACTIVE LINE BIT MASK.
7607  MOV #1,R40 ; SET UP ACTIVE LINE BIT MASK.
7608  MOV #1,R41 ; SET UP ACTIVE LINE BIT MASK.
7609  MOV #1,R42 ; SET UP ACTIVE LINE BIT MASK.
7610  MOV #1,R43 ; SET UP ACTIVE LINE BIT MASK.
7611  MOV #1,R44 ; SET UP ACTIVE LINE BIT MASK.
7612  MOV #1,R45 ; SET UP ACTIVE LINE BIT MASK.
7613  MOV #1,R46 ; SET UP ACTIVE LINE BIT MASK.
7614  MOV #1,R47 ; SET UP ACTIVE LINE BIT MASK.
7615  MOV #1,R48 ; SET UP ACTIVE LINE BIT MASK.
7616  MOV #1,R49 ; SET UP ACTIVE LINE BIT MASK.
7617  MOV #1,R50 ; SET UP ACTIVE LINE BIT MASK.
7618  MOV #1,R51 ; SET UP ACTIVE LINE BIT MASK.
7619  MOV #1,R52 ; SET UP ACTIVE LINE BIT MASK.
7620 033236 012705 17777
7621 033242 012700 000200
7622 033246 004767 164300
7623 033252 012704 000012
7624 033256 004767 161000
7625

7626
7627
7628
7629 033262 004767 162102
7630 033266 103044
7631
7632
7633
7634
7635 033270 016705 146724
7636 033274 012700 000214
7637 033300 004767 164246
7638
7639
7640
7641
7642 033304 012704 000012
7643 033310 004767 160746
7644 033314 010502
7645 033316 004767 161354
7646 033322 017701 146702
7647 033326 100011
7648 033330 032701 020000
7649 033334 000001
7650 033340 004270 177400
7651 033344 004767 161300
7652 033348 000001
7653 033350 004000
7654 033352 000302
7655 033354 003362
7656 033356 005705
7657 033360 003141
7658 033362 012701 010344
7659
7660 033366
7661 033367 100455
7662
7663 033400 004767 163266
7664 033404 000507 146640
7665 033410
7666 033410 104401

7620 MOV @MAPLS,R5 ;SET UP LINE TO CLEAR BREAKS ON.
7621 MOV @200,R0 ;SET UP PARAMETER TO CLEAR BREAK BIT.
7622 JSR PC,WTLN
7623 MOV @0,R4 ;CLEAR BREAK BIT,RESELECT INTERNAL LOOPBACK.
7624 JSR PC,DELAY ;DELAY TO ALLOW BREAKS TO BE CLEARED.
7625
7626 PURGE THE FIFO OF UNWANTED CHARACTERS.
7627 *
7628 *
7629 JSR PC,PUTFIFO ;PURGE FIFO.
7630 BCC 501 ;GO REPORT ERROR IF FAILED TO CLEAR OUT FIFO.
7631
7632 SET UP PARAMETERS FOR SETTING THE BREAK BIT ON ALL ACTIVE LINES.
7633 THEN CALL ROUTINE TO DO IT.
7634
7635 MOV ACTLS,R5 ;SET UP ACTIVE LINE BIT MASK.
7636 MOV @214,R0 ;SET BREAK,RESELECT LOOPBACK,ENABLE RECEPTION.
7637 JSR PC,WTLN
7638 DELAY FOR 10 MILLI SECONDS, TO ALLOW TIME FOR RECEPTION.
7639 TEST FOR CHARACTERS IN FIFO WITH FRAME ERROR BIT SET.
7640
7641 MOV @10,R4 ;SET DELAY VALUE TO 10 MILLI SECS.
7642 JSR PC,DELAY ;ALLOW TIME FOR CHARACTER RECEPTION.
7643 MOV R5,R2 ;COPY ACTIVE LINE BIT MAP.
7644 JSR PC,MAPCNT ;COUNT THE NUMBER OF LINES AVAILABLE.
7645 MOV BRFUFA,R1 ;GET CHARACTER FROM BUF REGISTER.
7646 BPL 148 ;BRANCH IF DATA VALID NOT SET.
7647 MOV BIT,line13,R1 ;CHECK FOR FRAME ERROR BIT.
7648 BEQ 148 ;DO NOT CLR FLG FOR THIS LINE IF FRAME BIT CLR.
7649 MOV R0,R5 ;CLEAR LINE FLAG.
7650 MOV R1 ;GET LINE NUMBER IN LOW BYTE.
7651 MOV @177400,R1 ;CLEAR EVERYTHING BUT THE LINE NUMBER.
7652 JSR PC,LINBIT ;CALC BIT MASK FROM LINE NUMBER.
7653 BIT R0,R5 ;CLEAR LINE FLAG.
7654 DEC R2 ;DECREMENT THE LINE NUMBER COUNTER.
7655 BNE 128 ;LOOP TO GET THE NEXT CHARACTER.
7656 TST R5 ;CHECK IF ANY BREAKS NOT RECEIVED.
7657 BEQ 601 ;EXIT TEST IF ALL CLEAR.
7658 MOV @EM6402,R1 ;SELECT MESSAGE TO BE PRINTED.
7659 REPORT ERROR:"BREAK NOT RECEIVED ON LINE ###".
7660 ERRDF 6406,EM6401,ER6401 ERROR 6407
7661 TRAP CIEROF
7662 L.WORD 6405
7663 L.WORD EM6401
7664 L.WORD ER6401
7665 BR 601 ;EXIT THE TEST.
7666 JABORT THE TEST.
7667 MOV CTRL# ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7668 ENDTST
7669 L10046: TRAP CIERTST
HARDWARE TEST

SOFTWARE TST.

SBTIL HARDWARE TEST - NOERRR.

- NO OVERRUN ERROR TEST -

THIS TEST VERIFIES THAT THE DUT WILL NOT REPORT DATA OVERRUN ERRORS WHEN THEY DO NOT OCCUR.

THIS TEST PUTS 256 CHARACTERS IN THE OUT FIFO PLUS 4 IN EACH ACTIVE UART AND VERIFIES THAT NO OVERRUN ERRORS ARE REPORTED.

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.

BGNSTI

SETPRI #PRIOS.RO, ALLOW LTC INTERRUPTS.

MOV #PRIOS.RO, WEEK

T18:

TNUM = TNUM + 1; INCREMENT THE ASSEMBLY TIME TEST COUNTER.

MOV #TNUM, #TNUM; SET UP THE TEST NUMBER. (66)

MOV 0, 1; CTRLCF - INDICATE THAT WE ARE IN A TEST.

MOV #ERRYP; SET ERROR TYPE AS FATAL IN ERROR TABLE.

MOV #6601, #ERRBR; SET ERROR NUMBER TO 6601.

MOV #EM6601, ERRMSG; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.

- RELOAD 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 >>>>> 6601 <<<<.

- 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.

- RELOAD 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 >>>>> 6601 <<<<.
7721 033546 012703 000400
7722 033552 004767 160622
7723 033556 103515
7724 MOV @BUFHD BUFFAS.R3, ;PASS THE LENGTH OF THE DATA PATTERN.
7725 JSR PC,DDRMA ;TRANSMIT THE DATA PATTERN.
7726 BCC 501 ;EXIT IF ERROR FOUND DURING DMA TX.
7727
7728 033560 005267 150266
7729 033564 012701 170536
7730 033570 016702 146432
7731 033574 004767 163366
7732 033580 103516
7733 033602 012704 000005
7734 033606 004767 160450
7735 INC ERRBR ;SET ERROR NUMBER TO 6603.
7736 MOV $170536,R1 ;PASS TIME-OUT VALUE OF 350 MILLI SECS.
7737 MOV CSRA,R2 ;PASS THE ADDRESS OF THE CSR.
7738 JSR PC,WAIAN ;WAIT FOR 0% TO COMPLETE, TX_ACTION SET.
7739 BCC 501 ;ABORT THE TEST IF TIME-OUT ON DMA COMPLETION.
7740 MOV $5,R4 ;PASS DELAY OF 5 Mlli SECS.
7741 JSR PC,DELAY ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
7742
7743 033612 016705 146402
7744 033616 012700 000204
7745 033622 004767 163724
7746 033626 012700 177670
7747 033632 004767 163744
7748 033636 012704 000012
7749 033642 004767 160414
7750 MOV ACTLNS,R5 ;ALTER PARAMETERS FOR ALL ACTIVE LINES.
7751 MOV #204,RO ;PASS PARAMETER FOR INTERNAL LOPBCK,ENABLE TX.
7752 JSR PC,WTU1.NC ;INITIALISE THE LINE CONTROL REGISTER.
7753 MOV $177670,R0 ;PASS THE LPR CONTENTS.
7754 JSR PC,WTU1PR ;SET THE LPR CONTENTS TO 38.4K BAUD.
7755 MOV $10,R4 ;PASS DELAY TIME OF 10 Mlli SECS.
7756 JSR PC,DELAY ;WAIT FOR LCTRL AND LPR_REGS TO BE UPDATED.
7757
7758 033646 012702 002650
7759 033652 012703 000004
7760 033656 005001
7761 033660 005267 150166
7762 033664 010100
7763 033666 006300
7764 033670 036067 002336 146322
7765 033676 001403
7766 033670 004767 160474
7767 033674 103104
7768 033670 005201
7769 033710 020127 00020
7770 033714 002763
7771 MOV @BUFHAS,R2 ;PASS THE START OF THE DATA PATTERN TO TX.
7772 MOV #4,R3 ;PASS THE LENGTH OF THE DATA PATTERN.
7773 CLR R1 ;CLEAR THE LINE COUNTER.
7774 INC ERRBR ;SET ERROR NUMBER TO 6604.
7775 2: MOV R1,R0 ;CALCULATE THE LINE OFFSET FROM THE LINE 0.
7776 ASL R0
7777 BIT BITBLC(R0),ACTLNS ;TEST FOR THIS LINE BEING ACTIVE.
7778 BEQ 4: ;SKIP THE TX ON THIS LINE IF IT IS NOT ACTIVE.
7779 JSR PC,DDRMA ;TRANSMIT THE 5 CHAR DATA PATTERN.
7780 BCC 501 ;ABORT IF ERROR FOUND DURING DMA TX.
7781 INC R1 ;INCREMENT THE LINE COUNTER.
7782 CMP R1,#NUMLNS ;TEST FOR ALL POSSIBLE LINES HANDLED.
7783 9LT 2: ;LOOP IF NOT ALL LINES HANDLED.
7784
7785 033716 005267 150130
7786 033722 012701 170040
7787 033726 016702 146274
7788 033732 004767 163500
7789 033740 103104
7790 033744 004767 160312
7791 MOV ERRBR ;SET ERROR NUMBER TO 6605.
7792 MOV $170040,R1 ;PASS TIME-OUT VALUE OF 32 MILLI SECS.
7793 MOV CSRA,R2 ;PASS THE ADDRESS OF THE CSR.
7794 JSR PC,WAIAN ;WAIT FOR A DMA TO COMPLETE, TX_ACTION SET.
7795 BCC 501 ;ABORT THE TEST IF TIME-OUT ON DMA COMPLETION.
7796 MOV $5,R4 ;PASS DELAY OF 5 MILLI SECS.
7797 JSR PC,DELAY ;WAIT FOR LAST CHAR TO ARRIVE IN THE FIFO.
7798
7799 033750 016702 146244
7800 033754 004767 160716
7801 033760 006302
7802 033762 006302
7803 033764 012705 000400
7804 033770 060205
7805 MOV ACTLNS,R2 ;GET THE NUMBER OF ACTIVE LINES.
7806 JSR PC,HAPCNT ;MULTIPLY NUMBER OF ACTIVE LINES BY 4.
7807 ASL R2
7808 ASL R2
7809 MOV $256,R3
7810 ADD R2,R5 ;CALCULATE NUMBER OF CHARACTERS TO RX.
CLR R4
; CLEAR THE CHARACTER COUNTER.

MOV #6606,.ERRNB
; SET UP ERROR NUMBER EACH TIME AROUND LOOP.

MOV #RBUF0,R2
; READ A CHARACTER FROM THE FIFO.

BPL 101
; EXIT THE READ LOOP IF THE FIFO IS EMPTY.

; CHECK IF THE READ CHARACTER IS A BMP CODE.

: IF IT IS A BMP CODE SAVE IT ON THE QUEUE TO BE REPORTED LATER, AND

ABORT THE TEST.

JSR PC,CkBmp
; CHECK IF CHARACTER IS A BMP CODE.

BCC 8:$
; BRANCH IF NOT A BMP CODE.

ERROR 8:
; >>>>> ERROR #6606 <<<<<.

BR 60:
; EXIT THIS TEST.

INC ERRNB
; SET ERROR NUMBER TO 6607.

INC R4
; COUNT THIS CHARACTER.

CMP R4,R5
; COMPARE # OF CHARS WITH MAX # OF CHARS.

BGT 50:
; ABORT TEST IF TOO MANY VALID CHAR READ.

BIT #BIT14,R2
; TEST THE OVERRUN BIT OF THE READ CHAR.

BEQ 6:
; LOOP TO READ THE NEXT CHAR IF NO ERROR.

MOV @ER7801,ERRBLK
; SELECT THE CORRECT ERROR REPORTING ROUTINE.

MOV @EN6602,R1
; PASS THE MESSAGE TO BE REPORTED.

MOV R2,R5

SWAP R3

BIC #17760,R3
; GET FAILING LINE NUMBER.

REPORT "OVERRUN ERROR REPORTED WHEN NONE FORCED, ON LINE NN ..."

ERROR 8:
; >>>>> ERROR #6608 <<<<<.

BR 60:
; EXIT THIS TEST.

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION
; EXIT WITH TEST FAILURE MESSAGE IF

BEQ 60:
; NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED.

; DURING THE SOFTWARE QUESTIONS.

BR 6:
; LOOP TO READ THE NEXT CHAR.

MOV #6609,.ERRNB
; SET ERROR NUMBER TO 6609.

CMP R4,R5
; COMPARE NUMBER OF CHAR READ WITH EXPECTED.

BEQ 60
; EXIT TEST WITHOUT ABORT IF CORRECT # OF CHAR.

JSR PC,TSABRT
; ABORT THE TEST, NON-RELATED TEST ERROR FOUND.

CLR CTRLCF
; INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDST

L10047:
; TRAP C;ESTT
**HARDWARE TEST**

**OVER-RUN ERROR TEST**

- **THIS TEST VERIFIES THAT THE OUT WILL REPORT DATA OVER-RUN ERRORS WHEN THEY OCCUR.**
- **THIS TEST PUTS 256 CHARACTERS IN THE OUT FIFO PLUS 5 IN EACH ACTIVE UART AND VERIFIES THAT OVER-RUN ERRORS ARE REPORTED ON ALL ACTIVE LINES.**
- **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. IT WILL BE REPORTED LATER.**

**BDMST**

**SET PRI #PRI05**

**ALLOW LTO INTERUPTS**

**MOV @PRIO5,RO**

**TRAP CSRPR**

**T19:**

**MOV @NUM,1**

**TMPL := TMPL + 1**

**INC THE ASSEMBLY TIME TEST COUNTER.**

**MOV @TMPL,ISTMPL**

**SET THE TEST NUMBER.**

**MOV @L,CTRCL**

**INDICATE THAT WE ARE IN A TEST.**

**MOV @L,ERRPR**

**SET ERROR TYPE AS FATAL IN ERROR TABLE.**

**MOV @EDT1,ERRMAR**

**SET ERROR NUMBER TO 6701.**

**MOV @EDT1,ERRMSG**

**SET ERROR MESSAGE ADDRESS IN ERROR TABLE.**

**RESET THE OUT TO A KNOWN STATE. REMOVE THE STATUS CODES FROM THE FIFO.**

**CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.**

**THIS SUBROUTINE REPORTS ERRORS >>>>> 6701 <<<<.**

**JSR PC,CLRST**

**RESET THE DMM-11, REPORT ANY ERRORS FOUND.**

**BCS .6**

**SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.**

**JMP 601**

**EXIT THE TEST, FATAL ERROR WAS FOUND.**

**.6**

**FIN D AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.**

**INITIALIZE THE 256 BYTE DATA PATTERN.**

**JSR PC,FINACT**

**FIN D AN ACTIVE LINE.**

**BCS .6**

**IF ACTIVE LINE IS FOUND, DON'T ABORT TEST.**

**JMP 601**

**ABORT THE TEST, NO ACTIVE LINES WERE FOUND.**

**JSR PC,INDATP**

**INITIALISE DATA PATTERN.**

**JSR PC,INAT**

**TRANS M I T A 263 CHARACTER DATA PATTERN USING DMA, ON A SINGLE CHANNEL**

**AT 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY, 2 STOP BITS.**

**.6**

**SET INTERNAL LOOPBACK ON THE SELECTED LINE.**

**TRANS M I T THE DATA PATTERN ON THE FIRST AVAILABLE ACTIVE LINE.**

**INC ERMR**

**SET ERROR NUMBER TO 6702.**

**MOV @204,RO**

**PASS PARAMETER FOR INTERNAL LOOPBACK, ENABLE RX.**

**JSR PC,HTCLN**

**INITIALISE THE LINE CONTROL REGISTER.**

**MOV @17670,RO**

**PASS THE LPR CONTENTS.**

**JSR PC,HTLP**

**SET THE LPR CONTENTS TO 38.4K BAUD.**

**MOV @10,RA**

**PASS DELAY TIME OF 10 MILLISEC SECONDS.**

**JSR PC,DELAY**

**WAIT FOR LINCTRL AND LPR REGS TO BE UPDATED.**

**MOV @BUF89,RO**

**PASS THE START OF THE DATA PATTERN 10 TX.**
6786 034302 005267 147544
6787 034301 012701 170536
6788 034312 016702 145170
6789 034316 004767 161314
6790 034324 000161 000532
6791 034330 012704 000005
6792 034334 004767 157722

; TRANSIT 5 CHARACTERS ON EACH ACTIVE LINE.

6794 034340 016705 145554
6795 034344 012700 000204
6796 034340 012700 163216
6797 034340 012700 157666
6798 034340 012700 000005
6799 034404 005001
6800 034406 005267 147440
6801 034412 010010
6802 034416 006300
6803 034416 036067 002336 145574
6804 034442 014005
6805 034442 004767 157746
6806 034442 013402
6807 034442 000167 000422
6808 034440 005201
6809 034442 020127 000020
6810 034446 002761

;/transmit the start of the data pattern to TX.

6812 034450 005267 147376
6813 034454 0170040,145542
6814 034454 0170040,162746
6815 034470 103174
6816 034472 012704 000005
6817 034476 004767 157560

;read 256 chars from the fifo checking for bmp codes.

6819 034502 012704 000000
6820 034506 012767 019562 147336
6821 034514 017002 145510
6822 034520 100160

; set up the character counter.

6824 034526 ,r4
6825 034606 ,r4
6826 034514 ,r2
6827 034520 ,r2

;set up error number each time around loop.

6828 034526 ,r4
6829 034606 ,r4
7935 034522 005267 147324 INC ERRNBR: SET ERROR NUMBER TO 6707.
7936 034526 004767 157244 JSR PC,CHKBMP: CHECK IF CHARACTER IS A BMP CODE.
7937 034532 103521 BCS 241: REPORT ERROR AND ABORT TEST IF A BMP CODE.
7938 034536 010351 INC R1: COUNT THIS CHARACTER.
7939 034536 001363 BNE 64: LOOP IF NOT 256 CHARS READ FROM FIFO.
7940 1 'READ THE REMAINING AND VERIFY 1 OVERRUN PLUS 1 CHAR FROM EACH LINE.
7941 1 'CLEAR THE OVERRUN ERROR FLAGS.
7942 1 'CLEAR THE OVERRUN ERROR FLAGS.
7943 034540 005004 CLR R4: CLEAR THE OVERRUN ERROR FLAGS.
7944 034542 012700 003710 MOV #RNCNTB,R0: CLEAR RX CHAR COUNT TABLE.
7945 034546 004767 157466 JSR PC,CLR16W: SET UP ERROR NUMBER EACH TIME AROUND LOOP.
7946 034576 015064 147272 B8: MOV #ertz 70, ERRNBR: READ A CHARACTER FROM THE FIFO.
7947 034580 017902 145444 MOV @RBPUAF,R2: GO ANALYZE THE RESULTS IF ALL CHARS READ.
7948 034586 100047 BPL 142: CHECK IF CHAR IS A BMP CODE.
7949 034586 004767 157204 JSR PC,CHKBMP: REPORT ERROR AND ABORT TEST IF A BMP CODE.
7950 034594 010351 INC R1: SET ERROR NUMBER TO 6709.
7951 034594 005267 147252 INC ERRNBR: SET ERROR NUMBER TO 6709.
7952 034600 010200 MOV R2,R0: REMOVE LINE NUMBER FROM THE READ CHAR.
7953 034602 003000 SWAB R0: REMOVE LINE NUMBER FROM THE READ CHAR.
7954 034604 042700 007760 BIC #177760,RO: CALCULATE THE LINE NUMBER OF THE CHAR.
7955 034606 006300 177760 ASCL R0: FORM WORD TABLE OFFSET FOR TABLE ACCESS.
7956 034612 042700 007400 BIC #7400,RO: REMOVE LINE NUMBER FROM THE READ CHAR.
7957 034616 036067 002356 145374 BEQ 504: ABORT TEST IF FOR INACTIVE LINE.
7958 034624 001516 BIT BITBL(R0),ACTLN: TEST FOR ACTIVE LINE.
7959 034626 005267 147220 INC ERRNBR: SET ERROR NUMBER TO 6710.
7960 034632 005760 003710 1ST RXCNTB(R0): CHECK THE RX CHAR COUNTER FOR THIS LINE.
7961 034636 001000 BNE 102: IS THIS CHAR A NULL?
7962 034640 020227 140000 CMP R2,#140000: YES, TEST FOR NULL CHAR WITH OVERRUN.
7963 034644 012121 008960 BEQ 128: IS CHAR A NULL?
7964 034646 056004 002356 12$: BIS BITBL(R0),R4: SET THE OVERRUN BIT ERROR FLAG FOR LINE.
7965 034652 000411 BR 128: GO COUNT THE CHAR AND CONTINUE.
7966 034654 026027 003710 000004 CMP RXCNTB(R0),#4: 5TH CHAR ON THIS LINE? YES, ABORT.
7967 034662 000277 000000 10$: CMP RXCNTB(R0),#4: 5TH CHAR ON THIS LINE? YES, ABORT.
7968 034664 032702 004000 BIT #BIT14,R2: NO, CHECK OVERRUN BIT.
7969 034670 001402 BEQ 128: IS OVERRUN BIT CLEAR? YES, GO COUNT CHAR.
7970 034672 056004 002356 12$: BIS BITBL(R0),R4: SET THE OVERRUN BIT ERROR FLAG FOR LINE.
7971 034676 005260 003710 12$: INC RXCNTB(R0): COUNT THIS CHARACTER.
7972 034676 000723 BR 04: LOOP UNTIL ALL CHARs ARE READ FROM FIFO.
7973 1 'TEST FOR ABORT CONDITIONS. ONLY NONE ABORT CONDITIONS ARE:
7974 1 '1) 2 CHARs RXED ON A LlNE AND NO OVERRUN ERROR BIT FAILURE DETECTED.
7975 1 '2) 2 TO 4 CHARs RXED ON A LlNE AND AN OVERRUN BIT FAILURE DETECTED.
7976 1 '2 TO 4 CHARs RXED ON A LlNE AND AN OVERRUN BIT FAILURE DETECTED.
7977 1 '1) 2 CHARs RXED ON A LlNE AND NO OVERRUN ERROR BIT FAILURE DETECTED.
7978 034704 005001 CLR R1: INITIALIZE LINE LOOP, CLEAR LINE OFFSET.
7979 034706 012767 015067 147136 MOV #6711,.ERRNBR: SET UP ERROR NUMBER EACH TIME AROUND LOOP.
7980 034714 002356 145276 BIT BITBL(R1),ACTLN: LINE ACTIVE? NO, NEXT LINE.
7981 034722 001145 BEQ 18$: LINE ACTIVE? NO, NEXT LINE.
7982 034724 026127 003710 000002 CMP RXCNTB(R1),#2: YES.
7983 034732 002453 BLT 504: FEWER THAN 2 CHARs RXED? YES, ABORT.
7984 034736 002356 141040 BIT BITBL(R1),R4: NO.
7985 034740 010006 BNE 18$: OVERRUN BIT ERROR FLAG SET? YES, NEXT LINE.
7986 034744 005267 147104 INC ERRNBR: SET LINE NUMBER TO 6712.
7987 034746 026127 003710 000002 CMP RXCNTB(R1),#2: NOT 2 CHARs RXED? YES, ABORT. NO, NEXT LINE.
7988 034754 001042 BNE 504: SET LINE OFFSET TO THE NEXT LINE.
7989 034756 062791 000002 ADD R2,R1: ALL LINES DONE? NO, LOOP. YES, CONTINUE.
7990 034762 020127 000040 CMP R1,NUMLNS+2:
7991 034766 002747 BLT 16$:
CHECK FOR OVERRUN ERROR BIT FAILURES, PRINT ERROR MESSAGE IF FOUND.

MOV $6713,.ERRADDR
CLR R1
MOV R1,R2
BIT BITBBL(R1),R4
MOV R1,R3
ASR R3
MOV $87801,.ERRBLK
MOV $876702,R1
DIV R2
BR 20:
REPORT "OVERRUN ERROR NOT REPORTED CORRECTLY WHEN FORCED. ON LINE NNO ...
ERROR
>>>ERROR 06713 <<<
TRAP C_ERROR

EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED

BIT #BIT06,OPTION
BEQ 60:
EXIT WITH TEST FAILURE MESSAGE IF
NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
DURING THE SOFTWARE QUESTIONS.

MOV R2,R1
RESTORE THE LINE OFFSET THAT WAS DESTROYED.

BIC BITBBL(R1),R4
CLEAR THE LINE ERROR FLAG WE JUST HANDLED.
ADD R2,R1
NO, INCREMENT THE LINE OFFSET.
BR 20:
LOOP TO HANDLE THE NEXT LINE.

REPORT "BMP CODE FOUND IN FIFO. TEST INVALIDATED."
ERROR
>>>ERROR <<<
TRAP C_ERROR

EXIT THIS TEST.

ENDTST
L10050:
TRAP CIENTST
.SBTTL HARDWARE TEST
.REPBMP

;** .................................................................
;* - 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.
;* .................................................................
;** .................................................................

BGNTST

TNUM -- TNUM + 1 ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV @TNUM,TSTNUM ; SET UP THE TEST NUMBER.

MOV 0-1,CTRLCF ; INDICATE THAT WE ARE IN A TEST.

MOV BMPCO8,R2 ; GET THE CONTENTS OF THE POINTER.
MOV BMPCO8,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 N\nERRDF 9301,EM9301,ER9301 ; ERRNO 9301 <<<<

TRAP CIERDF
.WORD 9301
.WORD EM9301
.WORD ER9301

MOV @BMPCO8,BMPCO8 ; SET POINTER BACK TO THE BEGINNING OF THE QUEUE.

CLR CTRLCF ; INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDST

L10051: TRAP CIEST

8029
8030
8031
8032
8033
8034
8035
8036
8037
8038
8039 035074
8040 035074 000024
8041 035074 012767 000024 145152
8042 035102 012-67 177777 145140
8043 035110 016702 145274
8044 035114 01703 002412
8045 035120 020203
8046 035122 001411
8047
8048
8049
8050
8051
8052 035124 012701 011354
8053 035130
8054
8055 035140 012767 002412 145242
8056
8057 035146 005067 145076 60$:
8058 035152
8059 035152
8060 035152 104401
SBITL  HARDWARE PARAMETER CODING SECTION

; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
; WITH THE OPERATOR.

BGNHRD

ENDHRD

.DEVICE CSR ADDRESS QUESTION:
    GPRMA  HWPTQ1,0,0,160000,177776,YES

.DEVICE INTERRUPT VECTOR QUESTION:
    GPRMA  HWPTQ2,2,0,40,776,YES

.ACTIVE LINES BIT MAP QUESTION:
    GPRMD  HWPTQ3,4,0,MAPNLS,0,177777,YES

.TYPE OF LOOPBACK QUESTION:
    GPRMD  HWPTQ4,6,0,377,1,2,YES

.HWPTQ1: .ASCIZ /CSR ADDRESS /

.L10052:  L1HRD/2
HWPTQ2: .ASCIZ /interrupt vector address:

8113 035240 111 116 124
035243 105 122 124
035246 125 120 124
035251 040 126 105
035254 103 124 117
035257 122 040 101
035262 104 104 122
035265 105 123 123
035270 072 040 000

HWPTQ3: .ASCIZ /active line bit map:

8114 035273 101 103 124
035276 111 126 105
035301 040 114 111
035304 116 105 040
035307 102 111 124
035312 040 115 101
035315 120 072 040
035320 000

HWPTQ4: .ASCIZ /type of loopback (1=internal, 2=m3229 or m3277):

8115 035321 124 131 120
035324 105 040 117
035327 106 040 114
035332 117 117 120
035335 102 101 103
035340 113 040 050
035343 061 075 111
035346 116 124 105
035351 122 116 101
035354 114 054 040
035357 062 075 110
035362 063 060 062
035365 071 040 117
035370 122 040 110
035373 063 062 067
035376 067 051 072
035401 000

8116
8117 .EVEN
THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS WITH THE OPERATOR.

UNIT NUMBER PRINTOUT QUESTION:
GPRML SWPTQ1.0,20,YES

EXTENDED ERROR REPORTING QUESTION:
GPRML SWPTQ2.0,100,YES

IF EXTENDED ERROR REPORTING IS NOT REQUIRED THEN SKIP THE NEXT QUESTION.
XFERF ENDD

NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE QUESTION:
GPRMD SWPTQ3.2,0,177777,0,177777,YES

NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE QUESTION:
GPRMD SWPTQ3.2,0,177777,0,177777,YES

ENDD: ENDSFT

UNIT NUMBER PRINTOUT AS EACH UNIT IS TESTED:

SWPTQ1: ASCIZ
SWPTQ2: .ASCIZ /EXTENDED ERROR REPORTING: /

SWPTQ3: .ASCIZ /NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE: /

.EVEN
8184
8185
8186 035632
8187 035632
8188
8189
8190
8191
8192 035702
8193
8194 LASTAD
035702 000000
035704 000000
035706
8200 035706
8201
8202
8203
8204
8205
8206
8207 8208 000001
8209

PATCH:
.BLKW 24

.EVEN
.WORD 0
.WORD 0
LILAST:
ENDMOD

.END
VIRTUAL MEMORY USED: 28661 WORDS (112 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:05:03
CZMHVBO.BIN,CZMHVBO.LST/-SP=SVC5AR/ML,CZMHVBO.P11