

.REM &

IDENTIFICATION
.....

PRODUCT CODE: AC-T800A-MC
PRODUCT NAME: CZDHXAO DHU-11 FUNC TST PART4
PRODUCT DATE: 3 MARCH 1984
MAINTAINER: ENE - DIAGNOSTICS GROUP
AUTHOR: ANTHONY HART
MODIFIED BY:

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSIDERED 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	VIBUS	MASSBUS
DEC	DECUS	DECWARE	

C1

***** MODIFICATION HISTORY *****

ORIGINAL RELEASE: 3-MAR-1984 ANTHONY HART

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 ZFLAGS 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

CZDMXAO IS PART OF THE DHU-11 FUNCTIONAL VERIFICATION TEST. THIS PART OF THE TEST PERFORMS EXTENSIVE DATA TRANSMISSION AND RECEPTION TESTS. THIS PART ALSO INCLUDES A KEYBOARD ECHO AND MODEM LOOPBACK TEST.

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

1.2 SYSTEM REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE DHU-11 FVT:

- O UNIBUS PROCESSOR WITH AT LEAST 32K BYTES OF MEMORY.
- O DHU BOARDS INSTALLED ON THE UNIBUS.
- O APPROPRIATE PROGRAM LOAD DEVICE SUPPORTING XXDP+ MEDIA OR A DOWN LINE LOADING SYSTEM.

1.3 RELATED DOCUMENTS AND STANDARDS

- O XXDP+ USER'S MANUAL DESCRIBES THE RUNNING OF DIAGNOSTICS UNDER THE XXDP+ 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 XXDP+ USER'S MANUAL (CHQUS).

2.1 COMMANDS

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

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

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

2.2 SWITCHES

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

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

EXAMPLE OF SWITCH USAGE:

STAR*/TESTS:1-5/PASS:1000/EOP:100

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

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

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

2.3 FLAGS

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

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

*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>/EOP;<INCR>  
*****
```

2.4.1.1 TESTS SWITCH (/TESTS;<TEST-LIST>) -

<TEST-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (1:2 ETC.) OR RANGES OF DECIMAL NUMBERS (1-5:8-10 ETC.), SEPERATED BY COLONS, THAT SPECIFY THE TESTS TO BE EXECUTED. TESTS WILL BE EXECUTED IN NUMERICAL ORDER REGARDLESS OF THE ORDER OF SPECIFICATION. THE DEFAULT IS TO EXECUTE ALL TESTS. ON THIS AND ALL SWITCHES, THE ANGLE BRACKETS <> ARE PUNCTUATION USED IN THE DEFINITION ONLY, AND ARE NOT TO BE TYPED BY THE OPERATOR. SEE EXAMPLE AT END OF "EFFECT OF START COMMAND" SECTION.

2.4.1.2 PASS SWITCH (/PASS;<PASS-CNT>) -

<PASS-CNT> IS A DECIMAL NUMBER INDICATING THE DESIRED NUMBER OF PASSES. A PASS IS DEFINED AS THE EXECUTION OF THE FULL DIAGNOSTIC (ALL SELECTED TESTS). THE DEFAULT IS NON-ENDING EXECUTION. IN THIS CASE, EXIT FROM THE PROGRAM IS ACCOMPLISHED EITHER BY TYPING A CONTROL/C OR BY OCCURANCE OF AN ERROR WITH THE HALT ON ERROR FLAG BEING SET. THE EXIT IS A RETURN TO COMMAND MODE. SEE EXAMPLE AT END OF "EFFECT OF START COMMAND" SECTION.

2.4.1.3 FLAGS SWITCH (/FLAGS;<FLAG-LIST>) -

<FLAG-LIST> IS A SEQUENCE OF ELEMENTS OF THE FORM <FLAG>, <FLAG=1>, OR <FLAG=0>, SEPERATED BY COLONS, WHERE <FLAG> HAS ONE OF THE FOLLOWING VALUES:

MOE	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.
PPI	DIRECT ALL MESSAGES TO A LINE PRINTER.
PNT	PRINT NUMBER OF TEST BEING EXECUTED.
BOE	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/FLAG:TER:HOE=1

THIS COMMAND WILL CAUSE THREE PASSES TO BE MADE, WITH EACH PASS
CONSISTING OF TESTS 1, 3, AND 4. THERE IS NO DIFFERENCE BETWEEN SAYING
<FLAG> AND SAYING <FLAG=1>. THE NOTATION <FLAG=0> IS MEANINGFUL ONLY
ON A COMMAND OTHER THAN START TO CLEAR A FLAG THAT WAS PREVIOUSLY SET.
NOTE THAT ON ALL COMMANDS ONLY THE FIRST THREE LETTERS ARE SCANNED.

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 HALT ON ERROR FLAG SET, OR C) A CONTROL/C WAS ENTERED BY THE OPERATOR.

2.4.3 CONTINUE COMMAND

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

2.4.3.1 FLAG SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS SAME AS IN THE START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

2.4.3.2 EFFECT OF CONTINUE COMMAND -

CONTINUE MUST FOLLOW A START OR RESTART, AND COMMAND MODE MUST HAVE BEEN ENTERED DUE TO A HALT ON ERROR OR A CONTROL/C. THE EFFECT OF THE COMMAND IS TO GO TO THE BEGINNING OF THE TEST THAT WAS BEING EXECUTED WHEN THE HALT OR CONTROL/C TOOK PLACE. SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

2.4.4 PROCEED COMMAND -

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

2.4.4.1 FLAGS SWITCH (/FLAGS:<FLAG-LIST>) -

<FLAG-LIST> IS AS IN THE START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

2.4.4.2 EFFECT OF PROCEED COMMAND -

PROCEED MUST FOLLOW A START, RESTART, OR CONTINUE. COMMAND MODE MUST HAVE BEEN ENTERED VIA A HALT ON ERROR. THE EFFECT OF THE COMMAND IS TO BEGIN EXECUTION AT THE LOCATION FOLLOWING THE ERROR CALL. NEITHER HARDWARE NOR SOFTWARE PARAMETERS MAY BE ALTERED.

2.4.5 ADD COMMAND -

ADD/UNITS:<UNIT-LIST>

2.4.6 EFFECT OF ADD COMMAND -

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

2.4.7 DROP COMMAND -

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

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

2.4.9 PRINT COMMAND -

PRINT)

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 -

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

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 -

FLA(GS)

2.4.11.1 EFFECT OF FLAGS COMMAND -
THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.

2.4.12 ZFLAGS COMMAND -

ZFL(ABS)

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 F-TABLE DIALOGUE OR SOFTWARE P-TABLE DIALOGUE CAUSES THE DEFAULTS TO BE TAKEN FOR THE REMAINDER OF THAT DIALOGUE.

- O A CONTROL/O (O) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES ALL TELETYPE OUTPUT TO BE SUPPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR UNTIL ANOTHER CONTROL/O IS TYPED, WHICH RESTORES NORMAL TELETYPE OUTPUT.

2.5 HARDWARE QUESTIONS

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

1. CSR ADDRESS - THIS QUESTION REQUESTS THE CSR ADDRESS OF THE SPECIFIED DHU-11. THE DEFAULT ANSWER FOR THIS QUESTION IS ADDRESS 160460 (OCTAL).
2. INTERRUPT VECTOR ADDRESS - THIS QUESTION REQUESTS THE INTERRUPT VECTOR ADDRESS OF THE SPECIFIED DHU-11. THE DEFAULT ANSWER FOR THIS QUESTION IS 310 (OCTAL).
3. ACTIVE LINES BIT MAP - THIS QUESTION REQUESTS AN OCTAL BIT MAP OF THE SERIAL COMMUNICATION LINES ON THE DHU11 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. THE DEFAULT ANSWER FOR THIS QUESTION IS ALL LINES I.E. 177777.
4. TYPE OF LOOPBACK (1=INTERNAL, 2=H3029 OR H3277, 3=H325
4=MODEM, 5=KEYBOARD ECHO).
THIS QUESTION REQUESTS THE TYPE OF LOOPBACK TO BE USED WHEN TESTING THE DHU-11.
THE FOLLOWING TYPES ARE SUPPORTED:
 - 0 INTERNAL - ONLY INTERNAL UART LOOPBACK IS TO BE USED IN TESTING THE DHU-11.
 - 0 H3029 OR H3277 - STAGGERED LOOPBACK CONNECTORS ARE PROVIDED ON THE DHU11 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 DHU11.
 - 0 H325 - SINGLE LINE, 25 PIN LOOPBACK CONNECTORS (TYPE H325) ARE INSTALLED ON THE LINES TO BE TESTED.
 - 0 MODEM - THE OPERATOR IS ALLOWED TO SET UP A MODEM LINK AND THEN PERFORM A TRANSMISSION AND RECEPTION TEST AT A SINGLE BAUDRATE WITH THE MODEM CONTROL SIGNALS DTR AND RTS ACTIVE. THIS IS A SPECIAL TEST AND ALL OTHER TESTS IN THIS PART WILL BE PERFORMED IN INTERNAL LOOPBACK.
 - 0 KEYBOARD ECHO - THE UARTS ON THE DUT ARE PLACED IN REMOTE LOOPBACK. TERMINALS (OR OTHER COMMUNICATIONS EQUIPMENT) WILL HAVE WHATEVER THEY TRANSMIT TO THE DHU LOOPED BACK TO THEM.

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. REPORT NUMBER OF BITS TESTED IN DMA ADDR TEST - THIS QUESTION ASKS WHETHER THE OPERATOR WANTS A PRINTOUT DESCRIBING WHICH ADDRESS BITS HAVE BEEN TESTED WHEN THE DMA ADDRESSING TEST EXECUTES.
3. 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.
4. 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 REPEATING.

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

♦ UNITS (D) ? 8<CR>

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

UNIT 2
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 1<CR>
Q-FACTOR (O) 1 ? 0<CR>

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

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

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

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

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


```
UNIT 8  
CSR ADDRESS (0) 160000<CR>  
SUB-DEVICE # (0) ? 7<CR>  
Q-FACTOR (0) 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 (0) ? 8<CR>
```

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

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

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

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

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

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

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

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

2.8 QUICK START-UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXDP+
2. GIVE THE DATE AND ANSWER THE LSI/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 "IXR" FLAGS ARE SET (SEE THE FLAGS SECTION OF THIS DOCUMENT). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

3.2 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 NEED 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 SUMMARIES SECTION OF THIS DOCUMENT. AN EXAMPLE OF SUCH A AN ERROR MESSAGE IS THE FOLLOWING:

```
CZDHX DVC FTL ERR 4409 ON UNIT 00 TST 04 SUB 000 PC: XXXXXX  
DMA ADDRESS 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:

```
CZDHX DVC FTL ERR 4409 ON UNIT 00 TST 04 SUB 000 PC: XXXXXX  
DMA ADDRESS TEST FAILED  
BAD BITS BETWEEN BITS 0 AND 15.
```

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 FUTHER INFORMATION SEE THE SWITCHES SECTION OF THIS DOCUMENT.

5.0 TEST SUMMARIES

THE FOLLOWING ARE INCLUDED WITHIN CZDHXA:

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. KEYBOARD ECHO TEST - ALLOWS THE OPERATOR TO TEST TERMINAL LINKS (OR OTHER COMMUNICATIONS LINKS), WHICH ARE ATTACHED TO UUT SERIAL PORTS, FROM REMOTE ENDS OF THE LINKS.
3. MODEM LOOPBACK TEST - ALLOWS THE OPERATOR TO TEST MODEM LINKS WHICH ARE ATTACHED TO THE UUT SERIAL PORTS.
4. DMA ADDR TEST - VERIFIES THAT THE UUT CAN ACCESS THE FULL MEMORY WHICH IS ON THE MACHINE VIA DMA ACCESS.
5. FRAMING ERROR TEST - VERIFIES THAT FORCED FRAMING ERRORS ARE REPORTED CORRECTLY.
6. PARITY ERROR TEST - VERIFIES THAT FORCED PARITY ERRORS ARE REPORTED CORRECTLY.
7. DMA MODE TEST - VERIFIES THAT THE UUT WILL TX AND RX DATA CORRECTLY USING DMA TRANSMISSION.
8. SPLIT SPEED TEST - VERIFIES THAT THE UUT WILL FUNCTION CORRECTLY USING DIFFERENT TX AND RX SPEEDS ON EACH ACTIVE LINE.
9. 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 CZDHXAO
CZDHXAO.BIN
DRS
CZDHX-A-0
DHU-11 FUNC TST PART4
UNIT IS DHU-11
RESTR1 ADDR: 147670
DR>STA/PAS:1

CHANGE HW (L) ? Y

* UNITS (D) ? 2

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

UNIT 1
CSR ADDRESS: (0) 160460 ? 160500
INTERRUPT VECTOR ADDRESS: (0) 310 ? 320
ACTIVE LINE BIT MAP: (0) 177777 ? <CR>
TYPE OF LOOPBACK (1=INTERNAL, 2=H3029 OR H3277, 3=H325
4=MODEM, 5=KEYBOARD ECHO): (0) 2 ? 1

CHANGE SW (L) ? Y

REPORT UNIT NUMBER AS EACH UNIT IS TESTED: (L) Y ? <CR>
REPORT NUMBER OF BITS TESTED IN DMA ADDR TEST: (L) N ? <CR>
EXTENDED ERROR REPORTING: (L) N ? <CR>

TESTING UNIT : 0

TESTING UNIT : 1

CZDHX EOP 1
0 TOTAL ERRS

DR>

```

1016          .LIST SEQ,LOC,BIN,MEB
1017          .NLIST CND
1025
1026
1027          .SBTTL PROGRAM HEADER
1028
1029
1030          .MCALL SVC
1031 000000    SVC          ; INITIALIZE SUPERVISOR MACROS
1032
1033          ;*****
1034          ; IF STRUCTURED MACROS ARE TO BE USED, ADD ".MCALL STRUCT" AND "STRUCT"
1035          ; TO INITIALIZE THE STRUCTURED MACROS.
1036
1037          SVCINS= 1      ; LIST INSTRUCTIONS, SHIFTED RIGHT
1038          SVCTST= 1     ; LIST TEST TAGS, SHIFTED RIGHT
1039          SVCSUB= 1     ; LIST SUBTEST TAGS, SHIFTED RIGHT
1040          SVCGBL= 1    ; LIST GLOBAL TAGS, SHIFTED RIGHT
1041          SVCTAG= 1    ; LIST OTHER TAGS, SHIFTED RIGHT
1042
1043          ; CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH
1044          ; TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE
1045          ; SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY
1046          ; CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.
1047          ;*****
1048
1049 000000    .ENABL ABS
1050          ;.ENABL AMA
1051          *          2000
1052
1053 002000    BGNMOD
1054
1055
1056          ;**
1057          ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
1058          ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
1059          ;--
1060 002000    POINTER BGNRPT,BGNSW,BGNSFT,BGNDU,ERRTBL
1061
1062
1063
1064
1065
1066
1067
1068
1069 002000    HEADER CZDHX,A,0,200,0,PRI07
1070
1071 002000
1072 002000    103
1073 002001    132
1074 002002    104
1075 002003    110
1076 002004    130
1077 002005    000
1078 002006    000
1079 002007    000
1080 002010
1081 002010    101
1082 002011
1083 002011    060
1084 002012
1085 002012    000000
1086 002014

```

```

L$NAME::
        .ASCII /C/
        .ASCII /I/
        .ASCII /D/
        .ASCII /H/
        .ASCII /X/
        .BYTE 0
        .BYTE 0
        .BYTE 0
L$REV::
        .ASCII 'A'
L$DEPO::
        .ASCII 0
L$UNIT::
        .WORD 0
L$TIML::

```

002014 000200
002016
002016 036620
002020
002020 037136
002022
002022 002150
002024
002024 002162
002026
002026 037526
002030
002030 000000
002032
002032 000000
002034
002034 000000
002036
002036 000000
002040
002040 002124
002042
002042 000340
002044
002044 000000
002046
002046 000000
002050
002050 003
002051 003
002052
002052 000000
002054 000000
002056
002056 000000
002060
002060 005364
002062
002062 027560
002064
002064 000000
002066
002066 000000
002070
002070 000000
002072
002072 030430
002074
002074 000000
002076
002076 005374
002100
002100 104035
002102
002102 005314
002104
002104 027574

L\$HPCP: .WORD 200
L\$SPCP: .WORD L\$HARD
L\$HPTP: .WORD L\$SOFT
L\$SPTP: .WORD L\$HW
L\$LADP: .WORD L\$SW
L\$STA: .WORD L\$LAST
L\$CO: .WORD 0
L\$DTYP: .WORD 0
L\$APT: .WORD 0
L\$DTP: .WORD 0
L\$PRIO: .WORD L\$DISPATCH
L\$ENVI: .WORD PRI07
L\$EXP1: .WORD 0
L\$MREV: .WORD 0
L\$EF: .BYTE C\$REVISION
.BYTE C\$EDIT
L\$SPC: .WORD 0
L\$DEVP: .WORD 0
L\$REPP: .WORD L\$DVTYP
L\$EXP4: .WORD L\$RPT
L\$EXP5: .WORD 0
L\$AUT: .WORD 0
L\$DUT: .WORD L\$DU
L\$LUN: .WORD 0
L\$DESP: .WORD L\$DESC
L\$LOAD: EMT E\$LOAD
L\$FTP: .WORD L\$ERRTBL
L\$ICP: .WORD L\$INIT

002106
002106 030412
002110
002110 030410
002112
002112 027566
002114
002114 000000
002116
002116 000000
002120
002120 000000
1080

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

1092
1093
1094
1095
1096
1097
1098
1099

.SBTTL DISPATCH TABLE

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

DISPATCH 9

002122
002122 000011
002124
002124 030546
002126 031030
002130 031272
002132 032224
002134 033710
002136 034324
002140 035002
002142 035770
002144 036536

.WORD 9
L\$DISPATCH:
.WORD T1
.WORD T2
.WORD T3
.WORD T4
.WORD T5
.WORD T6
.WORD T7
.WORD T8
.WORD T9

1100

1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125

.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.

1126 002146
002146 000004
002150
002150

BGNHW DFPTBL

.WORD L10000-L1HW/2
L1HW::
DFPTBL::

1127
1128 002150 160460
1129 002152 000310
1130 002154 177777
1131 002156 002
1132
1133 002160
002160

.WORD 160460 ;DEFAULT CSR ADDRESS
.WORD 310 ;DEFAULT VECTOR ADDRESS
.WORD 177777 ;DEFAULT ACTIVE LINES BIT MAP
.BYTE 2 ;DEFAULT LOOPBACK MODE
.EVEN
ENDHW

L10000:

1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158

002160
002160 000002
002162
002162
002162
000020
000000
002166
002166

.SBTTL 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

.WORD L10001-L\$SW/2
L\$SW::
SFPTBL::

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

ENDSW

LJ0001:

1167
 1168
 1169
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1222 002166

000020
 177777

 000000
 000002
 000002
 000004
 000006
 000006
 000010
 000012
 000014
 000016

 000020
 000030
 000100

 100000
 040000
 020000
 010000
 004000
 002000
 001000
 000400
 000200
 000100
 000040
 000020
 000010
 000004
 000002
 000001

 001000
 000400
 000200
 000100

.SBTTL GLOBAL EQUATES SECTION

 ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
 ; ARE USED IN MORE THAN ONE TEST.
 ;--

NUMLNS**20 ;NUMBER OF LINES ON DHV11 IS 8.
 MAPLNS**177777 ;BIT MAP OF LINES ON DHV11.

***** DEVICE REGISTER OFFSETS FROM THE CSR'S ADDRESS *****
 CSRO**0 ;CSR REGISTER OFFSET FROM THE CSR ADDRESS
 RBUFO**2 ;RECEIVE REGISTER OFFSET FROM THE CSR ADDRESS
 RXTIMO**2 ;RECIEVE TIMER REGISTER OFFSET FROM THE CSR ADDRESS
 LPRO**4 ;LINE PARAMETER REGISTER OFFSET FROM THE CSR ADDRESS
 FLSO**6 ;FIFOSIZE/STATUS REGISTER OFFSET FROM THE CSR ADDRESS
 FDATA**6 ;FIFO DATA REGISTER OFFSET FROM THE CSR ADDRESS
 LNCTRO**10 ;LINE CONTROL REGISTER OFFSET FROM THE CSR ADDRESS
 TXAD10**12 ;TRANSMIT ADDRESS 1 REGISTER OFFSET FROM THE CSR ADDRESS
 TXAD20**14 ;TRANSMIT ADDRESS 2 REGISTER OFFSET FROM THE CSR ADDRESS
 TXBFCO**16 ;TRANSMIT COUNT REGISTER OFFSET FROM THE CSR ADDRESS

***** EQUATES USED WITH RESPECT TO THE RX BUFFER *****
 RXBETX**16. ;LEVEL OF RX BUFFER AT WHICH TO RE-ENABLE TRANSMISSION.
 RXBDTX**24. ;LEVEL OF RX BUFFER AT WHICH TO DISABLE TRANSMISSION.
 RXBFUL**64. ;TOTAL CHARACTER CAPACITY OF THE RX BUFFER.

EQUATE

;
 ; BIT DEFINITIONS

;
 BIT15** 100000
 BIT14** 40000
 BIT13** 20000
 BIT12** 10000
 BIT11** 4000
 BIT10** 2000
 BIT09** 1000
 BIT08** 400
 BIT07** 200
 BIT06** 100
 BIT05** 40
 BIT04** 20
 BIT03** 10
 BIT02** 4
 BIT01** 2
 BIT00** 1
 ;
 BIT9** BIT09
 BIT8** BIT08
 BIT7** BIT07
 BIT6** BIT06

```

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

```

1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245 002166 000300
1246 002170 000304
1247 002172 000377
1248 002174 000
1249 002175 004
1250 002176 000000
1251
1252
1253
1254
1255
1256 002200
1257 002200 160020
1258 002202 160022
1259 002204 160024
1260 002206 160026
1261
1262 002210 160030
1263 002212 160032
1264 002214 160034
1265 002216 160036
1266
1267
1268
1269
1270 002220 000000
1271 002222 000000
1272 002224 000000
1273 002226 000000
1274 002230 000000
1275 002232 000000
1276 002234 000000
1277 002236 031463
1278 002240 146314
1279 002242 000000
1280 002244 000000
1281 002246 000000
1282 002250 000000
1283 002252 000000
1284 002254 000000
1285 002256 000000
1286 002260 000001
1287 002262 000000
1288 002264 000000

.SBTTL GLOBAL DATA SECTION

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

; UNIT VARIABLE AREA

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

; DEVICE REGISTER ADDRESS TABLE

DRADR::
CSRA:: .WORD 160020 ;DMU-11 CSR ADDRESS.
RXTMA:: RBUFA:: .WORD 160022 ;DMU-11 RECIEVE BUFFER/TIMER ADDRESS.
LPRA:: .WORD 160024 ;DMU-11 LINE PARAMETER REGISTER ADDRESS.
FDATA:: FLSA:: .WORD 160026 ;DMU-11 FIFO SIZE/LINE STATUS REGISTER ADDRESS,
;AND FIFO DATA REGISTER ADDRESS.
LNCTRA:: .WORD 160030 ;DMU-11 LINE CONTROL REGISTER ADDRESS.
TXAD1A:: .WORD 160032 ;DMU-11 TRANSMIT BUFFER 1 REGISTER ADDRESS
TXAD2A:: .WORD 160034 ;DMU-11 TRANSMIT BUFFER 2 REGISTER ADDRESS
TXBFCA:: .WORD 160036 ;DMU-11 TRANSMIT BUFFER COUNT REGISTER ADDRESS

; ASSORTED GLOBAL VARIABLES:

CTRLC:: .WORD 0 ;STORAGE FOR THE CONTROL-C FLAG.
DMTSTA:: .WORD 0 ;STO'G FOR DMA TEST ADDRESS (IN PAR FORM).
FERROR:: .WORD 0 ;STORAGE FOR "AT LEAST ONE ERROR" INDICATOR.
FFREM:: .WORD 0 ;STO'G FOR ADR OF FIRST FREE WORD AFTER THE DIAG' TIC
GMANWD:: .WORD 0 ;WORD FOR GMANXX CALL RETURN PARAMETERS.
IBM:: .WORD 0 ;INACTIVE TX/RX BITS MASK.
IESTAT:: .WORD 0 ;STORAGE FOR STATES OF THE OUT INT ENABLE BITS.
LGRP1M:: .WORD 31463 ;BIT MAP OF LINES IN LINE GROUP I.
LGRP2M:: .WORD 146314 ;BIT MAP OF LINES IN LINE GROUP II.
PASCNT:: .WORD 0 ;STO'G FOR PASS COUNT USED IN ROM VERSION# 151.
PMSFLG:: .WORD 0 ;FLAG INDICATING WHETHER TO PRINT MODEM STATUS.
RXTOUT:: .WORD 0 ;TIME-OUT VALUE FOR WAITING FOR LAST RX CHAR.
SAVPRI:: .WORD 0 ;STO'G FOR PROCESSOR PRIORITY. (TXROFF, TXRON).
SAVTEN:: .WORD 0 ;STORAGE FOR TX.ENABLE STATES. (TXROFF, TXRON).
TP4FLG:: .WORD 0 ;FLAGS SET WHEN AN EXPECTED 004 TRAP OCCURS.
TP4VEC:: .WORD 0 ;STORAGE FOR THE NORMAL 004 TRAP VECTOR.
TSTNUM:: .WORD 1 ;STORAGE FOR THE TEST NUMBER.
TXENBM:: .WORD 0 ;STORAGE FOR TX.ENABLE STATES. (BUFFER MGM NT).
TXINTF:: .WORD 0 ;STORAGE FOR TRANSMIT INTERRUPT FLAGS.

1289 002266 000000
 1290
 1291
 1292
 1293
 1294 002270 177546
 1295 002272 000300
 1296 002274 000100
 1297 002276 000074
 1298 002300 000000
 1299 002302 000000
 1300 002304 000170
 1301 002306 000170
 1302 002310 000021
 1303 002312 000062
 1304
 1305
 1306
 1307
 1308 002314 177572
 1309 002316 172516
 1310 002320 000000
 1311 002322 000000
 1312
 1313 002324
 1314 002324 172340
 1315 002326 172342
 1316 002330 172344
 1317 002332 172346
 1318 002334 172350
 1319 002336 172352
 1320 002340 172354
 1321 002342 172356
 1322 002344
 1323
 1324 002344
 1325 002344 172300
 1326 002346 172302
 1327 002350 172304
 1328 002352 172306
 1329 002354 172310
 1330 002356 172312
 1331 002360 172314
 1332 002362 172316
 1333 002364
 1334
 1335
 1336
 1337
 1338 002364 000001
 1339 002366 000002
 1340 002370 000004
 1341 002372 000010
 1342 002374 000020
 1343 002376 000040
 1344 002400 000100
 1345 002402 000200

```

WORD1:: .WORD 0 ;LOCATION FOR PASSING INDIRECT PARAMETERS.

;*****
; LINE TIME CLOCK VARIABLES AND STORAGE.
;*****
CLKCSR:: .WORD 177546 ;CSR ADDRESS OF THE LTC.
CLKBRI:: .WORD PRI06 ;INTERRUPT PRIORITY LEVEL OF THE LTC.
CLKVEC:: .WORD 100 ;INTERRUPT VECTOR ADDRESS OF THE LTC.
CLKHRZ:: .WORD 60. ;INTERRUPT FREQUENCY OF THE LTC.
TIMER1:: .WORD 0 ;HARDWARE CLOCK COUNTER #1.
TIMER2:: .WORD 0 ;HARDWARE CLOCK COUNTER #2.
TIMER3:: .WORD 120. ;HARDWARE BREAK COUNTER LOCATION.
BCOUNT:: .WORD 120. ;BREAK COUNT VALUE IN CLOCK TICKS.
MSTICK:: .WORD 17. ;NUMBER OF MILLI-SECONDS PER LTC TICK.
MSLCNT:: .WORD 62 ;LOOP COUNT (USED BY MSLOOP) TO DELAY 1 MS.

;*****
; MEMORY MANAGEMENT VARIABLES AND FLAGS.
;*****
MMSR0:: .WORD 177572 ;ADDRESS OF MEM MGT STATUS REGISTER #0.
MMSR3:: .WORD 172516 ;ADDRESS OF MEM MGT STATUS REGISTER #3.
MMPRES:: .WORD 0 ;MEM MGT PRESENT FLAG (0 IF MM NOT PRESENT).
MMENAB:: .WORD 0 ;MEM MGT ENABLED FLAG (0 IF MM NOT ENABLED).

PARATB:: ;BASE OF MEM MGT PAR ADDRESS TABLE.
PAR0A:: .WORD 172340 ;ADDRESS OF MEM MGT PAR #0.
PAR1A:: .WORD 172342 ;ADDRESS OF MEM MGT PAR #1.
PAR2A:: .WORD 172344 ;ADDRESS OF MEM MGT PAR #2.
PAR3A:: .WORD 172346 ;ADDRESS OF MEM MGT PAR #3.
PAR4A:: .WORD 172350 ;ADDRESS OF MEM MGT PAR #4.
PAR5A:: .WORD 172352 ;ADDRESS OF MEM MGT PAR #5.
PAR6A:: .WORD 172354 ;ADDRESS OF MEM MGT PAR #6.
PAR7A:: .WORD 172356 ;ADDRESS OF MEM MGT PAR #7.

PARATE:: ;END OF PAR ADDRESS TABLE.

PDRATB:: ;BASE OF MEM MGT PDR ADDRESS TABLE.
PDR0A:: .WORD 172300 ;ADDRESS OF MEM MGT PDR #0.
PDR1A:: .WORD 172302 ;ADDRESS OF MEM MGT PDR #1.
PDR2A:: .WORD 172304 ;ADDRESS OF MEM MGT PDR #2.
PDR3A:: .WORD 172306 ;ADDRESS OF MEM MGT PDR #3.
PDR4A:: .WORD 172310 ;ADDRESS OF MEM MGT PDR #4.
PDR5A:: .WORD 172312 ;ADDRESS OF MEM MGT PDR #5.
PDR6A:: .WORD 172314 ;ADDRESS OF MEM MGT PDR #6.
PDR7A:: .WORD 172316 ;ADDRESS OF MEM MGT PDR #7.

PDRATE:: ;END OF MEM MGT PDR ADDRESS TABLE.

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

```



```

1346 002404 000400      .WORD 400      ;BIT 8 SET.
1347 002406 001000      .WORD 1000     ;BIT 9 SET.
1348 002410 002000      .WORD 2000     ;BIT 10 SET.
1349 002412 004000      .WORD 4000     ;BIT 11 SET.
1350 002414 010000      .WORD 10000    ;BIT 12 SET.
1351 002416 020000      .WORD 20000    ;BIT 13 SET.
1352 002420 040000      .WORD 40000    ;BIT 14 SET.
1353 002422 100000      .WORD 100000   ;BIT 15 SET.
1354
1355
1356 ;*****
1357 ;*      TABLE OF DUT BAUDRATES
1358 ;*****
1358 002424      BRTBLB::      ;BASE OF DUT BAUD RATE TABLE.
1359 002424 000062      .WORD 50.      ;BAUD RATE ENTRY FOR CODE 0.
1360 002426 000113      .WORD 75.      ;BAUD RATE ENTRY FOR CODE 1.
1361 002430 000156      .WORD 110.     ;BAUD RATE ENTRY FOR CODE 2.
1362 002432 000206      .WORD 134.     ;BAUD RATE ENTRY FOR CODE 3.
1363 002434 000226      .WORD 150.     ;BAUD RATE ENTRY FOR CODE 4.
1364 002436 000454      .WORD 300.     ;BAUD RATE ENTRY FOR CODE 5.
1365 002440 001130      .WORD 600.     ;BAUD RATE ENTRY FOR CODE 6.
1366 002442 002260      .WORD 1200.    ;BAUD RATE ENTRY FOR CODE 7.
1367 002444 003410      .WORD 1800.    ;BAUD RATE ENTRY FOR CODE 8.
1368 002446 003720      .WORD 2000.    ;BAUD RATE ENTRY FOR CODE 9.
1369 002450 004540      .WORD 2400.    ;BAUD RATE ENTRY FOR CODE 10.
1370 002452 011300      .WORD 4800.    ;BAUD RATE ENTRY FOR CODE 11.
1371 002454 016040      .WORD 7200.    ;BAUD RATE ENTRY FOR CODE 12.
1372 002456 022600      .WORD 9600.    ;BAUD RATE ENTRY FOR CODE 13.
1373 002460 045400      .WORD 19200.   ;BAUD RATE ENTRY FOR CODE 14.
1374 002462 113000      .WORD 38400.   ;BAUD RATE ENTRY FOR CODE 15.
1375 002464      BRTBLE::      ;LABEL AFTER END OF DUT BAUDRATE TABLE.
1376 ;*****
1377 ;*      GPR SAVE AREAS ZERO AND ONE.
1378 ;*****
1379 002464      GPRS0B::      ;BASE OF GPR SAVE AREA NUMBER ZERO.
1380 002464 000000      .WORD 0        ;WORD 1, STORAGE FOR R1.
1381 002466 000000      .WORD 0        ;WORD 2, STORAGE FOR R2.
1382 002470 000000      .WORD 0        ;WORD 3, STORAGE FOR R3.
1383 002472 000000      .WORD 0        ;WORD 4, STORAGE FOR R4.
1384 002474 000000      .WORD 0        ;WORD 5, STORAGE FOR R5.
1385 ;*****
1386 ;*      TRANSMISSION AND RECEPTION VARIABLES, POINTERS, AND FLAGS.
1387 ;*****
1388 002476 000000      CHRTOT:: .WORD 0      ;TOTAL RECEIVED CHARACTER COUNTER.
1389 002500 000000      ERSMRF:: .WORD 0      ;"PRINT ERROR SUMMARY" FLAGS.
1390 002502 000000      TXDNF::  .WORD 0      ;TRANSMISSION DONE FLAGS.
1391 002504 000000      RXDNF::  .WORD 0      ;RECEPTION DONE FLAGS.
1392 002506 000000      TXDBLF:: .WORD 0      ;"TX HAS BEEN DISABLED" FLAG.
1393 ;*****
1394 ;      STORAGE AREA FOR THE BMP CODE QUEUE.
1395 ;*****
1396 002510 000000      BMPCQP::  .WORD 0      ;POINTER USED TO ACCESS THE NEXT CELL IN QUE.
1397 002512      BMPCQB::  .BLKW 64.     ;STORAGE FOR 32 CELLS, TEST# PLUS BMP CODE.
1398 002712      BMPCQE::      ;LAST ADDRESS PLUS 2 OF THE BMP CODE QUEUE.
1399 ;*****
1400 ;*      RECEIVE BUFFER AND ASSOCIATED VARIABLES.
1401 ;*****
1402 002712 000000      RXBQPT:: .WORD 0      ;RX BUFFER OUTPUT POINTER.

```

```

1403 002714 000000 RXBIPT:: .WORD 0 ;RX BUFFER INPUT POINTER.
1404 002716 000000 RXBCNT:: .WORD 0 ;COUNT OF NUMBER OF CHARS IN RX BUFFER.
1405 002720 RXBSTA:: ;LABEL AT BEGINNING OF THE RX BUFFER.
1406 002720 .BLKW RXBFUL ;LEAVE ENOUGH ROOM FOR A FULL BUFFER.
1407 003120 000000 RXBEND:: .WORD 0 ;LABEL AFTER END OF RX BUFFER.
1408 ;*****
1409 ;* TX/RX CONTROL BLOCK.
1410 ;*****
1411 003122 CBB:: ;BASE OF TX/RX CONTROL BLOCK.
1412 003122 000000 CBLPRA:: .WORD 0 ;LINE PARAMETER REGISTER CONTENTS.
1413 003124 000000 CBLNCA:: .WORD 0 ;LINE CONTROL REGISTER CONTENTS.
1414 003126 000000 CBDPAA:: .WORD 0 ;START ADDRESS OF DATA PATTERN.
1415 003130 000000 CBDPLA:: .WORD 0 ;LENGTH OF DATA PATTERN.
1416 003132 000000 CBDPNA:: .WORD 0 ;NUMBER OF REPEAT TRANSMISSIONS OF THE DATA PATTERN.
1417 003134 000000 CBMAPA:: .WORD 0 ;BIT MAP OF LINES TO INITIALISE.
1418 003136 000000 CBLPBA:: .WORD 0 ;LOOPBACK MODE (AS IN LOPBCK).
1419 003140 000000 CBOFSA:: .WORD 0 ;AMOUNT OF OFFSET BETWEEN EACH TX START.
1420 ;*****
1421 ;* TRANSMISSION AND RECEPTION TABLES OF POINTERS AND COUNTERS.
1422 ;*****
1423 003142 DPENDB:: .BLKW 16. ;TABLE OF END ADDRESSES OF DATA PATTERNS.
1424 003202 DPLENB:: .BLKW 16. ;TABLE OF LENGTH OF DATA PATTERNS FOR LINES.
1425 003242 EXCNTB:: .BLKW 16. ;EXTRA RECEIVED CHARACTER COUNTERS TABLE.
1426 003302 ERCNTB:: .BLKW 16. ;CHARACTER RECEIVE ERROR COUNTERS TABLE.
1427 003342 TXPTRB:: .BLKW 16. ;TRANSMISSION DATA POINTERS TABLE.
1428 003402 RXPTRB:: .BLKW 16. ;RECEPTION DATA POINTERS TABLE.
1429 003442 CHCNTB:: .BLKW 16. ;NUMBER OF CHARACTERS TO BE TXED AND RXED.
1430 003502 TXCNTB:: .BLKW 15. ;TRANSMISSION CHARACTER COUNTERS TABLE.
1431 003542 RXCNTB:: .BLKW 16. ;RECEPTION CHARACTER COUNTERS TABLE.
1432 ;*****
1433 ;* GENERAL TABLE AND BUFFER AREA--513 WORDS.
1434 ;*****
1435 003602 BUFBAS:: ;BASE OF MEMORY BUFFER.
1436 003602 ERLTBL:: .BLKW 128. ;FIRST HALF OF GENERAL TABLE OR BUFFER.
1437 004202 BUFMID:: .BLKW 64. ;SECOND HALF OF GENERAL TABLE OR BUFFER.
1438 004402 BUF3QT:: .BLKW 64. ;LAST QUARTER OF THE BUFFER AREA.
1439 004602 BUFBAS:: .BLKW 64. ;BASE OF MEMORY BUFFER.
1440 004602 BUFBAS:: .BLKW 64. ;BASE OF MEMORY BUFFER.
1441 ENDETB:: .BLKW 16. ;BUFFER OVERFLOW SPACE.
1442 ;*****
1443 ;* TABLE OF DATA PATTERN RESYNC QUEUES.
1444 ;*****
1444 004642 DPRSQB:: ;DATA PATTERN RESYNC QUEUES TABLE BASE.
1445 004642 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 0.
1446 004652 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 1.
1447 004662 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 2.
1448 004672 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 3.
1449 004702 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 4.
1450 004712 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 5.
1451 004722 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 6.
1452 004732 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 7.
1453 004742 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 8.
1454 004752 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 9.
1455 004762 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 10.
1456 004772 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 11.
1457 005002 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 12.
1458 005012 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 13.
1459 005022 .BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 14.

```

1460 005032
1461 005042
1462
1463
1464
1465 005042
1466 005042 000000
1467 005044 000010
1468 005046 000020
1469 005050 000030
1470 005052
1471 005052
1472 005052 000000
1473 005054 073400
1474 005056 177400
1475 005060
1476 005060
1477 005060 000000
1478 005062 000200
1479 005064
1480 005064
1481 005064 000000
1482 005066 000040
1483 005070 000140
1484 005072
1485
1486
1487
1488
1489
1490 005072
1491 005072 156470
1492 005074 167070
1493 005076 177470
1494 005100
1495
1496
1497
1498 005100
1499 005100 170070
1500 005102 007470
1501 005104 000001
1502 005106 000120
1503 005110 070470
1504 005112 013470
1505 005114 000001
1506 005116 000016
1507 005120 115070
1508 005122 124470
1509 005124 000001
1510 005126 000002
1511 005130
1512
1513
1514
1515 005130 000
1516 005131 001

```
.BLKW 4 ;DATA PATTERN RESYNC QUEUE FOR LINE 15.
DPRSQE:: ;END OF DATA PATTERN RESYNC QUEUES TABLE.
;*****
; SINGLE CHARACTER MODE LPR FIELD TABLES.
;*****
SCBCTB:: ;BASE OF NUMBER OF BITS PER CHAR FIELDS TABLE.
        .WORD 0 ;5 BITS/CHAR LPR FIELD.
        .WORD 10 ;6 BITS/CHAR LPR FIELD.
        .WORD 20 ;7 BITS/CHAR LPR FIELD.
        .WORD 30 ;8 BITS/CHAR LPR FIELD.
SCBCTE:: ;END OF NUMBER OF BITS/CHAR FIELDS TABLE.
SCBRTB:: ;BASE OF BAUDRATE FIELDS TABLE.
        .WORD 0 ;50 BAUD LPR FIELDS.
        .WORD 73400 ;1.2K BAUD LPR FIELDS.
        .WORD 177400 ;38.4K BAUD LPR FIELDS.
SCBRTE:: ;END OF BAUDRATE FIELDS TABLE.
SCNSTB:: ;BASE OF NUMBER OF STOP BITS FIELDS TABLE.
        .WORD 0 ;1 STOP BIT LPR FIELD.
        .WORD 200 ;2 STOP BITS LPR FIELD.
SCNSTE:: ;END OF BAUDRATE FIELDS TABLE.
SCTPTB:: ;BASE OF TYPE OF PARITY FIELDS TABLE.
        .WORD 0 ;NO PARITY LPR FIELD.
        .WORD 40 ;ODD PARITY LPR FIELD.
        .WORD 140 ;EVEN PARITY LPR FIELD.
SCTPTE:: ;END OF TYPE OF PARITY FIELDS TABLE.
;*****
; DMA MODE LPR FIELD TABLES.
; SET UP WITH SPECIFIED BAUDRATES, 1 STOP BIT, ODD PARITY, 8 BITS/CHAR.
;*****
DLPRTB:: ;BASE OF DMA TEST LPR FIELDS TABLE.
        .WORD 156470 ;9.6K BAUD.
        .WORD 167070 ;19.2K BAUD.
        .WORD 177470 ;38.4K BAUD.
DLPRTF:: ;END OF DMA TEST LPP FIELDS TABLE.
;*****
; SPLIT SPEED LPR PARAMETER TABLE.
;*****
SPLPRB:: ;BASE OF SPLIT SPEED LPR TABLE.
        .WORD 170070 ;TX: 38.4K, RX: 50 BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 7470 ;TX: 50, RX: 38.4K BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 1 ;NUMBER OF REPEAT TRANSMISSIONS AT 50 BAUD.
        .WORD 80 ;NUMBER OF REPEAT TRANSMISSIONS AT 38.4K BAUD.
        .WORD 70470 ;TX: 1200, RX: 75 BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 13470 ;TX: 75, RX: 1200 BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 1 ;NUMBER OF REPEAT TRANSMISSIONS AT 75 BAUD.
        .WORD 16 ;NUMBER OF REPEAT TRANSMISSIONS AT 1200 BAUD.
        .WORD 115070 ;TX: 2000, RX:2400 BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 124470 ;TX: 2400, RX:2000 BAUD, 1 STOP ODD PAR 8 BITS.
        .WORD 1 ;NUMBER OF REPEAT TRANSMISSIONS AT 2400 BAUD.
        .WORD 2 ;NUMBER OF REPEAT TRANSMISSIONS AT 2000 BAUD.
SPLPRE:: ;END OF SPLIT SPEED LPR TABLE.
;*****
; SINGLE CHARACTER DATA PATTERN TABLE.
;*****
SOPBAS::.BYTE 0 ;START OF SINGLE CHARACTER DATA PATTERN TABLE.
        .BYTE 1
```

1517	005132	010	.BYTE	10
1518	005133	017	.BYTE	17
1519	005134	063	.BYTE	63
1520	005135	074	.BYTE	74
1521	005136	125	.BYTE	125
1522	005137	177	.BYTE	177
1523	005140	200	.BYTE	200
1524	005141	252	.BYTE	252
1525	005142	303	.BYTE	303
1526	005143	314	.BYTE	314
1527	005144	360	.BYTE	360
1528	005145	367	.BYTE	367
1529	005146	376	.BYTE	376
1530	005147	377	.BYTE	377

```
SDPEND::                                ;END OF SINGLE CHARACTER DATA PATTERN TABLE.
                                           ;START OF FIRST SHORT DATA PATTERN OVERFLOW AREA.
                                           .BYTE 0
                                           .BYTE 1
                                           .BYTE 10
                                           .BYTE 17
```

1536
1537
1538
1539

```
*****
; SINGLE CHARACTER DATA PATTERN TABLE NUMBER TWO.
*****
SDP2B:: .BYTE 125 ;START OF SECOND SHORT DATA PATTERN.
```

1540	005154	125	.BYTE	125
1541	005155	252	.BYTE	252
1542	005156	124	.BYTE	124
1543	005157	253	.BYTE	253
1544	005160	122	.BYTE	122
1545	005161	255	.BYTE	255
1546	005162	112	.BYTE	112
1547	005163	265	.BYTE	265
1548	005164	052	.BYTE	52
1549	005165	325	.BYTE	325
1550	005166	152	.BYTE	152
1551	005167	225	.BYTE	225
1552	005170	132	.BYTE	132
1553	005171	245	.BYTE	245
1554	005172	126	.BYTE	126
1555	005173	251	.BYTE	251

```
SDP2E::                                ;END OF SECOND SHORT DATA PATTERN.
                                           ;START OF SECOND SHORT DATA PATTERN OVERFLOW AREA.
                                           .BYTE 125
                                           .BYTE 252
                                           .BYTE 124
                                           .BYTE 253
                                           .BYTE 122
                                           .BYTE 255
                                           .BYTE 112
                                           .BYTE 265
                                           .BYTE 52
                                           .BYTE 325
                                           .BYTE 152
                                           .BYTE 225
                                           .BYTE 132
                                           .BYTE 245
                                           .BYTE 126
                                           .BYTE 251
```

1556	005174	125	.BYTE	125
1557	005174	125	.BYTE	125
1558	005175	252	.BYTE	252
1559	005176	124	.BYTE	124
1560	005177	253	.BYTE	253
1561	005200	122	.BYTE	122
1562	005201	255	.BYTE	255
1563	005202	112	.BYTE	112
1564	005203	265	.BYTE	265
1565	005204	052	.BYTE	52
1566	005205	325	.BYTE	325
1567	005206	152	.BYTE	152
1568	005207	225	.BYTE	225
1569	005210	132	.BYTE	132
1570	005211	245	.BYTE	245
1571	005212	126	.BYTE	126
1572	005213	251	.BYTE	251
1573				

```
*****
```

1574
1575
1576 005214 372
1577 005215 252
1578 005216 167
1579 005217 143
1580 005220 132
1581 005221 062
1582 005222 036
1583 005223 024
1584 005224 021
1585 005225 020
1586 005226 017
1587 005227 015
1588 005230 014
1589 005231 014
1590 005232 013
1591 005233 012
1592
1593
1594
1595
1596
1597
1598
1599 005234
1600 005234 000000
1601 005236 000002
1602 005240 000004
1603 005242 000006
1604 005244 000010
1605 005246 000012
1606 005250 000014
1607 005252 000016
1608 005254 000020
1609 005256 000022
1610 005260 000024
1611 005262 000026
1612 005264 000030
1613 005266 000032
1614 005270 000034
1615 005272 000036
1616 005274
1617
1618
1619
1620
1621
1622
1623
1624
1625 005274
1626 005274 004
1627 005275 006
1628 005276 000
1629 005277 002
1630 005300 014

```

; SINGLE CHARACTER SAFE PROPORTIONAL DELAY TABLE.
;*****
PROTBL:;.BYTE 250. ;DELAY IN MILLI SECONDS AT 50 BAUD
        .BYTE 170. ;DELAY IN MILLI SECONDS AT 75 BAUD
        .BYTE 119. ;DELAY IN MILLI SECONDS AT 110 BAUD
        .BYTE 99.  ;DELAY IN MILLI SECONDS AT 134.5 BAUD
        .BYTE 90.  ;DELAY IN MILLI SECONDS AT 150 BAUD
        .BYTE 50.  ;DELAY IN MILLI SECONDS AT 300 BAUD
        .BYTE 30.  ;DELAY IN MILLI SECONDS AT 600 BAUD
        .BYTE 20.  ;DELAY IN MILLI SECONDS AT 1200 BAUD
        .BYTE 17.  ;DELAY IN MILLI SECONDS AT 1800 BAUD
        .BYTE 16.  ;DELAY IN MILLI SECONDS AT 2000 BAUD
        .BYTE 15.  ;DELAY IN MILLI SECONDS AT 2400 BAUD
        .BYTE 13.  ;DELAY IN MILLI SECONDS AT 4800 BAUD
        .BYTE 12.  ;DELAY IN MILLI SECONDS AT 7200 BAUD
        .BYTE 12.  ;DELAY IN MILLI SECONDS AT 9600 BAUD
        .BYTE 11.  ;DELAY IN MILLI SECONDS AT 19200 BAUD
        .BYTE 10.  ;DELAY IN MILLI SECONDS AT 38400 BAUD
        .EVEN
;*****
;* TABLE FOR STORAGE OF RX/TX LINE NUMBER 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:;.WORD 0 ;BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
        .WORD 2. ;TX/RX LINE OFFSET FOR RX/TX LINE 0.
        .WORD 4. ;TX/RX LINE OFFSET FOR RX/TX LINE 1.
        .WORD 6. ;TX/RX LINE OFFSET FOR RX/TX LINE 2.
        .WORD 8. ;TX/RX LINE OFFSET FOR RX/TX LINE 3.
        .WORD 10. ;TX/RX LINE OFFSET FOR RX/TX LINE 4.
        .WORD 12. ;TX/RX LINE OFFSET FOR RX/TX LINE 5.
        .WORD 14. ;TX/RX LINE OFFSET FOR RX/TX LINE 6.
        .WORD 16. ;TX/RX LINE OFFSET FOR RX/TX LINE 7.
        .WORD 18. ;TX/RX LINE OFFSET FOR RX/TX LINE 8.
        .WORD 20. ;TX/RX LINE OFFSET FOR RX/TX LINE 9.
        .WORD 22. ;TX/RX LINE OFFSET FOR RX/TX LINE 10.
        .WORD 24. ;TX/RX LINE OFFSET FOR RX/TX LINE 11.
        .WORD 26. ;TX/RX LINE OFFSET FOR RX/TX LINE 12.
        .WORD 28. ;TX/RX LINE OFFSET FOR RX/TX LINE 13.
        .WORD 30. ;TX/RX LINE OFFSET FOR RX/TX LINE 14.
        .WORD 30. ;TX/RX LINE OFFSET FOR RX/TX LINE 15.
TXRXLE:;.EVEN ;END OF TX/RX LINE NUMBER ASSOCIATION TABLE.
        .EVEN ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
;*****
;* TABLE OF TX/RX LINE NUMBER ASSOCIATIONS IN STAGGERED LOOPBACK.
;* THE ASSOCIATIONS ARE STORED AS LINE NUMBER TIMES 2 FOR USE AS OFFSETS
;* WHEN ACCESSING A TABLE OF WORDS.
;* THIS IS A TABLE OF DATA FOR READING ONLY. USE TO LOAD THE ABOVE TABLE.
;* NOTE: MUST CONVERT FROM BYTES TO WORDS WHEN LOADING ABOVE TABLE.
;*****
STGRB:;.BYTE 4. ;BASE OF STAGGERED TX/RX LINE NUMBER TABLE.
        .BYTE 6. ;TX/RX LINE OFFSET FOR RX/TX LINE 0.
        .BYTE 0. ;TX/RX LINE OFFSET FOR RX/TX LINE 1.
        .BYTE 2. ;TX/RX LINE OFFSET FOR RX/TX LINE 2.
        .BYTE 12. ;TX/RX LINE OFFSET FOR RX/TX LINE 3.
        .BYTE 12. ;TX/RX LINE OFFSET FOR RX/TX LINE 4.

```

```

1631 005301      016      .BYTE      14.      ;TX/RX LINE OFFSET FOR RX/TX LINE 5.
1632 005302      010      .BYTE      8.       ;TX/RX LINE OFFSET FOR RX/TX LINE 6.
1633 005303      012      .BYTE     10.       ;TX/RX LINE OFFSET FOR RX/TX LINE 7.
1634 005304      024      .BYTE     20.       ;TX/RX LINE OFFSET FOR RX/TX LINE 8.
1635 005305      026      .BYTE     22.       ;TX/RX LINE OFFSET FOR RX/TX LINE 9.
1636 005306      020      .BYTE     16.       ;TX/RX LINE OFFSET FOR RX/TX LINE 10.
1637 005307      022      .BYTE     18.       ;TX/RX LINE OFFSET FOR RX/TX LINE 11.
1638 005310      034      .BYTE     28.       ;TX/RX LINE OFFSET FOR RX/TX LINE 12.
1639 005311      036      .BYTE     30.       ;TX/RX LINE OFFSET FOR RX/TX LINE 13.
1640 005312      030      .BYTE     24.       ;TX/RX LINE OFFSET FOR RX/TX LINE 14.
1641 005313      032      .BYTE     26.       ;TX/RX LINE OFFSET FOR RX/TX LINE 15.
1642              .EVEN
1655 005314      ERRTBL  ;GUARANTEE THAT NEXT TABLE IS ON WORD BOUNDARY.
      005314
      005314 000000      ERRTP::      .WORD      0
      005316 000000      ERRNBR::     .WORD      0
      005320 000000      ERRMSG::     .WORD      0
      005322 000000      ERRBLK::    .WORD      0
1656
1657              .EVEN

```

1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695

```

.SBTTL GPR HANDLING ROUTINES FOR SUBROUTINE CALLS.
*****
;*
;* THERE ARE 4 ROUTINES AND MACRO DEFINITIONS USED FOR THE HANDLING OF
;* GPR VALUES DURING SUBROUTINE CALLS WITHIN THIS PROGRAM. THE FOUR
;* ROUTINES/MACRO CALLS HAVE THE FOLLOWING NAMES:
;*
;* SAVE - MACRO DEFINITION USED AT THE BEGINNING OF A SUBROUTINE TO
;* SAVE THE GPR CONTENTS FOR LATER RESTORATION.
;* PASS - MACRO DEFINITION USED AT THE END OF A SUBROUTINE TO RESTORE
;* THE PREVIOUSLY SAVED GPR CONTENTS AND TO LEAVE THE CONTENTS
;* OF THE SPECIFIED GPR(S) INTACT (NOT RESTORED).
;* PREGOS - SUBROUTINE WHICH IS CALLED FROM THE SAVE AND PASS MACRO
;* EXPANSIONS WHICH ACTUALLY PERFORMS THE ACTIONS ON THE GPRS.
;*
;* DURING A SUBROUTINE WHICH USES THESE GPR SAVE ROUTINES THE VALUES
;* OF THE GPRS ARE STORED ON THE STACK IN THE FOLLOWING STACK FRAME:
;*
;*          SP    -> RET PC INTO PREGOS ROUTINE.
;*          SP+2  -> GPR R0 CONTENTS.
;*          SP+4  -> GPR R1 CONTENTS.
;*          SP+6  -> GPR R2 CONTENTS.
;*          SP+8  -> GPR R3 CONTENTS.
;*          SP+10 -> GPR R4 CONTENTS.
;*          SP+12 -> GPR R5 CONTENTS.
;*          SP+14 -> RET PC INTO CALLER OF SUB' TNE WHICH CALLED PREGOS.
;*
;* EACH LEVEL OF SUB' TNE CALLING USES 8 WORDS OF STACK OVERHEAD.
;* THE SAVE AND PASS MACROS CAN ALSO BE USED IN "STRAIGHT LINE CODE"
;* TO SAVE AND RESTORE THE GPR VALUES. IN ANY CASE, AFTER THE
;* ISSUING OF A PASS CALL THE GPRS WILL BE RESTORED TO THE VALUES
;* THEY HAD PRIOR TO THE LAST SAVE CALL (EXCEPT FOR THE EXCEPTED,
;* OR PASSED INTACT, GPRS SPECIFIED AS PARAMETERS TO THE PASS CALL)
;* AND THE SP WILL ALSO BE RESTORED TO ITS CONDITION BEFORE THE LAST
;* SAVE CALL. THE PROGRAMMER MUST BE SURE THAT THE SP HAS THE SAME
;* VALUE WHEN THE PASS MACRO IS CALLED AS IT HAD IMMEDIATELY AFTER
;* THE SAVE MACRO WAS CALLED.
*****

```

1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711

.SBTTL GPR FRAME ACCESS EQUATES

!***
!EQUATES THAT ALLOW ACCESS TO THE STACK FRAME. THESE ARE THE
!OFFSETS INTO THE STACK FOR REGISTERS SAVED DURING THE PREG05
!ROUTINE.
!---

000036	LPCSLT**	36	!OFFSET FOR LAST RETURN PC.
000016	PCSLT**	16	!OFFSET FOR RETURN PC.
000014	R5SLOT**	14	!OFFSET FOR R5.
000012	R4SLOT**	12	!OFFSET FOR R4.
000010	R3SLOT**	10	!OFFSET FOR R3.
000006	R2SLOT**	6	!OFFSET FOR R2.
000004	R1SLOT**	4	!OFFSET FOR R1.
000002	R0SLOT**	2	!OFFSET FOR R0.

1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736

```

.SBTTL GLOBAL MACRO DEFINITION          - SAVE
*****
; * THIS MACRO IS USED AT THE BEGINNING OF A SUBROUTINE TO SAVE THE
; * CONTENTS OF THE GPRS R0 THRU R5.
; *
; * INPUTS:      SP - UNCHANGED SINCE SUBROUTINE WAS ENTERED
; *              R5SLOT - OFFSET TO STACK SLOT FOR R5 (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.
*****
      .MACRO SAVE
      .LIST
              JSR      R5,PREG05          ;CALL REGISTER SAVE SUBRT.
      .NLIST
      .ENDM SAVE

```

1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785

```

.SBTTL 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 PREG05 TO RESTORE THE GPRS TO THEIR SAVED VALUES.
; *
; * INPUTS:      ONLY ALLOWED ARGUMENTS ARE "R0" THRU "R5".
; *              ROSLOT THRU R5SLOT 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  R0,R1,...
; *
; * COMMENTS:    ANY COMBINATION OF GPR ARGUMENTS MAY BE LISTED IN ANY ORDER.
; *              FOR EXAMPLE, THE FOLLOWING ARE LEGAL:
; *              PASS  R1
; *              PASS  R4,R0,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  R0,R3
; * EXPANDS INTO THE FOLLOWING ASSEMBLY CODE:
; *              MOV   R0,ROSLOT(SP)          ;PUT R0 IN STACK SLOT.
; *              MOV   R3,R3SLOT(SP)        ;PUT R3 IN STACK SLOT.
; *              JSR   PC,@(SP)+            ;RETURN TO PREG05 SUBRT.
; * IN THIS EXAMPLE GPRS R1, R2, R4, AND R5 WILL BE RESTORED TO
; * THEIR VALUES CONTAINED IN THE STACK FRAME AND R0 AND R3
; * WILL BE LEFT AT THEIR VALUES PRIOR TO THIS PASS CALL.
; *
; * SUBORDINATE ROUTINES CALLED: (PREGRT - LABEL WITHIN PREG05. VALUE ON STACK.)
*****
; *
; * .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 PREG05 SUBRT.
; * .NLIST
; * .ENDM PASS

```

1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839

005324
005324 010446
005326 C10346
005330 010246
005332 010146
005334 010046
005336 010546
005340 016605 000014
005344 004736
012605
012600
012601
012602
012603
012604
005362 000205

```

.SBTTL GLOBAL SUBROUTINE - PREG05 -
;*****
; PRESERVE REGISTERS R0 THROUGH R5 FOR SUBROUTINE CALLS.
;
; INPUTS: THE RETURN ADDRESS BACK INTO THE CALLING ROUTINE MUST BE IN
; GPR R5. (I.E.- MACROS USE "JSR R5,PREG05".)
;
; OUTPUTS: REGISTERS R0 THROUGH R5 ARE SAVED ON THE STACK.
;
; CALLING SEQUENCE: SAVE [SUBROUTINE CODE]... ;MACRO EXPANSION CALLS PREG05.
; PASS ;MACRO EXPANSION RECALLS PREG05.
;
; COMMENTS: THIS ROUTINE IS RE-ENTRANT.
;
; PARAMETERS MAY BE PASSED OUT OF A SUBROUTINE BY MODIFYING THE
; REGISTER SAVE AREA ON THE STACK. USE THE PASS GPRN MACRO
; TO RETURN GPR VALUES INTACT.
; USE THE RNSLOT OFFSETS FROM THE SP TO PASS OTHER PARAMETERS.
; [EXAMPLE: MOV VALUE,R0SLOT(SP) ]
; MAKE SURE THE SP IS AT ITS ORIGINAL VALUE WHEN YOU DO THIS.
;
; SUBORDINATE ROUTINES CALLED: NONE.
;*****
PREG05: ;R5 HAS BEEN LOADED ON THE STACK BY THE SUBROUTINE CALL
MOV R4,-(SP) ;SAVE R4
MOV R3,-(SP) ;SAVE R3
MOV R2,-(SP) ;SAVE R2
MOV R1,-(SP) ;SAVE R1
MOV R0,-(SP) ;SAVE R0
MOV R5,-(SP) ;PUSH RETURN PC ON TOP OF STACK
MOV R5SLOT(SP),R5 ;RESTORE R5 TO VALUE IT HAD BEFORE CALLS
JSR PC,@(SP)+ ;CALL THE SUBROUTINE AT THE RETURN ADDRESS
;FROM THE PREG05 CALL, PUTTING THE PRESENT
;PC ON THE STACK AS A RETURN ADDRESS INTO
;THIS (PREG05) ROUTINE.
;
; THE FOLLOWING CODE IS EXECUTED WHEN THE CALLING ROUTINE DOES A
;"RETURN" (JSR PC,@(SP)+) USING THE PC DEPOSITED ON THE STACK ABOVE.
;
PREGRT:; MOV (SP)+,R5 ;PUT RETURN PC IN R5.
MOV (SP)+,R0 ;RESTORE R0.
MOV (SP)+,R1 ;RESTORE R1.
MOV (SP)+,R2 ;RESTORE R2.
MOV (SP)+,R3 ;RESTORE R3.
MOV (SP)+,R4 ;RESTORE R4.
RTS R5 ;RETURN TO THE SUBROUTINE WHICH CALLED PREG05.
;RESTORING R5 IN THE PROCESS.

```

```

1841          .SBTTL GLOBAL TEXT SECTION
1849
1850
1851          ;**
1852          ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
1853          ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
1854          ; MORE THAN ONE TEST.
1855          ;**
1856
1857          ;
1858          ; NAMES OF DEVICES SUPPORTED BY PROGRAM
1859          ;
1860          DEVTYP <DHU-11>
1861
1862          L$DVTYP::
1863          .ASCIZ /DHU-11/
1864
1865          .EVEN
1866
1867          ; TEST DESCRIPTION
1868          ;
1869          DESCRIPT <DHU-11 FUNC TST PART4>
1870
1871          L$DESC::
1872          .ASCIZ /DHU-11 FUNC TST PAR
1873
1874          .EVEN
1875
1876          .EVEN
1877
1878
1879

```

005364	104	110	125
005364	051	061	061
005367	000		
005372			
005374	104	110	125
005374			
005374			
005377	055	061	061
005402	040	106	125
005405	116	103	040
005410	124	123	124
005413	040	120	101
005416	122	124	064
005421	000		

```

1888
1889      .NLIST BIN
1890
1891
1892      ; ***** FORMAT STATEMENTS USED IN PRINT CALLS *****
1893
1894
1895 005422 EDPFMT: .ASCII /#AMODEM LOOPBACK TEST STATUS REPORT: /
1896 005470      .ASCIZ /PATTERN #D5#A (D) COMPLETED.#N/
1897 005530 EF0503: .ASCIZ /#T#N/
1898 005535 EF1601: .ASCIZ /#A #T#A, TEST ABORTED #N/
1899 005567 EF1602: .ASCIZ /#A ACTUAL DATA: #06#A (O).#N/
1900 005631 EF4401: .ASCII /#N#A DMA ADDRESS TEST SUCCESSFUL, BITS 0 TO #D2#A (D) TESTED/
1901 005725      .ASCIZ / (#D2#A BITS).#N/
1902 005746 EF6201: .ASCIZ \#A FRAMING/PARITY ERROR DETECTION AND REPORTING BAD ON LINES:#D2#A : #D2#N\
1903 006061 EF6202: .ASCIZ /#A CHAR RECEIVED WITH FRAMING ERROR BIT #T#A, SHOULD BE #T#N/
1904 006157 EF6203: .ASCIZ /#A CHAR RECEIVED WITH PARITY ERROR BIT #T#A, SHOULD BE #T#N/
1905 006254 EF9001: .ASCIZ /#A UNEXPECTED #T#A FOUND IN RECEIVE CHAR FIFO:#N/
1906 006336 EF9002: .ASCIZ /#A CODE IS ASSOCIATED WITH LINE: #D2#N/
1907 006410 EF9003: .ASCIZ /#A CODE IS: #03#N/
1908 00 37 EF9004: .ASCIZ /#A #T#A VALUE: #03#N/
1909 006467 EF9005: .ASCIZ /#A #T#A VALUE: NONF#N/
1910 006520 EF9006: .ASCIZ /#A #T#A #D2#A(D)#N/
1911 006544 EF9007: .ASCIZ /#A CHARACTER RECEIVED WITH ERROR FLAG(S) SET ON LINE #D2#N/
1912 006640 EF9008: .ASCIZ /#A CHARACTER READ AS: #03#N/
1913 006677 EF9009: .ASCIZ /#A #T#A ERROR FLAG SET.#N/
1914 006736 EF9010: .ASCIZ /#A NUMBER OF ERRORS DETECTED ON LINE #D2#A IS #D5#N/
1915 007025 EF9011: .ASCII /#A LINE#D2#A ONLY #T#D5#A BYTES OF#D5#A BYTE/
1916 007101      .ASCIZ / DATA PAT'N IX'D FROM LINE#D2#N/
1917 007141 EF9013: .ASCIZ /#A DATA PATTERN NOT COMPLETELY #T#N/
1918 007206 EF9019: .ASCIZ /#A #T#A #06#N/
1919 007225 EF9020: .ASCIZ /#A TOO FEW TX.ACTIONS GENERATED ON LINE #D2#N/
1920 007306 EF9101: .ASCIZ /#N/
1921 007311 EF9103: .ASCIZ /#A ERROR CONDITION ON LINE #D2#N/
1922 007357 EF9301: .ASCIZ /#A #T#D2#A(D), BMP CODE REPORTED :#03#A(O)#N/
1923 007435 EF9302: .ASCIZ /#A OVERFLOW OCCURRED (MORE THAN 31 BMP CODES FOUND IN QUEUE)#N/
1924 007535 MFUNIT: .ASCIZ /#N#A TESTING UNIT :#D4#N/
1925 007566 MSFMT1: .ASCIZ /#AMODEM STATUS SIGNAL REPORT:#N/
1926 007626 MSFMT2: .ASCIZ /#A LINE #D2#A: DSR=#B1#A, RI=#B1#A, DCD=#B1#A, CTS=#B1#N/
1927 007722 UBRFMT: .ASCIZ /#D5#A IS NOT A SUPPORTED BAUDRATE, ENTER ANOTHER OR CTRL C.#N/
1928      .EVEN
1929      .LIST BIN

```

```

1938
1939      .NLIST BIN
1940
1941
1942      ;***** GLOBAL ERROR MESSAGES *****
1943
1944 010020 BDRMSG: .ASCIZ /MODEM BAUDRATE IN BPS:/
1945 010047 EM0105: .ASCIZ /DEVICE REGISTER ACCESS ERRORS/
1946 010105 EM0509: .ASCIZ /SET/
1947 010111 EM1601: .ASCIZ /TIMEOUT OCCURRED WAITING FOR MASTER RESET TO CLEAR/
1948 010174 EM4401: .ASCIZ /DMA ADDRESS TEST FAILED/
1949 010224 EM4402: .ASCIZ /NO SUITABLE ADDR FOUND,TEST ABANDONED/
1950 010272 EM4403: .ASCIZ /**HOST FAILURE**,WRITE FAILED TO AN ADDR WHICH HAD BEEN READ/
1951 010367 EM1404: .ASCIZ /NO ACTIVE LINES,TEST ABANDONED/
1952 010426 EM4405: .ASCIZ /DMA_START BIT FOUND SET BEFORE DMA INITIATED,TEST ABANDONED/
1953 010522 EM4406: .ASCIZ /TIME-OUT OCCURED WAITING FOR DMA TO FINISH/
1954 010576 EM4407: .ASCIZ /TOO FEW CHARACTERS FOUND IN THE RXFIFO,DMA FAILED/
1955 010660 EM4408: .ASCIZ /TOO MANY BMP CODES FOUND IN RXFIFO/
1956 010723 EM4409: .ASCIZ /BAD BITS BETWEEN BITS 0 AND /
1957 010760 EM4410: .ASCIZ /RXFIFO FAILED TO PURGE/
1958 011007 EM4411: .ASCIZ /**HOST FAILURE**WRITE ATTEMPT FAILED/
1959 011054 EM6201: .ASCIZ /FRAMING ERROR TEST FAILED/
1960 011106 EM6202: .ASCIZ /CLEAR /
1961 011115 EM6301: .ASCIZ /PARITY ERROR TEST FAILED/
1962 011146 EM8901: .ASCIZ /MODEM LOOPBACK TEST /
1963 011173 EM9003: .ASCIZ /MODEM STATUS CODE/
1964 011215 EM9004: .ASCIZ /SELFTEST CODE/
1965 011233 EM9006: .ASCIZ /CHARACTER RECEIVED ON INACTIVE LINE, LINE:/
1966 011306 EM9007: .ASCIZ /UNEXPECTED CHAR RECEIVED AFTER RX COMPLETE ON LINE/
1967 011371 EM9008: .ASCIZ /RECEIVED CHAR MISCOMPARE AGAINST TX DATA ON LINE/
1968 011452 EM9009: .ASCIZ /EXPECTED OR CORRECT/
1969 011476 EM9010: .ASCIZ /ACTUAL OR MEASURED /
1970 011522 EM9011: .ASCIZ /OVERRUN/
1971 011532 EM9012: .ASCIZ /FRAMING/
1972 011542 EM9013: .ASCIZ /PARITY/
1973 011551 EM9014: .ASCIZ /SUMMARY REPORTS FOR LINES WITH EXCESSIVE NUMBERS OF ERRORS:/
1974 011645 EM9015: .ASCIZ /TRANSMITTED/
1975 011661 EM9016: .ASCIZ /RECV'D/
1976 011670 EM9017: .ASCII / FIFO WILL NOT PURGE (DATA,VALID STUCK SET),/
1977 011745      .ASCIZ / REMAINDER OF TEST SKIPPED./
1978 012001 EM9025: .ASCIZ /MORE THAN TWICE THE EXPECTED NUMBER OF CHARACTERS RECEIVED./
1979 012075 EM9026: .ASCIZ / LPR CONTENTS: /
1980 012121 EM9027: .ASCIZ /EXTRA CHAR RECEIVED WITHIN DATA PATTERN ON LINE/
1981 012201 EM9028: .ASCIZ /SINGLE CHAR MISSING FROM RECEIVED DATA ON LINE/
1982 012260 EM9030: .ASCIZ /#A (NO TX COMPLETION INTERRUPTS RECEIVED)#N/
1983 012335 EM9101: .ASCIZ /DMA TRANSMISSION MODE TEST FAILED/
1984 012377 EM9102: .ASCIZ /DMA_START BIT SET AFTER RESET OR TX ACTION ON LINE(S):/
1985 012466 EM9104: .ASCIZ / UNEXPECTED DATA FOUND IN FIFO FROM LINE: /
1986 012542 EM9201: .ASCIZ /SPLIT SPEED TEST FAILED/
1987 012572 EM9301: .ASCIZ /BMP CODES WERE REPORTED DURING THIS DIAGNOSTIC/
1988 012651 EM9302: .ASCIZ /BMP CODE FOUND IN TEST /
1989 012701 EM9303: .ASCIZ /THE LAST BMP CODE WAS FOUND IN TEST /
1990 012746 EM9304: .ASCIZ /UNEXPECTED BMP CODES FOUND DURING THIS PASS/
1991 013022 EM9401: .ASCIZ /KEYBOARD ECHO (DHU REMOTE LOOPBACK) TEST /
1992 013074 EM1150: .ASCIZ /TYPE <CR> WHEN MODEM LINK ESTABLISHED:/
1993 013143 EXITMG: .ASCIZ /EXIT THE TEST (N = LOOP BACK TO SEND MORE DATA):/
1994 013224 NDRMSG: .ASCII /NUMBER OF 256 BYTE PATTERNS TO SEND ON EACH SELECTED LINE/

```

1995 013315 .ASCIZ <15><12>/ (1-255, 0=SEND UNTIL ↑C);/
1996 013352 PMSMSG:: .ASCIZ /PRINT MODEM STATUS SIGNAL REPORT AFTER EACH PATTERN;/
1997 013437 TERMSG:: .ASCIZ /TYPE <CR> TO TERMINATE THE TEST;/
1998 .EVEN
1999 .LIST BIN

2008
2009
2010
2011
2012
2013
2014
2015
2016

.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.
;--

2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057

013500
013500
013500 004567 171520
013504 012700 000100
013510 046700 166446
013514 001036
013516 032705 000001
013522 001410
013524 012746 013616
013530 012746 000001
013534 010600
013536 104414
013540 062706 000004
013544 032705 000002
013550 001410
013552 012746 013674
013556 012746 000001
013562 010600
013564 104414
013566 062706 000004
013572 012746 013753
013576 012746 000001
013602 010600
013604 104415
013606 062706 000004

```
.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER0101 -
*****
; 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 DHU MAY BE AT THE WRONG UNIBUS ADDRESS IS ALSO PRINTED.
;
; INPUTS: R5 - ERROR FLAG WORD,
; IF BIT 0 IS SET, A READ ERROR OCCURED
; IF BIT 1 IS SET, A WRITE ERROR OCCURED.
;
; OUTPUTS: MESSAGES ARE PRINTED AT THE OPERATOR CONSOLE.
;
; CALLING SEQUENCE: INCLUDE THE LABEL "ER0101" AS THE MESSAGE POINTER
; PARAMETER IN THE DRS ERROR REPORT MACRO CALL.
;
; COMMENTS:
;
; SUBORDINATE ROUTINES USED: NONE.
*****
BGNMSG ER0101
ER0101::
SAVE JSR ;SAVE THE GPR CONTENTS.
R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV #BIT06,R0 ;SET-UP THE BIT MAP FOR 'REPORT EXT'D ERROR INFO'
BIC OPTION,R0 ;TRY AND CLEAR THE FLAG.
RNE 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,R0
TRAP C$PNTB
ADD #4,SP
2$: 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,R0
TRAP C$PNTB
ADD #4,SP
4$: PRINTX #MSG3 ;SUGGEST THAT DHU MAY BE AT WRONG ADDRESS.
MOV #MSG3,-(SP)
MOV #1,(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
```

```

2058 013612          6$:  PASS          ;RESTORE THE GPR CONTENTS.
      013612 004736          JSR          PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2059 013614          ENDMSG
      013614          L100G2: TRAP      C$MSG
      013614 104423
2060
2061 013616          045      101      102 MSG1:: .ASCIZ /*ABUS TIME-OUT TRAP CAUSED BY READ ATTEMPT,*/N/
      013621          125      123      040
      013624          124      111      115
      013627          105      055      117
      013632          125      124      040
      013635          124      122      101
      013640          120      040      103
      013643          101      125      123
      013646          105      104      040
      013651          102      131      040
      013654          122      105      101
      013657          104      040      101
      013662          124      124      105
      013665          115      120      124
      013670          056      045      116
      013673          000
2062 013674          045      101      102 MSG2:: .ASCIZ /*ABUS TIME-OUT TRAP CAUSED BY WRITE ATTEMPT,*/N/
      013677          125      123      040
      013702          124      111      115
      013705          105      055      117
      013710          125      124      040
      013713          124      122      101
      013716          120      040      103
      013721          101      125      123
      013724          105      104      040
      013727          102      131      040
      013732          127      122      111
      013735          124      105      040
      013740          101      124      124
      013743          105      115      120
      013746          124      056      045
      013751          116      000
2063 013753          045      101      104 MSG3:: .ASCIZ /*ADHU MAY BE AT THE WRONG UNIBUS ADDRESS,*/N*/N/
      013756          110      125      040
      013761          115      101      131
      013764          040      102      105
      013767          040      101      124
      013772          040      124      110
      013775          105      040      127
      014000          122      117      116
      014003          107      040      125
      014006          116      111      102
      014011          125      123      040
      014014          101      104      104
      014017          122      105      123
      014022          123      056      045
      014025          116      045      116
      014030          000
2064
2065          .EVEN

```

```

2067 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER0503 -
2068 ;*****
2069 ;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS AN ADDITIONAL ERROR
2070 ;* MESSAGE WHOSE ADDRESS IS PASSED AS AN INPUT PARAMETER, PROVIDED
2071 ;* EXTENDED ERROR REPORTING HAS BEEN REQUESTED.
2072 ;*
2073 ;* INPUTS: R1 - ADDRESS OF THE MESSAGE TO PRINT.
2074 ;*
2075 ;* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
2076 ;*
2077 ;* CALLING SEQUENCE: LOAD THE ADDRESS OF THE MESSAGE IN R1.
2078 ;* INCLUDE THE LABEL "ER0503" AS THE MESSAGE POINTER
2079 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2080 ;*
2081 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION.
2082 ;*
2083 ;* SUBORDINATE ROUTINES USED: NONE.
2084 ;*****
2085
2086 014032 BGNMSG ER0503
2087 014032 ER0503:;
2088 014032 012700 000100 MOV #BIT06,R0 ;TRY TO CLEAR THE
2089 014036 046700 166120 BIC OPTION,R0 ;EXT'D ERROR REPORTING FLAG
2090 014042 001011 BNE 2$ ;EXIT IF FLAG NOT SET.
2091
2092
2093 014044 PRINTB #EF0503,R1 ;PRINT THE MESSAGE.
2094 014044 010146 MOV R1,-(SP)
2095 014046 012746 005530 MOV #EF0503,-(SP)
2096 014052 012746 000002 MOV #2,-(SP)
2097 014056 010600 MOV SP,R0
2098 014060 104414 TRAP C$PNTB
2099 014062 062706 000006 ADD #6,SP
2094
2095 014066 2$: ENDMSG
2096 014066
2097 014066 104423 L10003: TRAP C$MSG

```

```

2097 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER1603 -
2098 ;*****
2099 ;* THIS ERROR REPORTING ROUTINE IS USED TO PRINT OUT A BASIC ERROR
2100 ;* MESSAGE, ALONG WITH A MESSAGE INFORMING THE OPERATOR WHICH TEST IS
2101 ;* ABOUT TO BE ABORTED, PROVIDED EXTENDED ERROR INFORMATION HAS BEEN
2102 ;* REQUESTED, OTHERWISE ONLY A "TEST FAILURE" MESSAGE WILL BE PRINTED.
2103 ;*
2104 ;* INPUTS: R1 - CONTAINS THE ADDRESS OF THE MESSAGE TO BE PRINTED.
2105 ;* ERRMSG - CONTAINS THE ADDRESS OF THE MESSAGE THAT INDICATES
2106 ;* THE TEST THAT IS BEING PERFORMED, EG DMA, BREAK ETC.
2107 ;*
2108 ;* OUTPUTS: MESSAGES ARE PRINTED AT THE OPERATORS CONSOLE.
2109 ;* "TESTNAME TEST ABORTED"
2110 ;*
2111 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER1603" AS THE MESSAGE POINTER
2112 ;* PARAMETER IN THE DPS ERROR REPORT MACRO CALL.
2113 ;*
2114 ;* COMMENTS:
2115 ;*
2116 ;*
2117 ;* SUBORDINATE ROUTINES CALLED: NONE.
2118 ;*****
2119 014070 BGNMSG ER1603
2120 014070 ER1603::
2121 014070 004567 171230 SAVE JSR R5,PREG05 ;SAVE THE CONTENTS OF THE GPRS.
2122 014074 01700 000100 ;CALL REGISTER SAVE SUBRT.
2123 014100 046700 166056 MOV #BIT06,R0 ;TRY TO CLEAR THE
2124 014104 001024 BNE 2$ ;EXT'D ERROR REPORTING FLAG
2125 ;EXIT IF FLAG NOT SET.
2126
2127 014106 PRINTB #EF0503,R1 ;PRINT BASIC MESSAGE ON OPERATORS CONSOLE.
2128 014106 010146 MOV R1,-(SP)
2129 014110 012746 005530 MOV #EF0503,-(SP)
2130 014114 012746 000002 MOV #2,-(SP)
2131 014120 010600 MOV SP,R0
2132 014122 104414 TRAP C$PNTB
2133 014124 062706 000006 ADD #6,SP
2134
2135 014130 016702 171164 MOV ERRMSG,R2 ;GET THE "TEST MESSAGE".
2136 014134 PRINTB #EF1601,R2 ;PRINT "TEST ABORTED" MESSAGE.
2137 014136 012746 005535 MOV R2,-(SP)
2138 014142 012746 000002 MOV #EF1601,-(SP)
2139 014146 010600 MOV #2,-(SP)
2140 014150 104414 MOV SP,R0
2141 014152 062706 000006 TRAP C$PNTB
2142 ADD #6,SP
2143
2144 2$: PASS ;RESTORE THE CONTENTS OF THE GPRS.
2145 014156 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2146
2147 014160 ENDMSG
2148 014160 L10004:
2149 014160 104423 TRAP C$MSG
    
```

2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182

014162
014162
014162 004567 171136
014166 032767 000100 165766
014174 001507
014176 016304 005234
014202 006203
014204 006204
014206 010446
014210 010346
014212 012746 005746
014216 012746 000003
014222 010600
014224 104414
014226 062706 000010

```
.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER6201 -
*****
; THIS IS AN ERROR REPORTING SUBROUTINE WHICH IS INTENDED FOR USE IN THE
; FRAMING ERROR AND PARITY ERROR TESTS. IT REPORTS ERROR INFORMATION
; WHEN A CHARACTER HAS BEEN READ FROM THE DUT WITH THE INCORRECT
; COMBINATION OF FRAMING AND PARITY ERROR BITS. THESE ERRORS ARE REPORTED
; ONLY IF EXTENDED ERROR REPORTING HAS BEEN REQUESTED.
;
; INPUTS: R2 - DATA BYTE READ FROM THE DUT, INCLUDING ERROR FLAGS.
; R3 - LINE NUMBER MULTIPLIED BY 2.
; R5 - MESSAGE FLAGS, WHICH MESSAGES TO REPORT.
; BIT1 AND BIT3 - INDICATE WHICH MESSAGES ARE TO BE
; REPORTED, FRAMING OR PARITY RESPECTIVELY.
; BIT0 AND BIT 2 - "SET"/"CLEAR" MESSAGE FOR
; FRAMING AND PARITY ERRORS BITS.
;
; OUTPUTS: MESSAGES ARE PRINTED AT THE OPERATOR CONSOLE.
;
; CALLING SEQUENCE: INCLUDE THE LABEL "ER6201" AS THE MESSAGE POINTER
; PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
;
; COMMENTS: THE MESSAGE IS PRINTED /S BASIC AND EXTENDED ERROR INFORMATION.
; THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD OF THE DUT
; CSR MAY BE ALTERED.
;
; SUBORDINATE ROUTINES USED: PRTLPR.
*****
```

```
BGNMSG ER6201
ER6201::
SAVE R5,PREG05 ;SAVE THE CONTENTS OF THE GPR'S.
;CALL REGISTER SAVE SUBRT.

; EXIT 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.

MOV TXRXLB(R3),R4 ;GET THE ASSOCIATED TX LINE NUMBER.
ASR R3 ;CALCULATE THE RX LINE NUMBER.
ASR R4 ;CALCULATE THE ASSOCIATED LINE NUMBER.
PRINTB #EF6201,R3,R4 ;REPORT THE ERROR TYPE AND LINE NUMBERS.
MOV R4,-(SP)
MOV R3,-(SP)
MOV #EF6201,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C$PNTB
ADD #10,SP

; REPORT FRAMING ERROR PROBLEM.
MOV #EM6202,R4 ;SELECT THE "ERROR BIT CLEAR" MESSAGE.
MOV #EM0509,R1 ;SELECT EXPECTED "ERROR BIT SET" MESSAGE.
BIT #BIT1,R5 ;TEST IF FRAMING ERROR MESSAGE TO BE REPORTED.
```

C5

2183	014246	001427		BEQ	6:						
2184	014250	032705	000001	BIT		#BIT0,R5					
2185	014254	001403		BEQ		2:					
2186	014256	010401		MOV		R4,R1					
2187	014260	012704	010105	MOV		#EM0509,R4					
2188	014264			2:	PRINTX	#EF6202,R4,R1					
	014264	010146								MOV	R1,-(SP)
	014266	010446								MOV	R4,-(SP)
	014270	012746	006061							MOV	#EF6202,-(SP)
	014274	012746	000003							MOV	#3,-(SP)
	014300	010600								MOV	SP,R0
	014302	104415								TRAP	C1PNTX
	014304	062706	000010							ADD	#10,SP
2189	014310	032705	000010	BIT		#BIT3,R5					
2190	014314	001424		BEQ		10:					
2191	014316	012704	011106	MOV		#EM6202,R4					
2192	014322	012701	010105	MOV		#EM0509,R1					
2193											
2194											
2195											
2196											
2197	014326	032705	000004	6:	BIT	#BIT2,R5					
2198	014332	001403		BEQ		8:					
2199	014334	010401		MOV		R4,R1					
2200	014336	012704	010105	MOV		#EM0509,R4					
2201	014342			8:	PRINTX	#EF6203,R4,R1					
	014342	010146									
	014344	010446									
	014346	012746	006157								
	014352	012746	000003								
	014356	010600									
	014360	104415									
	014362	062706	000010								
2202											
2203	014366			10:	PRINTX	#EF1603,R2					
	014366	010246									
	014370	012746	005567								
	014374	012746	000002								
	014400	010600									
	014402	104415									
	014404	062706	000006								
2204											
2205	014410	004767	006152								
2206	014414			60:	PASS						
	014414	004736									
2207	014416				ENDMSG						
	014416										
	014416	104423									

L10005: TRAP C1MSG

```

2209 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9001 -
2210 ;*****
2211 ;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH REPORTS AN UNEXPECTED
2212 ;* CODE WHICH HAS BEEN FOUND IN THE DUT CSR. THIS CODE CAN BE A BMP
2213 ;* CODE, A SELF-TEST CODE, OR A MODEM STATUS CODE.
2214 ;*
2215 ;* INPUTS: R1 - ADDRESS OF MESSAGE TO PRINT FIRST.
2216 ;* R2 - SINGLE BYTE CODE WHICH HAS BEEN READ FROM THE DUT.
2217 ;* R4 - LINE NUMBER ASSOCIATED WITH THE CODE.
2218 ;*
2219 ;* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
2220 ;*
2221 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9001" AS THE MESSAGE POINTER
2222 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2223 ;*
2224 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
2225 ;*
2226 ;* SUBORDINATE ROUTINES USED: NONE.
2227 ;*****
2228
2229 014420 BGNMSG ER9001
2230
2231
2232
2233
2234 014420 032767 000100 165534 BIT #BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
2235 014426 001433 BEQ 2; ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
2236 ;DURING THE SOFTWARE QUESTIONS.
2237
2238 014430 PRINTB #EF9001,R1 ;REPORT TYPE OF CODE FOUND.
2239 014430 010146 MOV R1,-(SP)
2240 014432 012746 006254 MOV #EF9001,-(SP)
2241 014436 012746 000002 MOV #2,-(SP)
2242 014442 010600 MOV SP,R0
2243 014444 104414 TRAP C#PNTB
2244 014446 062706 000006 ADD #6,SP
2245 014452 PRINTX #EF9002,R4 ;REPORT THE LINE NUMBER OF THE CODE.
2246 014452 010446 MOV R4,-(SP)
2247 014454 012746 006336 MOV #EF9002,-(SP)
2248 014460 012746 000002 MOV #2,-(SP)
2249 014464 010600 MOV SP,R0
2250 014466 104415 TRAP C#PNTX
2251 014470 062706 000006 ADD #6,SP
2252 014474 PRINTX #EF9003,R2 ;REPORT THE CODE WHICH WAS FOUND.
2253 014474 010246 MOV R2,-(SP)
2254 014476 012746 006410 MOV #EF9003,-(SP)
2255 014502 012746 000002 MOV #2,-(SP)
2256 014506 010600 MOV SP,R0
2257 014510 104415 TRAP C#PNTX
2258 014512 062706 000006 ADD #6,SP
2259
2260
2261
2262 014516 2;: ENDMMSG
2263 014516 L10006: TRAP C#MSG
2264 014516 104423

```

```

2244 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9002 -
2245 ;*****
2246 ;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH IS INTENDED FOR USE IN THE
2247 ;* TRANSMISSION AND RECEPTION TESTS. IT REPORTS THE TYPE OF ERROR WHICH
2248 ;* HAS OCCURRED WHEN INCORRECT DATA IS RECEIVED FROM THE OUT. THIS
2249 ;* ROUTINE ALSO REPORTS THE READ AND EXPECTED DATA VALUES.
2250 ;*
2251 ;* INPUTS: R1 - ADDRESS OF MESSAGE TO PRINT FIRST.
2252 ;* R2 - DATA BYTE READ FROM THE OUT.
2253 ;* R3 - LINE NUMBER MULTIPLIED BY 2.
2254 ;* R4 - EXPECTED DATA BYTE, BIT 15 SET IF "NONE".
2255 ;*
2256 ;* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
2257 ;*
2258 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9002" AS THE MESSAGE POINTER
2259 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2260 ;*
2261 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
2262 ;*
2263 ;* SUBORDINATE ROUTINES USED: PRTLPR.
2264 ;*****
2265
2266 014520 BGNMSG ER9002
2267 014520 ER9002::
2268
2269 ;*
2270 ;* EXIT IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
2271 014520 032767 000100 165434 BIT #BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
2272 014526 001462 BEQ 62# ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
2273 ;* DURING THE SOFTWARE QUESTIONS.
2274
2275 014530 006203 ASR R3 ;CALCULATE THE LINE NUMBER.
2276 014532 042702 177400 BIC #177400,R2 ;MASK OUT ALL BUT DATA IN READ CHAR.
2277 014536 PRINTB #EF9006,R1,R3 ;PRINT THE FIRST LINE OF THE MESSAGE.
2278 014536 010346 MOV R3,-(SP)
2279 014540 010146 MOV R1,-(SP)
2280 014542 012746 006520 MOV #EF9006,-(SP)
2281 014546 012746 000003 MOV #3,-(SP)
2282 014552 010600 MOV SP,R0
2283 014554 104414 TRAP C:PNTB
2284 014556 062706 000010 ADD #10,SP
2285 014562 PRINTX #EF9004,#EM9010,R2 ;PRINT ACTUAL DATA.
2286 014564 012746 011476 MOV R2,-(SP)
2287 014570 012746 006437 MOV #EM9010,-(SP)
2288 014574 012746 000003 MOV #EF9004,-(SP)
2289 014600 010600 MOV #3,-(SP)
2290 014602 104415 MOV SP,R0
2291 014604 062706 000010 TRAP C:PNTX
2292 014610 005704 ADD #10,SP
2293 014612 100414 TST R4 ;CHECK FOR "NONE" CODE SET IN EXPECTED DATA.
2294 014614 PRINTX #EF9004,#EM9009,R4 ;BRANCH TO PRINT "NONE" MESSAGE IF FLAG SET.
2295 014614 010446 MOV R4,(SP)
2296 014616 012746 011452 MOV #EM9009,-(SP)
2297 014622 012746 006437 MOV #EF9004,-(SP)
2298 014626 012746 000003 MOV #3,-(SP)

```



```

014632 010600
014634 104415
014636 062706 000010
2282 014642 000412
2283 014644
014644 012746 011452
014650 012746 006467
014654 012746 000002
014660 010600
014662 104415
014664 062706 000006
2284 014670 004767 005672
2285 014674
014674
014674 104423

2$: BR 60$ ;EXIT THIS ROUTINE.
PRINTX #EF9005,#EM9009 ;PRINT MESSAGE INDICATING NO EXPECTED DATA.
MOV #EM9009,-(SP)
MOV #EF9005,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C$PNTX
ADD #6,SP

60$: JSR PC,PRTLPR ;PRINT CONTENTS OF THE LPR REGISTER.
62$: ENDMSG

L10007: TRAP C$MSG

```

2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330

014676
014676

014676 032767 000100 165256
014704 001470

014706 006203
014710
014710 C10346
014712 012746 006544
014716 012746 000002
014722 010600
014724 104414
014726 062706 000006
014732 010201
014734 042701 177400
014740
014740 010146
014742 012746 006640
014746 012746 000002
014752 010600
014754 104415
014756 062706 000006

```
.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9003 -
*****
; THIS IS AN ERROR REPORTING SUBROUTINE WHICH IS INTENDED FOR USE IN THE
; TRANSMISSION AND RECEPTION TESTS. IT REPORTS ERROR INFORMATION WHEN
; A CHARACTER HAS BEEN READ FROM THE DUT WITH AN ERROR FLAG OR FLAGS
; SET (IE. OVER-RUN, FRAMING, OR PARITY FLAG).
;
; INPUTS: R2 - DATA BYTE READ FROM THE DUT, INCLUDING ERROR FLAGS.
; R3 - LINE NUMBER MULTIPLIED BY 2.
;
; OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
;
; CALLING SEQUENCE: INCLUDE THE LABEL "ER9003" 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 THE INDIRECT ADDRESS REGISTER FIELD OF THE DUT
; CSR MAY BE ALTERED.
;
; SUBORDINATE ROUTINES USED: NONE.
*****
```

BGNMSG ER9003

ER9003::

```
;;
; EXIT THE TEST IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
;
; BIT 0BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
; BEQ 62$ ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
; DURING THE SOFTWARE QUESTIONS.
;
; ASR R3 ;CALCULATE THE LINE NUMBER.
; PRINTB 0EF9007,R3 ;REPORT THE ERROR TYPE AND LINE NUMBER.
; MOV R3, -(SP)
; MOV 0EF9007, -(SP)
; MOV 02, -(SP)
; MOV SP,R0
; TRAP C$PNTB
; ADD 06,SP
;
; MOV R2,R1 ;EXTRACT THE RECEIVED CHARACTER FROM THE
; BIC 0177400,R1 ; PASSED IN CHAR VALUE WITH FLAGS.
; PRINTX 0EF9008,R1 ;REPORT THE VALUE OF THE RECEIVED CHAR.
; MOV R1, -(SP)
; MOV 0EF9008, -(SP)
; MOV 02, -(SP)
; MOV SP,R0
; TRAP C$PNTX
; ADD 06,SP
;
; REPORT OVERRUN FLAG SET IF NECESSARY.
;
; MOV 0EM9011,R1 ;SELECT THE OVERRUN ERROR MESSAGE.
; BIT 0BIT14,R2 ;CHECK OVERRUN ERROR FLAG IN PASSED IN CHAR.
; BEQ 2$ ;SKIP ERROR IF OVERRUN ERROR FLAG WAS CLEAR.
; JSR PC,50$ ;REPORT THE OVERRUN ERROR FLAG WAS SET.
;;
```

```

2331 ; REPORT FRAMING FLAG SET IF NECESSARY.
2332 ;
2333 015000 012701 011532 2$: MOV    $EM9012,R1    ;SELECT THE FRAMING ERROR MESSAGE.
2334 015004 032702 020000      BIT    $BIT13,R2    ;CHECK FRAMING ERROR FLAG IN PASSED IN CHAR.
2335 015010 001402              BEQ    4$            ;SKIP ERROR IF FRAMING ERROR FLAG WAS CLEAR.
2336 015012 004767 000020      JSR    PC,50$       ;REPORT THE FRAMING ERROR MESSAGE.
2337 ;
2338 ; REPORT PARITY FLAG SET IF NECESSARY.
2339 ;
2340 015016 012701 011542 4$: MOV    $EM9013,R1    ;SELECT THE PARITY ERROR MESSAGE.
2341 015022 032702 010000      BIT    $BIT12,R2    ;CHECK PARITY ERROR FLAG IN PASSED IN CHAR.
2342 015026 001415              BEQ    60$            ;EXIT ROUTINE IF PARITY ERRO FLAG WAS CLEAR.
2343 015030 004767 000002      JSR    PC,50$       ;REPORT THE PARITY ERROR MESSAGE.
2344 015034 000412              BR     60$            ;EXIT THIS ROUTINE.
2345 ;
2346 ;
2347 ; LOCAL SUBROUTINE TO REPORT AN ERROR FLAG STATUS.
2348 ;
2349 015036 50$: PRINTX $EF9009,R1
      015036 010146
      015040 012746 006677      MOV    R1,-(SP)
      015044 012746 000002      MOV    $EF9009,-(SP)
      015050 010600              MOV    $2,-(SP)
      015052 104415              MOV    SP,R0
      015054 062706 000006      TRAP  C:PNTX
      015060 000207              ADD   $6,SP
2350      RTS    PC
2351 ;
2352 015062 004767 005500 60$: JSR    PC,PRTLPR    ;REPORT THE LPR CONTENTS FOR THIS LINE.
2353 015066 62$: ENDMSG
      015066
      015066 104423      L10010: TRAP  C$MSG
    
```

```

2355 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9004 -
2356 ;*****
2357 ;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH REPORTS ERROR SUMMARIES
2358 ;* FOR LINES WHICH HAVE EXCEEDED THE SPECIFIED MAXIMUM NUMBER OF
2359 ;* INDIVIDUAL RECEPTION ERRORS, PROVIDED EXTENDED ERROR REPORTING HAS
2360 ;* BEEN REQUESTED BY THE OPERATOR.
2361 ;*
2362 ;* INPUTS: R1 - ADDRESS OF MESSAGE TO PRINT FIRST,
2363 ;* ERCNTB - LABEL AT BASE OF LINE ERROR COUNTERS TABLE,
2364 ;* ERSMRF - "REPORT ERROR SUMMARY FOR LINE" FLAGS.
2365 ;*
2366 ;* OUTPUTS: A MESSAGE IS PRINTED AT THE OPERATOR CONSOLE.
2367 ;*
2368 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9004" AS THE MESSAGE POINTER
2369 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2370 ;*
2371 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION,
2372 ;* THE CONTENTS OF GPR'S R2, R3, R4, AND R5 ARE DESTROYED.
2373 ;*
2374 ;* SUBORDINATE ROUTINES USED: NONE.
2375 ;*****
2376
2377 015070 BGNMSG ER9004
2378 015070 ER9004::
2378 015070 012700 000100 MOV #BIT06,R0 ;TRY TO CLEAR THE
2379 015074 046700 165062 BIC OPTION,R0 ;EXT'D ERROR REPORTING FLAG
2380 015100 001040 BNE 6$ ;EXIT IF FLAG NOT SET,
2381 015102 PRINTB #EF0503,#EM9014 ;REPORT THE SECONDARY ERROR MESSAGE.
2381 015102 012746 011551 MOV #EM9014,-(SP)
2381 015106 012746 005530 MOV #EF0503,-(SP)
2381 015112 012746 000002 MOV #2,-(SP)
2381 015116 010600 MOV SP,R0
2381 015120 104414 TRAP C$PNTB
2381 015122 062706 000006 ADD #6,SP
2382 015126 005002 CLR R2 ;CLEAR THE LINE COUNTER.
2383 015130 016703 165344 MOV ERSMRF,R3 ;GET THE ERROR SUMMARY FLAGS.
2384 015134 005004 CLR R4 ;CLEAR "LINE COUNTER TIMES 2" OFFSET.
2385 015136 000241 2$: CLC ;CLEAR THE CARRY FOR THE FOLLOWING ROTATE.
2386 015140 006003 ROR R3 ;SHIFT ANOTHER ERROR SUMMARY FLAG INTO CARRY.
2387 015142 103013 BCC 4$ ;SKIP PRINTING MESSAGE IF FLAG FOR LINE CLEAR.
2388 015144 PRINTX #EF9010,R2,ERCNTB(R4)
2388 015144 016446 003302 MOV ERCNTB(R4),-(SP)
2388 015150 010246 MOV R2,-(SP)
2388 015152 012746 006736 MOV #EF9010,-(SP)
2388 015156 012746 000003 MOV #3,-(SP)
2388 015162 010600 MOV SP,R0
2388 015164 104415 TRAP C$PNTX
2388 015166 062706 000010 ADD #10,SP
2389 015172 012405 4$: MOV (R4)+,R5 ;INCREMENT THE LINE OFFSET BY 2.
2390 015174 005202 INC R2 ;INCREMENT THE LINE COUNTER.
2391 015176 005703 TST R3 ;CHECK THE ERROR SUMMARY FLAGS.
2392 015200 001356 BNE 2$ ;IF MORE FLAGS SET, LOOP TO DO OTHER LINES.
2393 015202 6$: ENDMMSG
2393 015202 104423 L10011: TRAP C$MSG
    
```

```

2395 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9005 -
2396 ;*****
2397 ; THIS IS AN ERROR REPORTING SUBROUTINE WHICH REPORTS INCOMPLETE DATA
2398 ; TRANSMISSIONS OR RECEPTIONS.
2399 ;*
2400 ;* INPUTS: R1 - EITHER "TRANSMITTED" OR "RECEIVED" TO INDICATE TX OR RX.
2401 ;* R2 - BIT MAP OF LINES WHICH DID NOT COMPLETE TX OR RX.
2402 ;* R4 - ADDRESS OF BASE OF THE CORRECT CHARACTER COUNTERS TABLE.
2403 ;* DPLENB - LABEL AT BASE OF DATA PATTERN LENGTH TABLE.
2404 ;* EM9015 - SYMBOLIC ADDRESS OF THE "TRANSMITTED" MESSAGE.
2405 ;*
2406 ;* OUTPUTS: A MESSAGE IS PRINTED AT THE OPERATOR CONSOLE.
2407 ;*
2408 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9005" AS THE MESSAGE POINTER
2409 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2410 ;*
2411 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC AND EXTENDED ERROR INFORMATION.
2412 ;* THE CONTENTS OF THE INDIRECT ADDRESS FIELD IN THE OUT CSR MAY
2413 ;* BE ALTERED.
2414 ;*
2415 ;* SUBORDINATE ROUTINES USED: PRTLPR.
2416 ;*****
2417
2418 015204 BGNMSG ER9005
2419 015204 ER9005:;
015204 SAVE ;SAVE THE CONTENTS OF THE GPR'S.
015204 004567 170114 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2420
2421 015210 012700 000100 MOV #BIT06,R0 ;TRY TO CLEAR THE
2422 015214 046700 164742 BIC OPTION,R0 ;EXT'D ERROR REPORTING FLAG
2423 015220 001107 BNE 10$ ;EXIT IF FLAG NOT SET.
2424
2425 015222 PRINTB #EF9013,R1 ;REPORT THE SECONDARY ERROR MESSAGE.
015222 010146 MOV R1,-(SP)
015224 012746 007141 MOV #EF9013,-(SP)
015230 012746 000002 MOV #2,-(SP)
015234 010600 MOV SP,R0
015236 104414 TRAP C$PNTB
015240 062706 000006 ADD #6,SP
2426 015244 005003 CLR R3 ;CLEAR THE LINE COUNTER.
2427 015246 022701 011645 CMP #EM9015,R1 ;CHECK IF ADDRESS CORRESPONDS TO TX MESSAGE.
2428 015252 001032 BNE 6$ ;BRANCH IF RECEPTION MESSAGE TO BE PRINTED.
2429
2430 ;
2431 ; PERFORM TX INCOMPLETE ERROR MESSAGE REPORTING.
2432 ;
2433 015254 PRINTX #EM9030 ;PRINT "NO TX COMPLETION INTERRUPTS RECEIVED"
015254 012746 012260 MOV #EM9030,-(SP)
015260 012746 000001 MOV #1,-(SP)
015264 010600 MOV SP,R0
015266 104415 TRAP C$PNTX
015270 062706 000004 ADD #4,SP
2434 015274 000241 2$: CLC ;CLEAR THE CARRY FOR THE FOLLOWING ROTATE.
2435 015276 006002 ROR R2 ;SHIFT "TX NOT DONE" FLAG INTO CARRY.
2436 015300 103013 BCC 4$ ;SKIP PRINTING MESSAGE IF FLAG FOR LINE CLEAR.
2437 015302 PRINTX #EF9020,R3 ;PRINT "TOO FEW TX ACTIONS GENERATED" MSG.
015302 010346 MOV R3,-(SP)

```

```

015304 012746 007225                                MOV    #EF9020,-(SP)
015310 012746 000002                                MOV    #2,-(SP)
015314 010600                                        MOV    SP,RO
015316 104415                                        TRAP   C$PNTX
015320 062706 000006                                ADD    #6,SP
2438 015324 004767 005236                          JSR    PC,PRTLPR ;REPORT CONTENTS OF LPR REGISTER FOR THIS LINE.
2439 015330 005203 4$: INC R3 ;INCREMENT LINE COUNTER.
2440 015332 005702 TST R2 ;CHECK THE "TX NOT DONE FLAGS".
2441 015334 001357 BNE 2$ ;IF MORE FLAGS SET, LOOP TO DO OTHER LINES.
2442 015336 000440 BR 10$ ;EXIT THIS ROUTINE.
2443 ;
2444 ; PERFORM RX INCOMPLETE ERROR MESSAGE REPORTING.
2445 ;
2446 015340 000241 6$: CLC ;CLEAR THE CARRY FOR THE FOLLOWING ROTATE.
2447 015342 006002 ROR R2 ;SHIFT "RX NOT DONE" FLAG INTO CARRY.
2448 015344 103031 BCC 8$ ;SKIP PRINTING MESSAGE IF FLAG FOR LINE CLEAR.
2449 015346 006303 ASL R3 ;SHIFT LINE # TO GIVE CORRECT TABLE OFFSET.
2450 015350 016305 005234 MOV 1XRXL8(R3),R5 ;GET THE "ASSOCIATED" RECEIVE LINE OFFSET.
2451 015354 010246 MOV R2,-(SP) ;SAVE THE "RX NOT DONE" FLAGS ON THE STACK.
2452 015356 010502 MOV R5,R2 ;COPY THE ASSOCIATED TX LINE OFFSET.
2453 015360 016505 003442 MOV CHCNTB(R5),R5 ;GET THE TOTAL NUMBER OF EXPECTED CHARS.
2454 015364 006202 ASR R2 ;SHIFT THE TABLE OFFSET TO GIVE A LINE NUMBER.
2455 015366 006203 ASR R3 ;SHIFT TABLE OFFSET TO GIVE LINE NUMBER.
2456 015370 PRINTX #EF9012,R3,R1,(R4),R5,R2 ;REPORT NUMBER OF CHARS ON LINE.
015370 010246 MOV R2,-(SP)
015372 010546 MOV R5,-(SP)
015374 011446 MOV (R4),-(SP)
015376 010146 MOV R1,-(SP)
015400 010346 MOV R3,-(SP)
015402 012746 007025 MOV #EF9012,-(SP)
015406 012746 000006 MOV #6,-(SP)
015412 010600 MOV SP,RO
015414 104415 TRAP C$PNTX
015416 062706 000016 ADD #16,SP
2457 015422 012602 MOV (SP)+,R2 ;RESTORE THE "RX NOT DONE" FLAGS.
2458 015424 004767 005136 JSR PC,PRTLPR ;REPORT CONTENTS OF LPR REGISTER FOR THIS LINE.
2459 015430 005724 8$: TST (R4)+ ;INCREMENT THE CHARACTER COUNTER TABLE.
2460 015432 005203 INC R3 ;INCREMENT THE LINE COUNTER.
2461 015434 005702 TST R2 ;CHECK THE "RX NOT DONE FLAGS".
2462 015436 001340 BNE 6$ ;IF MORE FLAGS SET, LOOP TO DO OTHER LINES.
2463 015440 10$: PASS ;RESTORE THE CONTENTS OF THE GPRS.
015440 004736 JSR PC,@(SP)+ ;RETURN TO PREGO5 SUBRT.
2464 015442 ENDMMSG
015442 104423 L10012: TRAP C$MSG

```

```

2466 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9101 -
2467 ;*****
2468 ;* THIS IS A GENERAL ERROR REPORTING SUBROUTINE WHICH REPORTS A MESSAGE
2469 ;* WHICH TAKES A SINGLE, 2 DIGIT DECIMAL ARGUMENT AFTER THE END OF AN
2470 ;* ASCII MESSAGE.
2471 ;*
2472 ;* INPUTS: R1 - VALUE TO BE PRINTED AFTER MSG AS 2 DECIMAL DIGITS.
2473 ;* R2 - ADDRESS OF MESSAGE TO PRINT FIRST.
2474 ;*
2475 ;* OUTPUTS: A MESSAGES IS PRINTED AT THE OPERATOR CONSOLE.
2476 ;*
2477 ;* CALLING SEQUENCE: INCLUDE THE LABEL "ER9101" AS THE MESSAGE POINTER
2478 ;* PARAMETER IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2479 ;*
2480 ;* COMMENTS: THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION.
2481 ;*
2482 ;* SUBORDINATE ROUTINES USED: NONE.
2483 ;*****
2484
2485 015444 BGNMSG ER9101
2486 015444 ER9101::
2487 015444 012700 000100 MOV #BIT06,R0 ;TRY TO CLEAR THE
2488 015450 046700 164506 BIC OPTION,R0 ;EXT'D ERROR REPORTING FLAG
2489 015454 001012 BNE 2$ ;EXIT IF FLAG NOT SET.
2490
2491
2492 015456 PRINTB #EF9006,R2,R1 ;REPORT THE STRING FOLLOWED BY THE NUMBER.
2493 015456 010146 MOV R1,-(SP)
2494 015460 010246 MOV R2,-(SP)
2495 015462 012746 006520 MOV #EF9006,-(SP)
2496 015466 012746 000003 MOV #3,-(SP)
2497 015472 010600 MCV SP,R0
2498 015474 104414 TRAP C$PNTB
2499 015476 062706 000010 ADD #10,SP
2500
2501 2$: ENDMSG
2502
2503 L10013:
2504 015502 TRAP C$MSG
2505 015502 104423

```

```

2496 .SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9102 -
2497 ;*****
2498 ;* THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS ADDITIONAL ERROR
2499 ;* INFORMATION AFTER THE ERROR MESSAGE HEADER, PROVIDED EXTENDED ERROR
2500 ;* REPORTING HAS BEEN REQUESTED.
2501 ;* THIS ROUTINE IS PASSED A BIT MAP WHICH SPECIFIES THE LINES FOR WHICH
2502 ;* THE ERROR CONDITION SHOULD BE REPORTED.
2503 ;*
2504 ;* INPUTS: R1 - ADDRESS OF THE MESSAGE TO BE PRINTED BY THIS ROUTINE.
2505 ;* R2 - BIT MAP OF LINES FOR WHICH TO REPORT ERRORS.
2506 ;*
2507 ;* OUTPUTS: MESSAGES ARE PRINTED AT THE OPERATOR CONSOLE.
2508 ;*
2509 ;* CALLING SEQUENCE: LOAD THE ADDRESS OF THE MESSAGE IN R1.
2510 ;* LOAD THE BIT MAP OF LINES WITH ERRORS IN R2.
2511 ;* INCLUDE THE LABEL "ER9102" AS THE MESSAGE POINTER
2512 ;* (ERRBLK) IN THE DIAG SUPER ERROR REPORT MACRO CALL.
2513 ;*
2514 ;* COMMENTS: THE OUTPUT FORMAT OF THIS MESSAGE IS:
2515 ;* "TEXT MESSAGE POINTED TO BY R1"
2516 ;* "ERROR CONDITION ON LINE NN"
2517 ;* "ERROR CONDITION ON LINE ..."
2518 ;* THE TOP MESSAGE, AND THE MESSAGE FOR EACH LINE ARE PRINTED
2519 ;* AS BASIC ERROR INFORMATION.
2520 ;*
2521 ;* SUBORDINATE ROUTINES USED: NONE.
2522 ;*****
2523
2524 015504 BGNMSG ER9102
2525 015504 ER9102:
015504 SAVE ;SAVE THE CONTENTS OF THE GPRS.
015504 004567 167614 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2526 ;*
2527 ; EXIT IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
2528 ;*
2529 015510 032767 000100 164444 BIT #BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
2530 015516 001441 BEQ 60$ ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
2531 ; DURING THE SOFTWARE QUESTIONS.
2532
2533 015520 PRINTB #EF0503,R1 ;PRINT THE FIRST LINE OF THE MESSAGE.
015520 010146 MOV R1, -(SP)
015522 012746 005530 MOV #EF0503, -(SP)
015526 012746 000002 MOV #2, -(SP)
015532 010600 MOV SP,R0
015534 104414 TRAP C#PNTB
015536 062706 000006 ADD #6,SP
2534 015542 005003 CLR R3 ;CLEAR THE LINE NUMBER.
2535 015544 000241 CLC ;PREPARE TO ROTATE NEXT BIT OUT OF MAP.
2536 015546 006002 ROR R2 ;GET THE NEXT BIT OF THE BIT MAP.
2537 015550 103011 BCC 4$ ;SKIP PRINTING MESSAGE IF THE BIT IS CLEAR.
2538 015552 PRINTB #EF9103,R3 ;REPORT THIS LINE HAD THE ERROR.
015552 010346 MOV R3, (SP)
015554 012746 007311 MOV #EF9103, -(SP)
015560 012746 000002 MOV #2, -(SP)
015564 010600 MOV SP,R0
015566 104414 TRAP C#PNTB
015570 062706 000006 ADD #6,SP

```



```

2539 015574 005203      4$:      INC      R3          ;INCREMENT THE LINE COUNTER.
2540 015576 005702      TST      R2          ;CHECK THE BIT MAP.
2541 015600 001361      BNE      2$          ;LOOP IF NOT ALL SET BITS REMOVED FROM BIT MAP.
2542 015602      PRINTB  #EF9101 ;PRINT A BLANK LINE.
      015602 012746 007306      MOV      #EF9101,-(SP)
      015606 012746 000001      MOV      #1,-(SP)
      015612 010600      MOV      SP,R0
      015614 104414      TRAP    C$PNTB
      015616 062706 000004      ADD     #4,SP
2543 015622      50$:      PASS          ;RESTORE THE SAVED CONTENTS OF THE GPRS.
      015622 004736      JSR     PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2544 015624      ENDMSG
      015624      L10014:
      015624 104423      TRAP    C$MSG
    
```

2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567 015626
015626
2568 015626 004567 167472
015626
2569
2570 015632 012700 000100
2571 015636 046700 164320
2572 015642 001064
2573
2574 015644
015644 010146
015646 012746 005530
015652 012746 000002
015656 010600
015660 104414
015662 062706 000006
2575 015666 012703 002512
2576 015672 012705 012651
2577 015676 012301
2578 015700 012304
2579 015702 004767 000056
2580 015706 020302
2581 015710 103772
2582
2583
2584
2585
2586
2587
2588 015712 020227 002706
2589 015716 001036
2590 015720 005762 000002
2591 015724 001433
2592 015726 012301
2593 015730 011304
2594 015732 012705 012701

```
.SBTTL GLOBAL ERROR REPORTING ROUTINE - ER9301 -
*****
; THIS IS AN ERROR REPORTING SUBROUTINE WHICH PRINTS ANY BMP CODES
; THAT ARE FOUND IN THE BMP CODE QUEUE, TOGETHER WITH THE 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 MACRO CALL.
;
; COMMENTS: THE MESSAGE IS PRINTED AS BASIC ERROR INFORMATION.
;
; SUBORDINATE ROUTINES USED: NONE.
*****

BGNMSG ER9301
ER9301::
SAVE R5, PREG05 ;SAVE THE GPRS ON THE STACK.
JSR ER9301 ;CALL REGISTER SAVE SUBRT.

MOV #BIT06, R0 ;TRY TO CLEAR THE
BIC OPTION, R0 ;EXT'D ERROR REPORTING FLAG
BNE 601 ;EXIT IF FLAG NOT SET.

PRINTB #EF0503, R1 ;REPORT UNEXPECTED BMP CODES FOUND.
MOV R1, -(SP)
MOV #EF0503, -(SP)
MOV #2, -(SP)
MOV SP, R0
TRAP C:PNTB
ADD #6, SP

MOV #BMPCQB, R3 ;GET THE START ADDRESS OF THE BMP CODE QUEUE.
MOV #EM9302, R5 ;GET THE MESSAGE TO BE REPORTED.
2$: MOV (R3)+, R1 ;GET THE NUMBER OF THE TEST THAT WAS EXECUTING.
MOV (R3)+, R4 ;GET BMP CODE THAT WAS REPORTED OFF THE QUEUE.
JSR PC, 50$ ;GO REPORT THE BMP CODE.
CMP R3, R2 ;CHECK IF ALL CODES HAVE BEEN REPORTED.
BLO 2$ ;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.
;
1$: CMP R2, #BMPCQE-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 OCCURED, 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.
```

C6

```

2595 015736          PRINTX  #EF9302          ;REPORT OVERFLOW CONDITION.
      015736 012746 007435
      015742 012746 000001
      015746 010600
      015750 104415
      015752 062706 000004
2596 015756          JSR      PC,50$          ;REPORT THE LAST BMP CODE PLACED ON THE QUEUE.
2597 015762          BR       60$
2598
2599 015764          50$: PRINTX  #EF9301,R5,R1,R4 ;PRINT THE MESSAGE.
      015764 010446
      015766 010146
      015770 010546
      015772 012746 007357
      015776 012746 000004
      016002 010600
      016004 104415
      016006 062706 000012
2600 016012          60$: RTS      PC          ;RETURN.
2601 016014          PASS
      016014 004736          JSR      PC,@(SP)+ ;RESTORE THE GPR CONTENTS.
2602
2603 016016          ENDMSG
      016016
      016016 104423          L10015: TRAP  C$MSG

```

D6

2605
2613
2614
2615
2616
2617
2618

.SBTTL GLOBAL SUBROUTINES SECTION

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

```

2620 .SBTTL GLOBAL SUBROUTINE - ALTFLD -
2621 ;* *****
2622 ;* - ALTER DEVICE REGISTER FIELDS ROUTINE -
2623 ;* THIS SUBROUTINE ALTERS THE SPECIFIED FIELD OF THE SPECIFIED DEVICE
2624 ;* REGISTER FOR THE SPECIFIED LINES. THIS ROUTINE CAN BE USED TO SET
2625 ;* OR CLEAR BITS WITHIN SELECTED FIELDS OF SELECTED REGISTERS.
2626 ;* USE EXAMPLES: SET RX,BAUD,RATE FIELDS ON LINES 3 AND 6.
2627 ;* CLEAR TX,DMA BITS ON ALL LINES.
2628 ;*
2629 ;* INPUTS: R1 - ADDRESS OF THE REGISTERS TO ALTER.
2630 ;* R2 - BIT FIELDS SET TO DESIRED STATES.
2631 ;* R3 - BIT MAP OF LINES FOR WHICH TO ALTER REGISTER.
2632 ;* R4 - MASK OF BITS TO ALTER (1 INDICATES CHANGE BIT).
2633 ;* CSRA - CONTAINS THE ADDRESS OF THE DEVICE CSR.
2634 ;* IESTAT - SAVED STATES OF THE INTERRUPT ENABLE BITS.
2635 ;*
2636 ;* OUTPUTS: DEVICE REGISTERS - SPECIFIED REGISTER FIELDS ALTERED.
2637 ;* CSR IND.ADR.REG FIELD - DESTROYED.
2638 ;*
2639 ;* CALLING SEQUENCE: JSR PC,ALTFLD
2640 ;*
2641 ;* COMMENTS: THIS ROUTINE READS THE SPECIFIED REGISTERS FOR ALL LINES
2642 ;* WITH NUMBERS LOWER THAN THE HIGHER SPECIFIED LINE.
2643 ;* THIS ROUTINE DOES NOT READ THE CSR.
2644 ;*
2645 ;* SUBROUTINES CALLED: NONE.
2646 ;* - - - - -
2647
2648 016020 ALTFLD:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016020 004567 167300 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2649
2650 ;*
2651 ;* SET UP TO LOOP FOR EACH LINE:
2652 ;* PREPARE THE WORD TO BE ORED INTO THE REGISTER CONTENTS.
2653 ;* SET UP THE WORD TO WRITE INTO THE IND.ADR.REG FIELD OF THE CSR.
2654 ;* -
2655 016024 010400 MOV R4,R0 ;CALCULATE THE NEW CONTENTS OF THE
2656 016026 005100 COM R0 ; REGISTER FIELDS WHICH ARE TO BE
2657 016030 040002 BIC R0,R2 ; ALTERED BY THIS ROUTINE.
2658 016032 016705 164176 MOV IESTAT,R5 ;SET UP TO WRITE IND.ADR.REG FIELD TO 0.
2659 ;*
2660 ;* LOOP ONCE FOR EACH LINE, ALTERING THE SPECIFIED FIELD IN THE SPECIFIED
2661 ;* REGISTER IF THE LINE HAS BEEN SELECTED FOR ALTERING.
2662 ;* EXIT THE LOOP IF NO MORE LINES TO ALTER, OR IF WE HAVE ALTERED THE MAX
2663 ;* ALLOWABLE NUMBER OF LINES (AS SPECIFIED BY NUMLNS).
2664 ;* -
2665 016036 000241 CLC ;PREPARE FOR ROTATE, "TST R5" DOES THIS BELOW.
2666 016040 006003 2$: ROR R3 ;GET THE LINE SELECT BIT FOR THIS LINE.
2667 016042 103006 BCC 4$ ;SKIP SETUP IF LINE IS NOT SELECTED.
2668 016044 010577 164130 MOV R5,0CSRA ;SET OUT CSR IND.ADR.REG FIELD TO THIS LINE.
2669 016050 011100 MOV (R1),R0 ;GET THE PRESENT CONTENTS OF THE REG TO ALTER.
2670 016052 040400 BIC R4,R0 ;CLEAR THE BIT FIELDS WE ARE TO ALTER.
2671 016054 050200 BIS R2,R0 ;OR IN THE NEW STATES OF THE FIELDS.
2672 016056 010011 MOV R0,(R1) ;WRITE THE NEW REGISTER CONTENTS TO THE REG.
2673 016060 005205 4$: INC R5 ;SET LINE NUMBER TO THE NEXT LINE.
2674 016062 005703 TST R3 ;CHECK FOR UNHANDLED LINES, CLEAR CARRY FLAG.
2675 016064 001365 BNE 2$ ;LOOP IF SELECTED LINE(S) IS NOT HANDLED.

```

F6

```
2676  
2677 016066          60$: PASS          ;RESTORE GPRS.  
          016066 004736          PC,@(SP)+ ;RETURN TO PREG05 SUBRT.  
2678 016070 000207          RTS    PC    JSR    ;RETURN TO CALLING ROUTNE.
```

```

2680 .SBTTL GLOBAL SUBROUTINE - CALMSL -
2681 ;* *****
2682 ;* - CALIBRATE MILLI SECOND LOOP COUNT SUBROUTINE -
2683 ;* THIS SUBROUTINE CALIBRATES THE TIMING LOOP WHICH IS USED IN THE MSLOOP
2684 ;* ROUTINE. THIS SUBROUTINE CALCULATES A VALUE FOR THE MSLCNT VARIABLE
2685 ;* WHICH IS THE NUMBER OF SOFTWARE LOOPS WHICH TAKES 1 MS TO EXECUTE IN
2686 ;* THE MSLOOP ROUTINE. THIS ROUTINE CALIBRATES THE COUNT BY USING THE
2687 ;* LINE TIME CLOCK (LTC), SO IF NO LTC IS AVAILABLE THE DEFAULT VALUE FOR
2688 ;* THE DELAY COUNT MUST BE USED.
2689 ;*
2690 ;*
2691 ;* INPUTS: MSLCNT - DEFAULT 1 MS DELAY LOOP COUNT VALUE, OR
2692 ;* VALUE FROM PREVIOUS CALIBRATION.
2693 ;* MSTICK - NUMBER OF MS PER LTC CLOCK TICK.
2694 ;* TIMER1 - TIMER COUNTER CHANGED BY LTC INTERRUPT SERVICE RTN.
2695 ;* CLKHRZ - NUMBER OF LTC CLICKS PER SECOND (50 OR 60).
2696 ;*
2697 ;* OUTPUTS: CARRY - SET IF LTC IS AVAILABLE, AND NEW CALIBRATION PERFORMED.
2698 ;* MSLCNT - NEW 1 MS DELAY LOOP COUNT VALUE IF LTC AVAILABLE, OR
2699 ;* UNCHANGED IF NO LTC IS AVAILABLE.
2700 ;*
2701 ;* CALLING SEQUENCE: JSR PC,CALMSL
2702 ;*
2703 ;* COMMENTS:
2704 ;*
2705 ;* SUBORDINATE ROUTINES CALLED: UNSDIV,OOPS.
2706 ;* - - - - -
2707
2708 016072 CALMSL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
016072 004567 167226 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
2709 016076 005067 000210 CLR 62$ ;CLEAR THE 2ND TIME FLAG.
2710 ;*
2711 ;* SYNCHRONIZE WITH THE LTC.
2712 ;* - - - - -
2713 016102 012705 000001 2$: MOV #1,R5 ;SET OUTER LOOP COUNTER TO 1 LOOP.
2714 ;* ;INCREASE THE VALUE LOADED INTO THIS COUNTER IF THE <*&
2715 ;* ;FOLLOWING LOOP FAILS ON FUTURE, FASTER PROCESSORS. <*&
2716 016106 005000 CLR R0 ;CLEAR THE WAIT FOR CLOCK INT COUNTER.
2717 016110 012767 000001 164162 MOV #1,TIMER1 ;SET UP COUNT OF 1 TO SYNCH WITH LTC.
2718 016116 005767 164156 4$: TST TIMER1 ;CHECK FOR COUNTER HAVING GONE TO ZERO.
2719 016122 001410 BEQ 6$ ;JUMP OUT OF LOOP IF LTC HAS INTERRUPTED.
2720 016124 005200 INC R0 ;COUNT THIS ITERATION OF THE INNER LOOP.
2721 016126 001373 BNE 4$ ;LOOP IF COUNTER HAS NOT TURNED OVER.
2722 016130 005305 DEC R5 ;DECREMENT THE INNER LOOP COUNTER.
2723 016132 003371 BGT 4$ ;LOOP IF OUTER LOOP COUNT NOT UP.
2724 ;*
2725 ;* IF WE GOT NO LTC INTERRUPT, INDICATE THAT THERE IS NO LTC AVAILABLE.
2726 ;* LTC MUST BE FLAKEY, OR NOT REALLY AN LTC AT ALL.
2727 ;* - - - - -
2728 016134 005067 164136 CLR CLKHRZ ;CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.
2729 016140 000241 CLC ;INDICATE FAILURE FOR RETURN.
2730 016142 000461 BR 60$ ;BYPASS THE FOLLOWING CALIBRATION PROCEDURES.
2731 ;*
2732 ;* WE ARE NOW SYNCHRONIZED WITH THE LTC.
2733 ;* SET UP FOR THE CALIBRATION LOOP.
2734 ;* - - - - -
2735 016144 012704 002300 6$: MOV #TIMER1,R4 ;WILL TEST TIMER1 IN THE LOOP BELOW.

```

```

2736 016150 005001          CLR      R1          ;CLEAR THE OUTER LOOP COUNTER.
2737 016152 005002          CLR      R2          ;INDICATE TO CHECK ALL BITS OF TIMER1.
2738 016154 005003          CLR      R3          ;INDICATE TO CHECK FOR TIMER1 CLEAR.
2739 016156 012714 000001  MOV      #1,(R4)     ;LOAD TIMER1 WITH COUNT OF 1.
2740
2741 016162 016705 164124  8$:      MOV      MSLCNT,R5 ;LOAD MS LOOP COUNT.
2742 016166 011400 10$:      MOV      (R4),R0     ;GET THE TIMER1 VALUE.
2743 016170 010067 000120  MOV      R0,64$     ;SAVE WORD (LIKE IN THE REAL LOOP).
2744 016174 040200          BIC      R2,R0       ;LEAVE ALL THE BITS.
2745 016176 020003          CMP      R0,R3       ;COMPARE AGAINST ZERO.
2746 016200 000261          SEC          ;SET CARRY IN CASE OF SUCCESS.
2747 016202 001401          BEQ      12$        ;EXIT LOOP IF TIMER1 HAS CLEARED.
2748 016204 005305          DEC      R5         ;COUNT DOWN THE INSIDE MS LOOP COUNT.
2749 016206 001367          BNE     10$        ;LOOP IF MS NOT UP.
2750 016210 005301          DEC      R1         ;DECREMENT THE MS TIME COUNT.
2751 016212 001363          BNE     8$         ;KEEP LOOPING.
2752 016214 004767 003604  JSR      PC,OOPS     ;WE OVERFLOWED, SOMETHING IS WRONG, ABORT.
2753
2754          ;*
2755          ; WE HAVE NOW HAVE LOOP COUNT INFORMATION FOR ONE CLOCK TICK.
2756          ; WE HAVE NEGATIVE OF NUMBER OF OUTER LOOPS IN R1, EACH IS MSLCNT INNER LOOPS.
2757          ; WE HAVE THE PORTION OF THE LAST OUTER LOOP NOT EXECUTED, IN R5.
2758          ; NOW WE CALCULATE THE TOTAL NUMBER OF INNER LOOPS EXECUTED.
2759 016220 005401          ;-
2760 016222 016702 164064  12$:     NEG      R1          ;GET NUMBER OF OUTER LOOPS.
2761 016226 010203          MOV      MSLCNT,R2  ;GET THE NUMBER OF INNER LOOPS PER OUTER LOOP.
2762 016230 160502          MOV      R2,R3      ;COPY NUMBER OF LOOPS FOR MULTIPLY.
2763 016232 010204          SUB      R5,R2      ;CALC # OF INNER LOOPS DONE IN LAST OUTER LOOP
2764 016234 005005          MOV      R2,R4      ; AND ADD TO ACCUMULATOR LSWORD.
2765 016236 005301          CLR      R5         ;CLEAR ACCUMULATOR MSWORD.
2766 016240 100403          14$:     DEC      R1          ;CHECK R1 FOR 0 CONDITION
2767 016242 060304          BMI     16$        ; SKIP MULTIPLICATION IF ZERO
2768 016244 005505          ADD      R3,R4      ;MULTIPLY NUMBER OF INNER
2769 016246 000773          ADC      R5         ; LOOPS PER OUTER LOOP BY
2770          BR      14$   ;NUMBER OF OUTER LOOPS PERFORMED.
2771          ;*
2772          ; DIVIDE THE TOTAL NUMBER OF INNER LOOPS BY THE NUMBER OF MS PER LTC TICK.
2773 016250 016701 164034  16$:     MOV      MSTICK,R1   ;# OF MS PER LTC TICK IS DIVISOR.
2774 016254 010403          MOV      R4,R3      ;LSWORD OF LOOP COUNT IS LSWORD OF DIVIDEND.
2775 016256 010502          MOV      R5,R2      ;MSWORD OF LOOP COUNT IS MSWORD OF DIVIDEND.
2776 016260 004767 010076  JSR      PC,UNSDIV   ;DIVIDE NUMBER OF LOOPS BY MS PER LTC TICK.
2777 016264 103402          BCS     18$        ;BYPASS OOPS IF WE'RE OK.
2778 016266 004767 003532  JSR      PC,OOPS     ;CLOCK ROUTINES ARE NOT LONG ENOUGH, OR BUG.
2779 016272 010167 164014  18$:     MOV      R1,MSLCNT  ;SET NEW VALUE FOR MS LOOP COUNT.
2780 016276 005167 000010  COM      62$        ;SET THE 2ND ITERATION FLAGS IF 1ST ITERATION.
2781 016302 001277          BNE     2$         ;BRANCH IF ONLY ONE ITERATION DONE.
2782 016304 000261          SEC          ;SET THE SUCCESS FLAG FOR EXIT.
2783
2784 016306          60$:     PASS          ;RESTORE GPRS,
2785 016310 000207          RTS      PC        JSR      PC,@(SP) ;RETURN TO PREG05 SUBRT.
2786          ; CARRY - SUCCESS FLAG. SET IF SUCCESS.
2787 016312 000000          62$:     .WORD    0          ;2ND CALIBRATION ITERATION FLAGS.
2788 016314 000000          64$:     .WORD    0          ;DUMMY WORD FOR STORAGE OF THE READ WORD.

```



```

2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819 016316
      016316 004567 167002
2820 016322 016302 003402
2821 016326 005724
2822 016330 012400
2823 016332 100026
2824 016334 040500
2825 016336 112201
2826 016340 040501
2827 016342 120100
2828 016344 001021
2829 016346 016300 003542
2830 016352 005200
2831 016354 016301 005234
2832 016360 020061 003442
2833 016364 001407
2834 016366 011400
2835 016370 100005
2836 016372 040500
2837 016374 111201
2838 016376 040501
2839 016400 020001
2840 016402 001002
2841
2842
2843
2844
2845

```

```

.SBTTL GLOBAL SUBROUTINE - CHKEXT -
;*****
; - CHECK FOR EXTRA CHARACTER ROUTINE -
; THIS SUBROUTINE CHECKS FOR THE CONDITION WHICH INDICATES THAT AN EXTRA
; CHARACTER HAS BEEN RECEIVED DURING THE RECEPTION OF A DATA PATTERN.
; IF THIS ROUTINE DETERMINES THAT IT IS LIKELY THAT AN EXTRA CHARACTER
; HAS BEEN RECEIVED IT INDICATES THIS IN THE STATUS INFORMATION RETURNED
; TO THE CALLING ROUTINE.
;
; INPUTS:      R3 - RX LINE NUMBER MULTIPLIED BY 2 (OFFSET INTO WORD TABLES).
;              R4 - BASE ADDRESS OF RESYNC QUE CONTAINING RX CHARS.
;              R5 - MASK OF "INACTIVE" (NON-DATA) BITS OF RX AND TX CHARS.
;              CHCNTB - BASE OF NUMBER OF CHARS TO TX ON EACH LINE TABLE.
;              RXCNTB - BASE OF THE RX CHARACTER COUNTERS TABLE.
;              RXPTRB - BASE OF THE RX CHARACTER POINTERS TABLE.
;              TXRXLB - BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
;
; OUTPUTS:     CARRY - SET IF EXTRA CHARACTER CONDITION IS VERIFIED.
;
; CALLING SEQUENCE: JSR PC,CHKEXT
;
; COMMENTS: THE FOLLOWING SYMBOLS ARE USED IN LINE COMMENTS:
;           CHR0 - CHARACTER AT BOTTOM OF RESYNC QUE (FIRST RECEIVED).
;           CHR1, CHR2 - 2 CHARACTERS RECEIVED AFTER CHR0.
;           EXPO - CHARACTER EXPECTED TO BE RECEIVED NEXT.
;           EXP1, EXP2 - CHARACTER EXPECTED TO BE RECEIVED AFTER EXPO, ETC.
;
; SUBORDINATE ROUTINES CALLED: NONE.
;*****
CHKEXT:: SAVE
; SAVE CONTENTS OF GPRS R0 THRU R5.
      JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
; GET THE RX DATA POINTER.
      MOV RXPTRB(R3),R2
      TST (R4)+
; INCREMENT R4 BY 2 TO POINT TO CHR1.
      MOV (R4)+,R0
; GET CHR1 FROM THE QUE, DATA.VALID INTO N FLAG.
      BPL 52$
; EXIT WITH "FAILURE" IF CHR1 NOT VALID.
      BIC R5,R0
; REMOVE INACTIVE BITS FROM CHR1 VALUE.
      MOVB (R2)+,R1
; GET EXPO FROM THE DATA PATTERN.
      BIC R5,R1
; REMOVE INACTIVE BITS FROM EXPO VALUE.
      CMPB R1,R0
; COMPARE CHR1 AND EXPO.
      BNE 52$
; EXIT WITH "FAILURE" IF CHR1 <> EXPO.
      MOV RXCNTB(R3),R0
; COMPARE THE PRESENT RX CHARACTER COUNT PLUS 1
      INC R0
; WITH THE EXPECTED NUMBER OF CHARS TO RX ON
; LINE (NUMBER TRANSMITTED AND LOOPED BACK) TO
      MOV TXRXLB(R3),R1
      CMP R0,CHCNTB(R1)
; DETERMINE IF CHR1 IS LAST EXPECTED CHAR.
      BEQ 50$
; EXIT WITH "SUCCESS" IF CHR1 IS LAST CHAR.
      MOV (R4),R0
; GET CHR2 FROM THE QUE, DATA.VALID INTO N FLAG.
      BPL 50$
; EXIT WITH "SUCCESS" IF CHR1 WAS LAST IN QUE.
      BIC R5,R0
; REMOVE INACTIVE BITS FROM CHR2 VALUE.
      MOVB (R2),R1
; GET THE EXP1 VALUE.
      BIC R5,R1
; REMOVE INACTIVE BITS FROM EXP1 VALUE.
      CMP R0,R1
; COMPARE CHR2 AND EXP1.
      BNE 52$
; EXIT WITH "FAILURE" IF CHR2 <> EXP1.
;
; IT IS LIKELY THAT WE RECEIVED AN EXTRA CHARACTER WITHIN THE DATA PATTERN.
; INDICATE "SUCCESS" AND EXIT.
;
;*****

```

```

2846 016404 000261          50$:   SEC          ;SET THE SUCCESS FLAG.
2847 016406 000401          BR      60$          ;EXIT THE ROUTINE.
2848
2849
2850
2851          ;*
2852          ; WE DIDN'T RECEIVE A SINGLE EXTRA CHARACTER AT THIS POINT IN THE DATA PATTERN.
2853          ; INDICATE "FAILURE" AND EXIT.
2854          ;-
2855 016410 000241          52$:   CLC          ;CLEAR THE SUCCESS FLAG.
2856 016412 004736          60$:   PASS
                RTS      PC      JSR          ;RESTORE GPRS.
                ;PC,@(SP)+          ;RETURN TO PREG05 SUBRT.
                ;CARRY - SLF IF SUCCESS (EXTRA CHAR RXED).

```

```

2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888 016416
      016416 004567 166702
2889 016422 016301 003542
2890 016426 005201
2891 016430 016300 005234
2892 016434 016002 003442
2893 016440 020102
2894 016442 001423
2895 016444 005201
2896 016446 160201
2897 016450 016302 003402
2898 016454 005202
2899 016456 112200
2900 016460 162400
2901 016462 040500
2902
2903 016464 001012
2904 016466 005701
2905 016470 001406
2906 016472 011401
2907 016474 100004
2908 016476 111200
2909 016500 160001
2910 016502 040501
2911
2912 016504 001002
2913

```

```

.SBTTL GLOBAL SUBROUTINE - CHKLOS -
;*****
;* - CHECK FOR LOST CHARACTER ROUTINE -
;* THIS SUBROUTINE CHECKS FOR THE CONDITION WHICH INDICATES THAT A CHAR
;* HAS BEEN "LOST" FROM THE LOOPED BACK DATA PATTERN DURING A TRANSMISSION
;* AND RECEPTION TEST. IF THIS ROUTINE DETERMINES THAT IT IS LIKELY THAT
;* A CHARACTER HAS BEEN LOST, IT INDICATES THIS IN THE STATUS INFORMATION
;* RETURNED TO THE CALLING ROUTINE.
;*
;* INPUTS: R3 - RX LINE NUMBER MULTIPLIED BY 2 (OFFSET INTO WORD TABLES).
;* R4 - BASE ADDRESS OF RESYNC QUE CONTAINING RX CHARS.
;* R5 - MASK OF "INACTIVE" (NON-DATA) BITS OF RX AND TX CHARS WITH
;* ALL SET BITS IN A SINGLE, LEFT JUSTIFIED GROUP.
;* CHCNTB - BASE OF NUMBER OF CHARS TO TX ON EACH LINE TABLE.
;* RXCNTB - BASE OF THE RX CHARACTER COUNTERS TABLE.
;* RXPTRB - BASE OF THE RX CHARACTER POINTERS TABLE.
;* TXRXLB - BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
;*
;* OUTPUTS: CARRY - SET IF LOST CHARACTER CONDITION IS VERIFIED.
;*
;* CALLING SEQUENCE: JSR PC,CHKLOS
;*
;* COMMENTS: THE FOLLOWING SYMBOLS ARE USED IN LINE COMMENTS:
;* CHR0 - CHARACTER AT BOTTOM OF RESYNC QUE (FIRST RECEIVED).
;* CHR1, CHR2 - 2 CHARACTERS RECEIVED AFTER CHR0.
;* EXPO - CHARACTER EXPECTED TO BE RECEIVED NEXT.
;* EXP1, EXP2 - CHARACTER EXPECTED TO BE RECEIVED AFTER EXPO, ETC.
;*
;* SUBORDINATE ROUTINES CALLED: NONE.
;*****
CHKLOS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
      JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
      MOV RXCNTB(R3),R1 ;COMPARE THE PRESENT RX CHARACTER COUNT PLUS 1
      INC R1 ; WITH THE EXPECTED NUMBER OF CHARS TO RX ON
      MOV TXRXLB(R3),R0 ; LINE (NUMBER TXED AND LOOPED BACK) TO
      MOV CHCNTB(R0),R2 ; DETERMINE IF THE POSSIBLE LOST CHAR
      CMP R1,R2 ; WOULD BE THE LAST EXPECTED RX CHAR.
      BEQ 52$ ;EXIT WITH "FAILURE" IF LOST CHR WOULD BE LAST.
      INC R1 ;DETERMINE (AS ABOVE) IF CHR0 WOULD BE THE LAST
      SUB R2,R1 ; RX CHAR AND SAVE RESULT FOR LATER.
      MOV RXPTRB(R3),R2 ;GET THE RX DATA POINTER.
      INC R2 ;CALCULATE POINTER TO EXP1 LOCATION.
      MOVB (R2),R0 ;GET EXP1 VALUE FROM DATA PATTERN.
      SUB (R4),R0 ;COMPARE CHR0 AND EXP1 VALUES.
      BIC R5,R0 ;REMOVE INACTIVE BITS FROM RESULT. (NO ACTIVE
      ; BITS ALLOWED TO LEFT OF ANY INACTIVE BITS.)
      BNE 52$ ;EXIT WITH "FAILURE" IF CHR0 <> EXP1.
      TST R1 ;CHECK CHR0 TEST RESULT SAVED ABOVE.
      BEQ 50$ ;EXIT WITH "SUCCESS" IF CHR0 IS LAST CHAR.
      MOV (R4),R1 ;GET CHR1 FROM THE QUE. DATA.VALID INTO N FLAG.
      BPL 50$ ;EXIT WITH "SUCCESS" IF CHR0 WAS LAST QUE CHAR.
      MOVB (R2),R0 ;GET THE EXP2 VALUE FROM THE DATA PATTERN.
      SUB R0,R1 ;COMPARE THE EXP2 AND THE CHR1 VALUES.
      BIC R5,R1 ;REMOVE INACTIVE BITS FROM RESULT OF COMPARE.
      ; (NO ACTIVE BITS LEFT OF INACTIVE BITS.)
      BNE 52$ ;EXIT WITH "FAILURE" IF CHR1 <> EXP2.

```

```

2914
2915      ;+
2916      ; IT IS LIKELY THAT WE LOST A CHARACTER FROM THE DATA PATTERN.
2917      ; INDICATE "SUCCESS" AND EXIT.
2918      ;-
2918 016506 000261 50$:      SEC          ;SET THE SUCCESS FLAG.
2919 016510 000401          BR      60$      ;EXIT THE ROUTINE.
2920
2921
2922      ;+
2923      ; WE DIDN'T LOSE A SINGLE EXTRA CHARACTER AT THIS POINT IN THE DATA PATTERN.
2924      ; INDICATE "FAILURE" AND EXIT.
2925      ;-
2925 016512 000241 52$:      CLC          ;CLEAR THE SUCCESS FLAG.
2926
2927 016514          60$:      PASS         ;RESTORE GPRS.
2927      016514 004736          RTS      PC      JSR      PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
2928 016516 000207          ;CARRY - SET IF SUCCESS (LOST CHAR LIKELY).

```

2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965 016520
016520 004567 166600
2966
2967
2968
2969 016524 036367 002364 163752
2970 016532 001407
2971
2972
2973
2974
2975
2976
2977 016534 012701 011306
2978 016540 011402
2979 016542 040502
2980 016544 052704 100000
2981 016550 000452
2982
2983
2984
2985 016552 016302 003402

```

.SBTTL GLOBAL SUBROUTINE - CKCHR -
; * * * * *
; * - CHECK CHARACTER FOR ERRORS ROUTINE -
; * THIS SUBROUTINE CHECKS THE CHARACTER AT THE BOTTOM OF THE RESYNC QUEUE
; * TO DETERMINE IF IT IS CORRECT. POINTERS AND COUNTERS WHICH ARE RELATED
; * TO THE RECEPTION OF THE CHARACTER ARE UPDATED. IF THE CHARACTER IS
; * INCORRECT, AN ANALYSIS OF THE ERROR IS DONE AND PARAMETERS ARE SET UP
; * FOR THE REPORTING OF THE CORRECT ERROR.
; *
; * INPUTS: R3 - LINE OFFSET FOR ACCESS OF WORD TABLES OF LINE VARIABLES.
; * R4 - BASE ADDRESS OF THE RESYNC QUEUE FOR THIS LINE.
; * R5 - MASK OF THE INACTIVES BITS IN A TX OR RX CHAR BYTE.
; * BITTBL - TABLE OF WORDS WITH BITS SET FOR USE IN FORMING MAPS.
; * DPRSQ - DATA PATTERN RESYNC QUE WITH VALID CHAR AT BOTTOM.
; * EXCNTB - BASE OF THE EXTRA CHARACTER COUNTERS TABLE.
; * RXDNF - RECEIVE DONE FLAGS.
; * RXPTRB - BASE OF THE RX CHARACTER POINTERS TABLE.
; * ERROR MESSAGE LABELS - EM9007,EM9008,EM9027,EM9028
; *
; * OUTPUTS: R1 - CONTAINS THE ADDRESS OF THE ERROR MESSAGE TO BE REPORTED.
; * R2 - CONTAINS THE ACTUAL RECEIVED DATA.
; * R4 - CONTAINS THE EXPECTED DATA.
; * CARRY - "SUCCESS" FLAG (SET IF NO ERROR IS FOUND).
; * FOLLOWING VARIABLES UPDATED FOR LINE ON WHICH CHAR WAS RECEIVED:
; * EXCNT - COUNT OF THE NUMBER OF EXTRA CHARS RECEIVED ON LINE.
; * RXCNT - COUNT OF THE NUMBER OF CHARACTERS RECEIVED ON LINE.
; * RXPTR - UPDATED TO POINT TO THE NEXT EXPECTED CHAR ON LINE.
; * ERRBLK - CONTENTS DESTROYED.
; *
; * CALLING SEQUENCE: JSR PC,CKCHR
; *
; * COMMENTS:
; *
; * SUBORDINATE ROUTINES CALLED: CHKEXT,CHKLOS,UPDCHR.
; * - * * * *
CKCHR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;
; CHECK FOR THE RX OF A CHAR AFTER RX SHOULD BE COMPLETE ON THIS LINE.
;
; BIT BITTBL(R3),RXDNF ;TEST THE RX DONE FLAG FOR THIS LINE.
; BEQ 2$ ;SKIP ERROR REPORT IF RX NOT COMPLETE ON LINE.
;
; WE HAVE RECEIVED AN EXTRA CHARACTER ON THIS LINE.
; SET UP FOR ERROR REPORT AND EXIT TO REPORT THE ERROR.
; COUNT THE EXTRA CHARACTER.
; EXIT TO REPORT "UNEXPECTED CHAR RECEIVED AFTER RX COMPLETE ON LINE NN"
;
; MOV #EM9007,R1 ;SELECT "EXTRA CHAR ON LINE" ERROR MESSAGE.
; MOV (R4),R2 ;GET THE ACTUAL DATA FOR ERROR REPORT.
; BIC R5,R2 ;REMOVE THE INACTIVE BITS.
; BIS #BIT15,R4 ;INDICATE "NONE" EXPECTED DATA FOR ERROR RPT.
; BR 12$ ;GO COUNT EXTRA CHAR AND EXIT WITH "FAILURE".
;
; GET THE POINTER TO THE NEXT EXPECTED RECEIVE DATA CHARACTER.
;
; 2$: MOV RXPTRB(R3),R2
    
```

```

2986
2987
2988
2989 016556 011400
2990 016560 040500
2991 016562 111201
2992 016564 040501
2993 016566 120001
2994 016570 001003
2995 016572 004767 007720
2996 016576 000446
2997
2998
2999
3000
3001 016600 004767 177512
3002 016604 103010
3003
3004
3005
3006
3007
3008 016606 012701 012121
3009 016612 111200
3010 016614 040500
3011 016616 011402
3012 016620 040502
3013 016622 010004
3014 016624 000424
3015
3016
3017
3018
3019
3020 016626 004767 177564
3021 016632 103012
3022
3023
3024
3025
3026
3027
3028 016634 012701 012201
3029 016640 111200
3030 016642 040500
3031 016644 011402
3032 016646 040502
3033 016650 010004
3034 016652 004767 007640
3035 016656 000404
3036
3037
3038
3039
3040 016660 010002
3041 016662 010104
3042 016664 012701 011371

```

```

;+
; COMPARE THE ACTUAL DATA WITH THE EXPECTED DATA.
;-
      MOV      (R4),R0      ;GET THE ACTUAL DATA.
      BIC      R5,R0      ;REMOVE THE INACTIVE BITS.
      MOVB     (R2),R1      ;GET THE EXPECTED DATA.
      BIC      R5,R1      ;REMOVE THE INACTIVE BITS.
      CMPB     R0,R1      ;COMPARE ACTUAL AND EXPECTED.
      BNE      4$         ;CHECK FURTHER IF DATA MISCOMPARE.
      JSR      PC,UPDCHR   ;UPDATE PTRS AND COUNTERS FOR THE CHAR.
      BR       50$        ;EXIT WITH "SUCCESS", NO ERROR FOUND.

;+
; ACTUAL AND EXPECTED DATA MISCOMPARE.
; DETERMINE IF IT'S LIKELY WE RECEIVED AN EXTRA CHAR WITHIN THE DATA PATTERN.
;-
4$:   JSR      PC,CHKEXT   ;CHECK FOR EXTRA CHAR RX'ED IN PATTERN.
      BCC      6$         ;GO CHECK FOR LOST CHAR IF NO EXTRA CHAR.

;+
; IT IS LIKELY THAT WE RECEIVED AN EXTRA CHARACTER WITHIN THE DATA PATTERN.
; COUNT THE CHAR AS AN EXTRA CHAR, DON'T COUNT AS A STANDARD CHAR.
; REPORT "EXTRA CHAR RECEIVED WITHIN DATA PATTERN ON LINE NN"
;-
      MOV      #EM9027,R1  ;SELECT "EXTRA CHAR ON LINE" ERROR MSG.
      MOVB     (R2),R0      ;GET THE EXPECTED RECEIVE DATA.
      BIC      R5,R0      ;REMOVE THE INACTIVE BITS FROM EXPECTED DATA.
      MOV      (R4),R2      ;GET THE ACTUAL RECEIVE DATA.
      BIC      R5,R2      ;REMOVE THE INACTIVE BITS FROM ACTUAL DATA.
      MOV      R0,R4      ;PASS EXPECTED DATA TO ERROR REPORT ROUTINE.
      BR       12$        ;GO COUNT EXTRA CHAR AND EXIT WITH "FAILURE".

;+
; ACTUAL AND EXPECTED DATA MISCOMPARE.
; NOT LIKELY THAT WE RECEIVED AN EXTRA CHARACTER WITHIN THE DATA PATTERN.
; DETERMINE IF IT'S LIKELY WE LOST A CHARACTER FROM THE DATA PATTERN.
;-
6$:   JSR      PC,CHKLOS   ;CHECK FOR A LOST CHAR CONDITION.
      BCC      8$         ;GO REPORT BAD RX DATA IF NOT LOST CHAR.

;+
; IT IS LIKELY THAT WE LOST A CHARACTER FROM THE DATA PATTERN.
; COUNT THE CHAR IN THE RX CHAR COUNT AS IF IT HAD BEEN RECEIVED.
; ALSO, COUNT CHRO AS A VALID CHAR, BECAUSE WE HAVE VERIFIED IT ABOVE.
; REPORT "SINGLE CHAR MISSING FROM RECEIVED DATA ON LINE NN"
;-
      MOV      #EM9028,R1  ;SELECT "LOST CHAR ON LINE" ERROR MSG. +++++
      MOVB     (R2),R0      ;GET THE EXPECTED RECEIVE DATA.
      BIC      R5,R0      ;REMOVE THE INACTIVE BITS FROM EXPECTED DATA.
      MOV      (R4),R2      ;GET THE ACTUAL RECEIVE DATA.
      BIC      R5,R2      ;REMOVE THE INACTIVE BITS FROM ACTUAL DATA.
      MOV      R0,R4      ;PASS EXPECTED DATA TO ERROR REPORT ROUTINE.
      JSR      PC,UPDCHR   ;UPDATE PTRS AND COUNTERS FOR THE CHAR.
      BR       10$        ;GO EXIT WITH "FAILURE".

;+
; DID NOT LOSE OR GAIN A SINGLE CHARACTER FROM/TO THE DATA PATTERN.
; REPORT "RECEIVED CHAR MISCOMPARE AGAINST TX DATA ON LINE NN"
;-
8$:   MOV      R0,R2      ;PASS ACTUAL DATUM TO ERROR REPORT ROUTINE.
      MOV      R1,R4      ;PASS EXPECTED DATUM TO ERROR REPORT ROUTINE.
      MOV      #EM9008,R1  ;SELECT THE "DATA MISCOMPARE" MESSAGE.

```

```

3043      ;
3044      ; UPDATE THE CHARACTER COUNTER AND RX DATA PATTERN POINTER FOR THIS LINE.
3045      ;
3046 016670 004767 007622 101: JSR PC,UPDCHR ;UPDATE RX PTR AND COUNTER FOR THIS LINE.
3047 016674 000405      BR 141 ;GO EXIT WITH "FAILURE".
3048      ;
3049      ; COUNT THE CHARACTER AS AN EXTRA CHARACTER.
3050      ;
3051 016676 005263 003242 121: INC EXCNTB(R3) ;INCREMENT THE EXTRA CHAR COUNT FOR THIS LINE.
3052 016702 001002      BNE 141 ;EXIT WITH FAILURE IF NO OVERFLOW.
3053 016704 005363 003242      DEC EXCNTB(R3) ;DECREMENT BACK TO -1 (MAX VALUE) IF OVERFLOW.
3054      ;
3055      ; INDICATE "FAILURE" AND EXIT.
3056      ;
3057 016710 000241 141: CLC ;CLEAR THE "SUCCESS" FLAG.
3058 016712 000401      BR 601 ;EXIT THE ROUTINE.
3059      ;
3060      ;
3061      ; NO ERROR WAS FOUND.
3062      ; SET "SUCCESS" FLAG AND EXIT.
3063      ;
3064 016714 000261 501: SEC ;SET THE "SUCCESS" FLAG.
3065      ;
3066 016716 010166 000004 601: PASS R1,R2,R4 ;RESTORE GPRS, EXCEPT
      ; R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
      ; R2,R2SLOT(SP) ;PUT R2 IN STACK SLOT.
      ; R4,R4SLOT(SP) ;PUT R4 IN STACK SLOT.
      ; PC,0(SP) ;RETURN TO PREGO5 SUBRT.
3067      ;R1 - CONTAINS THE ADDRESS OF THE ERROR REPORT.
3068      ;R2 - CONTAINS THE ACTUAL DATA RECEIVED.
3069      ;R4 - CONTAINS THE EXPECTED DATA.
3070 016734 000207 RTS PC

```

C/

3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126

016736
016736 004567 166362
016742 016704 166350
016746 004767 006706

016752 016701 163270
016756 026767 163520 163206
016764 001402
016766 062701 000062
016772 052701 170000
016776 016702 163200
017002 004767 010004
017006 103054

017010 005367 163462
017014 001014
017016 010467 166274
017022 012701 012001
017026 012767 014032 166266

017034
017034 104460

```
.SBTTL GLOBAL SUBROUTINE - CKFRPR -
*****
- CHECK FRAMING AND PARITY ERROR REPORTING -
THIS SUBROUTINE IS USED IN THE FRAMING ERROR AND PARITY ERROR TESTS.
IT READS THE CHARACTERS FROM THE DUT RECEIVER CHARACTER FIFO,
AND CHECKS FOR THE CORRECT COMBINATION OF PARITY AND FRAMING
ERROR BITS IN THE MSB. IF CHARACTERS STOP APPEARING IN THE FIFO WITH
DATA.VALID SET OR IF MORE THAN THE ALLOWABLE NUMBER OF CHARACTERS
HAS BEEN READ FROM THE DUT THIS ROUTINE EXITS WITH AN RX COMPLETE
INDICATION. EACH READ CHAR IS ANALYSED AND ANY NECESSARY ERRORS ARE
REPORTED.
INPUTS: R5 - TEST FLAG, BIT15 SET = FRAMING ERR, CLEAR = PARITY ERR.
ERRNBR - SET TO ERROR NUMBER OF FIRST ERROR IN THIS ROUTINE.
OSTEND - ADDRESS OF THE END OF THE OUTPUT STORAGE FIFO BUFFER.
OSTPTR - POINTER TO THE NEXT BYTE TO READ FROM OSTORE.
OUTPUTS: RXCNTB - RECEIVE CHARACTER COUNT UPDATED FOR EACH LINE.
RXPNTB - RECEIVE CHARACTER PIONTER IS UPDATED FOR EACH LINE.
CALLING SEQUENCE: JSR PC,CKFRPR
COMMENTS: THIS ROUTINE REPORTS ERRORS WITH NUMBERS INITIAL ERRNBR
THRU INITIAL ERRNBR + 4.
ERRNBR IS RESTORED BEFORE THIS ROUTINE RETURNS.
SUBORDINATE ROUTINES CALLED: PRFRME,PRPARE,WAIBIS.
*****
CKFRPR:: SAVE
JSR R5,PREG05 ;SAVE CONTENTS OF GPRS R0 THRU R5.
MOV ERRNBR,R4 ;CALL REGISTER SAVE SUBRT.
JSR PC,TXIE1 ;PRESERVE THE INITIAL ERROR NUMBER.
;ENABLE TX INTERRUPTS.
; WAIT FOR A CHARACTER TO APPEAR IN THE FIFO.
; IF NO CHARACTER APPEARS WITHIN TIME-OUT PERIOD: EXIT ROUTINE, WE'RE DONE.
MOV RXTOUT,R1 ;GET MINIMUM TIME OUT VALUE.
CMP TXDONF,ACTLNS ;CHECK FOR TRANSMISSION DONE ON ACTIVE LINES.
BEQ 4$ ;SKIP ADDING 50 MS DELAY IF TX DONE ALL LINES.
ADD #50.,R1 ;ADD 50 MILLI SEC TO DELAY IF NOT LAST CHAR.
BIS #170000,R1 ;INDICATE TO TEST DATA.VALID BIT.
MOV RBUFA,R2 ;INDICATE TO CHECK DUT RECEIVE BUFFER (FIFO).
JSR PC,WAIBIS ;WAIT FOR RECEIVED CHAR OR TIME-OUT.
BCC 60$ ;EXIT ROUTINE IF TIME-OUT, WE'RE DONE.
DEC CHRTOT ;DECREMENT THE TOTAL CHAR COUNTER.
BNE 6$ ;SKIP ERROR IF NOT TOO MANY CHARS RECEIVED.
MOV R4,ERRNBR ;SET ERROR NUMBER TO INITIAL ERRNBR.
MOV #EM9025,R1 ;SELECT THE ERROR MESSAGE TO BE REPORTED.
MOV #ER0503,ERRBLK ;SELECT THE ERROR REPORT ROUTINE.
; REPORT ERROR AT INITIAL ERRNBR.
; "MORE THAN TWICE THE EXPECTED NUMBER OF CHARACTERS RECEIVED"
ERROR 1 >>>> ERROR <<<<<
TRAP C$ERRRUR
```


D7

```

3127 017036 012767 000001 163160      MOV    #1,FERROR      ;INDICATE THAT AN ERROR HAS BEEN FOUND.
3128
3129 017044 000435                    BR     60$            ;EXIT THIS ROUTINE WE HAVE GIVEN UP.
3130
3131
3132
3133
3134
3135 017046 010203                    6$:   MOV    R2,R3      ;COPY THE READ CHARACTER.
3136 017050 000303                    SWAB   R3             ;GET THE LINE NUMBER IN THE LSB.
3137 017052 042703 177760            BIC    #177760,R3    ;CLEAR THE UNWANTED BITS.
3138 017056 006303                    ASL    R3             ;SHIFT LEFT TO FORM OFFSET INTO TABLES.
3139
3140
3141
3142 017060 010505                    ; PROCESS THE READ CHARACTERS AS DICTATED BY THE TEST FLAG.
3143 017062 100012                    ;
3144
3145 017064 004767 003162            MOV    R5,R5          ;DETERMIN WHICH TEST CALLED THIS ROUTINE.
3146 017070 005767 163130            BPL    8$            ;BRANCH TO PROCESS CHARACTER IN PARITY TEST.
3147 017074 001416                    JSR    PC,PRFRME     ;PROCESS FRAMING ERRORS RECEIVED.
3148
3149 017076 032767 000100 163056    TST    FERROR        ;HAS AN ERROR BEEN DETECTED ?
3150 017104 001012                    BEQ    10$           ;NO, THEN SKIP PROCESSING CHARACTERS FOR PARITY
3151 017106 000414                    BIT    #BIT06,OPTION ;TEST.
3152
3153 017110 004767 003242            BNE   10$           ;HAS EXTENDED ERROR REPORTING BEEN ENABLED ?
3154 017114 005767 163104            BR    60$           ;BRANCH IF IT HAS,
3155 017120 001404                    ;OTHERWISE EXIT.
3156 017122 032767 000100 163032    JSR    PC,PRPARE     ;PROCESS PARITY ERRORS RECEIVED.
3157 017130 001403                    TST    FERROR        ;HAS AN ERROR BEEN DETECTED ?
3158
3159 017132 004767 007360            BEQ    10$           ;NO, THEN BRANCH TO UPDATE POINTERS.
3160 017136 000707                    BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN ENABLED ?
3161
3162 017140 010467 166152            BEQ    60$           ;EXIT IF IT HASN'T.
3163 017144 004736                    JSR    PC,UPDCHR     ;UPDATE POINTERS AND COUNTERS FOR THIS LINE.
3164 017146 000207                    BR    2$            ;LOOP TO READ NEXT CHAR FROM FIFO.
                                MOV    R4,ERRNBR      ;RESTORE THE ERROR NUMBER TO ITS INITIAL VALUE.
                                PASS                    ;RESTORE GPRS.
                                RTS     PC             ;RETURN TO PREG05 SUBRT.

```

```

3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195 017150
      017150 004567 166150
3196
3197
3198
3199
3200 017154 010203
3201 017156 000303
3202 017160 042703 177760
3203 017164 006303
3204
3205
3206
3207 017166 005702
3208 017170 100021
3209
3210
3211
3212
3213 017172 016301 005234
3214 017176 036167 002364 162766
3215 017204 001013
3216
3217
3218
3219
3220
3221

```

```

.SBTTL GLOBAL SUBROUTINE - CKINAC -
; * *****
; * - CHECK FOR NEW CHARACTER ON INACTIVE LINE ROUTINE -
; * THIS SUBROUTINE CHECKS A CHARACTER TO DETERMINE IF THE CHARACTER
; * WAS RECEIVED ON AN ACTIVE LINE. IF THE CHARACTER WAS RECEIVED ON
; * AN INACTIVE LINE THIS ROUTINE RECORDS THE FACT THAT THE CHARACTER
; * WAS RECEIVED ON AN INACTIVE LINE, PREPARES AN ERROR MESSAGE FOR
; * THE CALLING ROUTINE, AND RETURNS A "FAILURE" STATUS.
; *
; * INPUTS: R2 - THE RX CHARACTER INCLUDING ERROR FLAGS AND LINE NUMBER.
; * ACTLNS - BIT MAP OF ACTIVE DUT LINES.
; * BITTBL - TABLE OF WORDS WITH BITS SET FOR FORMING BIT MAPS.
; * EM9006 - LABEL AT "RX ON INACTIVE LINE" ERROR MESSAGE.
; * EXCNTB - BASE OF THE EXTRA CHARACTER COUNTERS TABLE.
; * TXRXLB - BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
; *
; * OUTPUTS: CARRY - "SUCCESS" FLAG (SET IF NO ERROR FOUND).
; * R1 - IF ERROR FOUND, ADDRESS OF ERROR MESSAGE.
; * R3 - LINE NUMBER OFFSET OF PASSED IN CHARACTER.
; * R4 - IF ERROR FOUND, EXPECTED DATA INDICATION FOR ERROR RPT.
; * EXCNT - EXTRA CHARACTER COUNT FOR LINE (UPDATED IF ERROR).
; *
; * CALLING SEQUENCE: JSR PC,CKINAC
; *
; * COMMENTS:
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
; * - *****
CKINAC:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
          JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
; *
; * EXTRACT THE LINE NUMBER FROM THE PASSED IN CHARACTER AND USE THE LINE
; * NUMBER TO FORM AN OFFSET FOR ACCESSING TABLES OF LINE VARIABLES.
; *
; * MOV R2,R3 ;EXTRACT THE LINE NUMBER
; * SWAB R3 ; FROM THE CHARACTER WE
; * BIC #177760,R3 ; ARE COMPARING.
; * ASL R3 ;FORM OFFSET INTO WORD TABLE FROM LINE NUMBER.
; *
; * IF THE CHARACTER IN QUESTION IS NOT A VALID CHARACTER, EXIT WITH "SUCCESS".
; *
; * TST R2 ;CHECK DATA VALID BIT.
; * BPL 50$ ;EXIT WITH SUCCESS IF CHAR IS NOT VALID.
; *
; * IF THE TX LINE WHICH IS ASSOCIATED WITH THIS RX LINE IS AN ACTIVE LINE,
; * EXIT THE ROUTINE WITH "SUCCESS".
; *
; * MOV TXRXLB(R3),R1 ;GET THE TX LINE # OFFSET FOR THIS RX LINE.
; * BIT BITTBL(R1),ACTLNS ;DETERMINE IF TX LINE IS AN ACTIVE LINE.
; * BNE 50$ ;EXIT ROUTINE WITH SUCCESS IF LINE IS ACTIVE.
; *
; * THE CHARACTER IN QUESTION WAS RECEIVED ON AN INACTIVE LINE.
; * COUNT THIS CHARACTER AS AN EXTRA CHAR.
; * SET UP ERROR INFORMATION.
; * EXIT ROUTINE WITH "FAILURE" INDICATION.
; * -

```

F7

```

3222 017206 005263 003242          INC    EXCNTB(R3)      ;INCREMENT THE EXTRA CHAR COUNT FOR THIS LINE.
3223 017212 001002                BNE    2$             ;SKIP SETTING TO MAX VALUE IF NO OVERFLOW.
3224 017214 005363 003242          DEC    EXCNTB(R3)      ;DECREMENT BACK TO -1 (MAX VALUE) IF OVERFLOW.
3225 017220 012701 011233          2$:   MOV    #EM9006,R1 ;SET UP RX ON INACTIVE LINE MESSAGE.
3226 017224 012704 100000          MOV    #BIT15,R4      ;SET UP "NONE" EXPECTED DATA INDICATION.
3227 017230 000241                CLC                    ;CLEAR THE "SUCCESS" FLAG.
3228 017232 000401                BR     60$            ;GO REPORT RX CHAR ON INACTIVE LINE.
3229
3230
3231                                ;+
3232                                ; WE HAVE NOT FOUND A "CHAR ON INACTIVE LINE" ERROR SITUATION.
3233                                ; SET THE "SUCCESS" FLAG AND EXIT THE ROUTINE.
3234 017234 000261          50$:   SEC                    ;SET THE "SUCCESS" FLAG.
3235
3236 017236                60$:   PASS    R1,R3,R4      ;RESTORE GPRS, EXCEPT OUTPUT GPRS.
                                MOV    R1,R1SLOT(SP)      ;PUT R1 IN STACK SLOT.
                                MOV    R3,R3SLOT(SP)      ;PUT R3 IN STACK SLOT.
                                MOV    R4,R4SLOT(SP)      ;PUT R4 IN STACK SLOT.
                                JSR    PC,@(SP)+          ;RETURN TO PREG05 SUBRT.
3237 017254 000207          RTS    PC              ;CARRY - SUCCESS FLAG (SET IF NO ERROR).

```

```

3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262 017256
      017256 004567 166042
3263 017262 005067 162766
3264 017266 011011
3265 017270 005767 162760
3266 017274 000261
3267 017276 001401
3268 017300 000241
3269 017302
      017302 004736
3270 017304 000207

```

```

.SBTTL 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: R0 - SOURCE ADDRESS FOR MOVE.
;          R1 - DESTINATION ADDRESS FOR MOVE.
;          (R0) - SOURCE FOR THE MOVE.
;*
;* OUTPUTS: (R1) - WRITTEN TO THE CONTENTS OF (R0).
;          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
          CLR TP4FLG JSR R5,PREG05 ;SAVE CONTENTS OF GPRS R0 THRU R5.
          MOV (R0),(R1) ;CLEAR THE 004 TRAP FLAGS.
ADRPTR:: TST TP4FLG ;PERFORM THE MOVE IN QUESTION.
          SEC ;CHECK FOR OCCURENCE OF TRAP.
          BEQ 60$ ;INDICATE SUCCESS.
          CLC ;EXIT WITH SUCCESS IF TRAP DID NOT OCCUR.
          PASS ;INDICATE FAILURE.
60$: ;RESTORE GPRS.
          RTS PC JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

```

3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298 017306
017306 004567 166012
3299
3300 017312 005067 162736
3301 017316 111011
3302 017320 005767 162730
3303 017324 000261
3304 017326 001401
3305 017330 000241
3306 017332
017332 004736
3307 017334 000207

```

.SBTTL GOBAL SUBROUTINE - CKTRPB -
*****
; * - CHECK FOR TRAP -
; * 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 A TRAP DOES NOT OCCUR, THIS ROUTINE RETURNS A SUCCESS INDICATION.
; *
; * INPUTS: R0 - SOURCE ADDRESS FOR MOVE
; * R1 - DESTINATION ADDRESS FOR MOVE
; * (R0) - SOURCE FOR THE MOVE
; *
; * OUTPUTS: (R1) - WRITEN TO THE CONTENTS OF (R0)
; * CARRY FLAG - SET ON RETURN IF NO 004 TRAP DETECTED
; * TP4FLG - NONZERO IF TRAP OCCURED, CLEARED OTHERWISE.
; *
; * CALLING SEQUENCE: JSR PC,CKTRPB
; *
; * COMMENTS: IF THIS SUBROUTINE CAUSES A TRAP, EITHER THE ADDRESS
; * WHICH IS LABELED TRPAD2 WILL BE THE TRAP PC ADDRESS ON
; * THE STACK OR SOME OTHER ADDRESS WHICH WAS PLACED ON
; * THE STACK BY AN UNEXPECTED TRAP.
; * THIS ROUTINE PERFORMS A BYTE MOV .
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
*****
CKTRPB:: SAVE JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;
; CLR TP4FLG ;CLEAR THE 004 TRAP FLAGS
; MOVB (R0),(R1) ;PERFORM THE BYTE MOVE
TRPAD2:: TST TP4FLG ;CHECK FOR OCCURENCE OF TRAP
; SEC ;INDICATE SUCCESS
; BEQ 60$ ;EXIT WITH SUCCESS IF TRAP DID NOT OCCUR
; CLC ;INDICATE FAILURE
60$: PASS
;
; JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
; RTS PC ;RETURN

```

```

3309 .SBTTL GLOBAL SUBROUTINE - CLNRST -
3310 ;*****
3311 ;* - CLEAN RESET OF THE DEVICE UNDER TEST -
3312 ;* THIS SUBROUTINE IS USED TO RESET THE DUT TO A KNOWN STATE.
3313 ;* THE DUT'S SELF-TEST IS SKIPPED, AND THE FIFO IS PURGED OF ANY ERROR
3314 ;* CODES, ETC.
3315 ;* IF THE RESET DOES NOT SUCCESSFULLY COMPLETE, THEN THE CARRY BIT IS
3316 ;* PASSED BACK TO THE CALLING ROUTINE (CLEAR).
3317 ;*
3318 ;* INPUTS: CSRA - CONTAINS THE ADDRESS OF THE CSR
3319 ;* TXBFCA - CONTAINS ADDRESS OF DUT DMA BUFFER COUNT REGISTER.
3320 ;* ERRNBR - ERROR NUMBER FOR POSSIBLE ERROR REPORT.
3321 ;* ERRTBL - ERRTP, ERNBR, AND ERRMSG SET UP CORRECTLY.
3322 ;*
3323 ;* OUTPUTS: THE DUT PERFORMS ITS RESET FUNCTION INTO A KNOWN STATE.
3324 ;* CARRY - CLEAR INDICATES THE TEST IS TO BE ABORTED.
3325 ;* ERRBLK - VALUE MAY BE DESTROYED.
3326 ;* IESTAT - TX AND RX INTERRUPT FLAGS ARE CLEARED.
3327 ;* TX AND RX INTERRUPT ENABLE BITS IN THE DUT'S CSR ARE CLEARED.
3328 ;*
3329 ;* CALLING SEQUENCE: JSR PC, CLNRST
3330 ;*
3331 ;* COMMENTS: THIS SUBROUTINE CAN REPORT ERRORS WITH NUMBERS ERRNBR.
3332 ;* THIS ROUTINE DOES NOT DESTROY THE VALUE OF ERRNBR.
3333 ;*
3334 ;* SUBORDINATE ROUTINES CALLED: DELAY, MSLGET, PUFIFO, RESETT.
3335 ;*****
3336
3337 017336 CLNRST:: SAVE JSR ;SAVE CONTENTS OF GPRS R0 THRU R5.
017336 004567 165762 R5, PREG05 ;CALL REGISTER SAVE SUBRT.
3338 ;*
3339 ;* RESET THE DUT.
3340 ;* THIS ROUTINE REPORTS ERRORS WITH NUMBERS FROM ERRNBR THRU ERRNBR+2.
3341 ;*
3342 017342 004767 004510 JSR PC, RESETT ;RESET THE DUT TO A KNOWN STATE.
3343 017346 103002 BCC 60$ ;EXIT ROUTINE WITH ABORT TEST INDICATOR.
3344 ;*
3345 ;* PURGE THE FIFO OF ERROR CODES, SAVE ANY BMP CODES FOUND.
3346 ;*
3347 017350 004767 003274 JSR PC, PUFIFO ;PURGE THE FIFO.
3348 ;*
3349 017354 60$: ;EXIT THE TEST USING RESETT OR PUFIFO STATUS.
3350 017354 PASS ;RESTORE GPRS, PASS THE FOLLOWING INTACT:
017354 004736 JSR PC, @SP+ ;RETURN TO PREG05 SUBRT.
3351 ;* CARRY BIT; IF CLEAR, THEN ABORT THE TEST.
3352 017356 000207 RTS PC

```

3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376

017360
017360 004567 165740
017364 012701 000020
017370 005020
017372 005301
017374 001375
017376
017376 004736
017400 000207

```

.SBTTL GLOBAL SUBROUTINE - CLR16W -
; * *****
; * - CLEAR SIXTEEN WORDS ROUTINE -
; * THIS SUBROUTINE CLEARS 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 ;SAVE CONTENTS OF GPRS R0 THRU R5.
;CALL REGISTER SAVE SUBRT.
JSR R5,PREG05
;SET THE LOOP COUNTER TO 16.
MOV #16,,R1
;CLEAR A WORD OF MEMORY.
2$: CLR (R0)+
;COUNT THIS LOOP.
DEC R1
;LOOP IF NOT 16 WORD CLEARED.
BNE 2$
;RESTORE GPRS.
60$: PASS
JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC

```

3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397

```

.SBTTL GLOBAL SUBROUTINE - CONMAP -
; * *****
; * - CONVERT LINE BIT MAP.
; * THIS SUBROUTINE IS USED TO CONVERT A BIT MAP PASSED TO IT , INTO
; * ANOTHER LINE BIT MAP THAT IS BASED UPON THE ASSOCIATED TX/RX LINE
; * NUMBER/OFFSET TABLE.
; *
; * INPUTS: R5 - CONTAINS THE LINE BIT MAP TO BE TRANSFORMED.
; * TXRXLB - BASE ADDRESS OF ASSOCIATED TX/RX LINE NUMBER TABLE.
; *
; * OUTPUTS: R5 - CONTAINS AN ASSOCIATED LINE BIT MAP.
; *
; * CALLING SEQUENCE: JSR PC,CONMAP
; *
; * COMMENTS: THE TX/RX ASSOCIATION TABLE MUST BE INITIALISED BEFORE THIS
; * ROUTINE IS CALLED.
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
; - - *****

```

```

3398 017402
      017402 004567 165716
3399 017406 012702 005234
3400 017412 010503
3401 017414 012704 000020
3402 017420 005005
3403 017422 006203
3404 017424 103005
3405 017426 011201
3406 017430 006201
3407 017432 004767 001250
3408 017436 050005
3409 017440 005722
3410 017442 005304
3411 017444 001366
3412 017446
      017446 010566 000014
      017452 004736
3413
3414 017454 000207

```

```

CONMAP::SAVE
      JSR      R5,PREG05 ;SAVE CONTENTS OF GPRS R0 THRU R5.
      MOV     #TXRXLB,R2 ;GET THE BASE ADDRESS OF THE LINE ASSOC TABLE.
      MOV     R5,R3      ;COPY THE BIT MAP TO BE TRANSFORMED.
      MOV     #NUMLNS,R4 ;SET MAX LINE COUNTER.
      CLR     R5         ;CLEAR ASSOCIATED LINE BIT MAP.
2$:   ASR     R3         ;SHIFT ACTLNS BIT MAP INT BOOLEAN REGISTER.
      BCC     4$        ;SKIP SETTING ASSOCIATED LINE NUMBER BIT MAP.
      MOV     (R2),R1   ;GET ASSOCIATED LINE NUMBER OFFSET FROM TABLE.
      ASR     R1         ;SHIFT RIGHT TO GET LINE NUMB FROM OFFSET.
      JSR     PC,LINBIT ;GENERATE AN SINGLE BIT MAP FOR THIS LINE.
      BIS     R0,R5     ;SET BIT FOR THIS LINE IN ASSOCIATED BIT MAP.
4$:   TST     (R2)+    ;INCREMENT ADDRESS FOR THE NEXT LINE NUMBER.
      DEC     R4         ;DECREMENT LINE COUNT.
      BNE     2$        ;LOOP IF NOT DONE.
60$:  PASS     R5       ;RESTORE GPRS, EXCEPT
      MOV     R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
      JSR     PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
;R5 - CONTAINS THE ASSOCIATED LINE BIT MAP.
      RTS     PC

```


3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445

017456
017456 004567 165642
017462 010401
017464 012702 177777
017470 005003
017472 012704 017514
017476 004767 001506
017502 103002
017504 004767 002314
017510
017510 004736
017512 000207
017514 177777

```

.SBTTL GLOBAL SUBROUTINE - DELAY -
;*****
;* - DELAY SUBROUTINE -
;* THIS SUBROUTINE IS USED TO DELAY A VARIABLE NUMBER OF MILLI-SECONDS.
;*
;* INPUTS: R4 - CONTAINS THE NUMBER OF MS TO DELAY.
;* MSLCNT.
;*
;* OUTPUTS: NONE.
;*
;* CALLING SEQUENCE: JSR PC,DELAY
;*
;* COMMENTS: IF NO HARDWARE CLOCK INTERRUPTS ARE OCCURRING, CONTROL-CS WILL
;* NOT BE HONORED FOR THE DURATION OF THE DELAY.
;*
;* SUBORDINATE ROUTINES CALLED: NONE.
;*****
DELAY:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;*****
MOV R4,R1 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV #1,R2 ;PASS NUMBER OF MS DELAY AS TIME-OUT VALUE.
CLR R3 ;TELL MSLOOP ROUTINE TO CHECK ALL BITS.
MOV #62,R4 ;TELL MSLOOP RTN TO CHECK FOR ALL BITS CLEAR.
JSR PC,MSLOOP ;TELL MSLOOP TO CHECK DUMMY NON-ZERO WORD.
BCC 60$ ;DELAY THE REQUESTED # OF MS.
JSR PC,OOPS ;EXIT ROUTINE IF WE TIMED-OUT.
PASS 60$ ;IF NO TIME-OUT, BAD PROGRAM OR HOST MACHINE.
;RESTORE GPRS.
RTS PC JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
;*****
62$: .WORD -1 ;DUMMY, NON-ZERO WORD.

```

3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480

017516
017516 004567 165602
017522 016700 162474
017526 012702 000006
017532 006300
017534 005302
017536 001375

017540 012701 000052
017544 032700 000100
017550 001402
017552 012701 000025

017556 060100

017560
017560 010066 000002
017564 004736
017566 000207

```
.SBTTL GLOBAL SUBROUTINE - DM16B -
; * *****
; * - CONVERT TO A 16-BIT PHYSICAL ADDRESS -
; * THIS ROUTINE CONVERTS FROM PAR FORM TO A 16-BIT PHYSICAL ADDRESS,
; * OF ALTERNATE 1'S AND 0'S.
; *
; * INPUTS:          DMTSTA: - CONTAINS THE ADDRESS IN PAR FORM
; *
; * OUTPUTS:         RO - CONTAINS THE 16 BIT PHYSICAL ADDRESS
; *
; * CALLING SEQUENCE:      JSR      PC,DM16B
; *
; * COMMENTS:         USED IN THE DMA ADDRESS TEST
; *
; * SUBROUTINES CALLED:   NONE.
; * *****
DM16B:: SAVE
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV DMTSTA,RO ;SHIFT THE DMA TEST ADDRESS
MOV #6,R2 ;SIX PLACES LEFT , TO
2$: ASL RO ;CONVERT IT INTO A ,
DEC R2 ;16-BIT PHYSICAL ADDRESS
BNE 2$ ;
;
MOV #52,R1 ;SET UP THE 6 LSB'S
BIT #100,RO ;IF BIT #6 OF THE PHYSICAL
BEQ 4$ ;ADDRESS IS CLEAR THEN BRANCH
MOV #25,R1 ;OTHERWISE CORRECT THE LSB'S
;
4$: ADD R1,RO ;MREG THE LSB'S WITH THE PHY ADDR
;
PASS RO ;RETURN WITH THE PHY ADDR.
MOV RO,ROSL0T(SP) ;PUT RO IN STACK SLOT.
JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC
```

3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537

017570
017570 004567 165530
017574 010004
017576 005767 162516
017602 001003
017604 004767 177706
017610 000416
017612 016777 162404 162520 6\$:
017620 012700 140052
017624 032767 000001 162370
017632 001402
017634 012700 140025
017640 012777 000001 162446 8\$:
017646 005705 10\$:

```
.SBTTL GLOBAL SUBROUTINE - DMRW -
; * *****^*****
; * - READ/WRITE DATA FROM/TO (DMTSTA) -
; * THIS ROUTINE READS DATA BYTES FROM OR WRITES DATA BYTES TO AN ADDR OF
; * ALTERNATE 1'S AND 0'S . BITS 21 TO 6 OF THE ADDR ARE CONTAINED AT
; * DMTSTA. THE ROUTINE APPENDS THE 6 LSB'S TO PRODUCE AN ADDR OF
; * ALTERNATE 1'S AND 0'S. THIS ROUTINE IS CALLED FROM THE DMA ADDRESS TEST.
; *
; * INPUTS:
; * R0 - ADDRESS OF THE DATA TO BE WRITTEN TO (DMTSTA),
; * IF A WRITE IS SPECIFIED.
; * R1 - ADDRESS OF THE AREA IN WHICH DATA FROM (DMTSTA),
; * IS TO BE SAVED, IF A READ IS SPECIFIED.
; * R3 - NUMBER OF DATA BYTES TO BE READ/WRTTEN
; * R5 - CLEAR , SPECIFIES A READ FROM (DMTSTA)
; * SET , SPECIFIES A WRITE TO (DMTSTA).
; * DMTSTA - CONTAINS BITS 21 TO 6 OF THE ADDR.
; * MMSRO - ADDRESS OF MEM MGT STATUS REG #0
; * MIPRES - BIT #0 SET, INDICATES MEM MGT PRESENT
; * R6A - ADDRESS OF MEM MGT PAR #6
; * TP4FLG - 004 TRAP FLAGS
; *
; * OUTPUTS:
; * DATA AT (DMTSTA) SAVED OR WRITTEN
; * PAR #6 - CONTENTS SET TO CONTENTS OF DMTSTA
; * TP4FLG - CLEAR IF READ/WRITE SUCCESSFUL
; * SET IF FAIL.
; *
; * CALLING SEQUENCE: JSR PC,DMRW
; *
; * COMENTS: IF MEM MGT IS PRESENT THE SUBROUTINE USES (DMTSTA)
; * AS THE PAGE ADDRESS . PLACING IT IN PAR #6, AND CREATES
; * A VIRTUAL ADDR IN THE RANGE OF PAR #6 WHICH CONTAINS
; * THE SIX LSB'S.
; * IF IT IS NOT PRESENT THE (DMTSTA) IS CONVERTED INTO
; * THE EQUIVALENT 16 BIT PHYSICAL ADDRESS.
; *
; * SUBORDINATE ROUTINES CALLED: CKTRAP,DM16B.
; * - - *****
```

```
DMRW:: SAVE
JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R0,R4 ;SAVE THE SOURCE ADDR
TST MIPRES ;IF MEM MGT IS PRESENT THEN
BNC 6$ ;JUMP AND SET UP THE PAR #3
JSR PC,DM16B ;OTHERWISE CONVERT DMTSTA INTO A 16-BIT
;PHYSICAL ADDRESS, IN R0.
BR 10$ ;JUMP TO PERFORM THE MOVE
MOV DMTSTA,@PAR6A ;SET PAR #6.
MOV #140052,R0 ;SET THE SIX LSB'S AND CONVERT TO
;A VIRTUAL ADDRESS WITHIN THE INFLUENCE
;OF PAR #6.
BIT #1,DMTSTA ;IF BIT #0 OF DMTSTA IS CLEAR THEN
BEQ 8$ ;AVOID CHANGING THE LSB'S
MOV #140025,R0 ;CHANGE THE LSB'S
MOV #BIT0,@MMSRO ;ENABLE MEM MGT.
TST R5 ;IF A READ IS SPECIFIED THEN
```


3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584 017712
017712 004567 165406
3585 017716 012704 000200
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596 017722
017722 104440
017724 010005
3597 017726
017726 012700 000340
017732 104441
3598 017734 056701 162274
3599 017740 010177 162234
3600 017744 105777 162244
3601 017750 000241
3602 017752 100411
3603 017754 010377 162236
3604 017760 010277 162226
3605 017764 110477 162224

```

.SBTTL GLOBAL SUBROUTINE - DODMA -
*****
- INITIATE DMA TRANSMISSION ROUTINE -
THIS ROUTINE WRITES THE DMA PARAMETER TO THE SPECIFIED DEVICE AND
INITIATES THE DMA TRANSMISSION.
INPUTS: R1 - LINE NUMBER ON WHICH TO INITIATE THE DMA.
R2 - START ADDRESS OF THE DMA BUFFER (16 BIT VIRTUAL).
R3 - CHARACTER COUNT OF THE DMA BUFFER.
CSRA - CONTAINS ADDRESS OF THE DUT CSR.
IESTAT - STORAGE FOR STATES OF THE INTERRUPT ENABLE BITS.
TXAD1A - CONTAINS ADDRESS OF DMA TX BUFFER ADDRESS REG #1.
TXAD2A - CONTAINS ADDRESS OF DMA TX BUFFER ADDRESS REG #2.
TXBFCA - CONTAINS ADDRESS OF DMA CHARACTER COUNT REGISTER.
OUTPUTS: CARRY - SUCCESS FLAG (SET IF DMA_START FOUND CLEAR).
DUT TBUFFAD1 - LS 16 BITS OF DMA BUFFER ADDRESS (INITIALIZED).
DUT TBUFFAD2 - MS 6 BITS OF DMA BUFFER ADDRESS (INITIALIZED).
DMA_START BIT SET.
DUT TBUFFCT - DMA BUFFER CHARACTER COUNT (INITIALIZED).
CALLING SEQUENCE: JSR PC,DODMA
COMMENTS: THIS ROUTINE 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 ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREGOS ;CALL REGISTER SAVE SUBRT.
MOV #200,R4 ;PREPARE TO CLEAR UPPER 6 BITS OF DMA BUFF ADR.

;WRITE THE DMA PARAMETERS OUT TO THE DUT DMA REGISTERS.
;DISABLE INTERRUPTS.
;SET UP DUT 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.
68: GETPRI R5 ;GET THE PRESENT PROCESSOR PRIORITY.
;TRAP C:GPRI
;MOV RO,R5
SETPRI #PRI07 ;DISABLE ALL HARDWARE INTERRUPTS.
;MOV #PRI07,R0
;TRAP C:SPRI
BIS IESTAT,R1 ;PREPARE FOR SETUP OF LINE NUMBER IN DUT CSR.
MOV R1,#CSRA ;SET UP THE DUT CSR IND.ADR.REG FIELD.
TSTB #TXAD2A ;TEST THE DUT DMA_START BIT.
CLC ;INDICATE FAILURE IN CASE DMA.HO BIT IS SET.
BMI 60$ ;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.

```


3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631 020002 004567 165316
3632
3633
3634
3635 020006 005001
3636 020010 012703 000020
3637 020014 016700 162152
3638 020020 012705 000001
3639 020024 030500
3640 020026 001006
3641 020030 006305
3642 020032 005201
3643 020034 020103
3644 020036 002772
3645 020040 000241
3646 020042 000401
3647 020044 000261
3648
3649 020046 010166 000004
020046 010566 000014
020052 010566 000014
020056 004736
3650
3651
3652
3653 020060 000207

```

.SBTTL GLOBAL SUBROUTINE - FINACT -
; ** *****
; * - 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:
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
; - *****
FINACT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
; *
; * FIND AN ACTIVE LINE ON WHICH TO PERFORM THE TEST.
; -
; CLR R1 ;CLEAR THE LINE NUMBER COUNTER.
; MOV #NUMLNS,R3 ;GET MAX LINE NUMBER.
; MOV ACTLNS,R0 ;GET THE ACTIVE LINE BIT MAP.
; MOV #1,R5 ;SET UP A LINE BIT MASK.
2$: BIT R5,R0 ;LOOK FOR AN ACTIVE LINE.
; BNE 4$ ;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 2$ ;LOOP TO TRY THE NEXT LINE.
; CLC ;CLEAR CARRY BIT, NO ACTIVE LINE FOUND.
; BR 60$ ;EXIT WITH FAILURE.
4$: SEC ;SET CARRY, SUCCESS.
60$: PASS R1,R5 ;RESTORE GPRS, EXCEPT
; MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
; MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
; JSR PC,0(SP) ;RETURN TO PREG05 SUBRT.
; R1 - CONTAINS THE NUMBER OF FIRST ACTIVE LINE.
; R5 - CONTAINS THE BIT MAP OF THE ACTIVE LINE.
; CARRY - SET INDICATES SUCCESS.
RTS PC

```

3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695 020062
3696 020066 004567 165236
3697 020072 010167 000226
3698
3699
3700
3701
3702 020076 010067 163020
3703 020102 012700 003124
3704 020106 012720 000004
3705 020112 010220
3706 020114 010320
3707 020116 012720 000001
3708 020122 016710 162044
3709 020126 005104
3710 020130 040420

```

.SBTTL GLOBAL SUBROUTINE - FRPSUP -
;*****
; - FRAMING AND PARITY ERROR TRANSMISSION/RECEPTION SET-UP -
;
; THIS ROUTINE IS USED TO INITIALISE BOTH THE DUT AND THE
; TRANSMISSION/RECEPTION CONTROL PARAMETERS TO THE CORRECT
; STATE, PRIOR TO A FRAMING OR PARITY ERROR DETECTION AND
; REPORTING TEST.
;
; INPUTS:  R0 - LPR CONTENTS FOR LINES IN THE BIT MAP IN GPR4.
;          R1 - LPR CONTENTS FOR LINES NOT IN THE BIT MAP IN GPR4.
;          R2 - START ADDRESS OF DATA PATTERN TO TRANSMIT.
;          R3 - LENGTH OF THE DATA PATTERN TO TX.
;          R4 - LOCAL LINE GROUP BIT MAP.
;          ACTLNS - CONTAINS A BIT MAP OF ALL CURRENTLY ACTIVE LINES.
;          LOPBCK - CONTAINS THE TYPE OF LOOPBACK MODE SELECTED.
;          CBB - LABEL AT BASE OF TX/RX CONTROL BLOCK.
;
; OUTPUTS: THE CONTENTS OF THE TXRCB ARE DESTROYED.
;          THE INDIRECT ADDRESS FIELD OF THE DUT CSR MAY BE DESTROYED.
;          THE DUT'S LPR'S AND LNC'S MAY BE MODIFIED.
;          THE FOLLOWING POINTERS AND COUNTERS ARE INITIALISED;
;          CHCNT,CHRTOT,DPEND,DPLEN,EXCNT,RXCNT,RXDONF,RXPTR,TXCNT,
;          TXDONF,TXPTR,TXRXL.
;
; CALLING SEQUENCE: JSR PC,FRPSUP
;
; COMMENTS: THIS ROUTINE SHOULD BE CALLED TWICE DURING THE TESTING OF
;           THE FRAMING AND PARITY ERROR DETECTION AND REPORTING TEST.
;           SO THAT BOTH LINE GROUPS ARE TESTED ON TRANSMISSION AND
;           RECEPTION.
;           JSR PC,FRPSUP ; DO SET-UP.
;           EXECUTE TEST FOR THE ABOVE SET-UP.
;           COMPLEMENT THE LINE GROUP BIT MAP.
;           JSR PC,FRPSUP ; DO SET UP AGAIN.
;           EXECUTE TEST AGAIN.
;
; SUBORDINATE ROUTINES CALLED: TXRINI.
;*****
FRPSUP:: SAVE ;SAVE THE CONTENTS OF THE GPR'S.
;          JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;          MOV R0,70$ ;SAVE LPR PARAMETER FOR LINE TX.
;          MOV R1,72$ ;SAVE LPR PARAMETER FOR LINE RX.
;
; SET UP THE TRANSMISSION/RECEPTION CONTROL BLOCK TO INITIALISE THE
; ACTIVE LINES IN THE BIT MAP PASSED INTO THIS ROUTINE.
;
; MOV R0,CBB ;SET CONTENTS OF LPR PARAMS IN TX/RX C.BLK.
; MOV #CBB+2,R0 ;GET ADDRESS OF THE NEXT WORD IN THE CNTRL BLK.
; MOV #4,(R0)+ ;LNCTRL PARAMETER, ENABLE RECEIVERS.
; MOV R2,(R0)+ ;START ADDRESS OF DATA PATTERN.
; MOV R3,(R0)+ ;SET DATA PATTERN LENGTH.
; MOV #1,(R0)+ ;NUMBER OF DATA PATTERNS TO TRANSMIT.
; MOV ACTLNS,(R0) ;BIT MAP OF LINES TO INITIALISE.
; COM R4 ;GENERATE A BIT MAP OF ACTIVE LINES IN GRP1.
; BIC R4,(R0)+ ;CLEAR THE UNWANTED LINES.

```



```

3711 020132 116720 162036      MOVB  LOPBCK,(R0)+  ;SET LOOPBACK MODE,STAGGARED.
3712 020136 005200              INC   RO           ;INCREMENT ADDRESS TO GET NEXT WORD IN TABLE.
3713 020140 012710 000001      MOV   #1,(R0)     ;SET AMMOUNT OF OFFSET FOR EACH TX START.
3714
3715
3716      ;+
3717      ; INITIALISE THE DUT AND THE ASSOCIATED POINTERS AND COUNTERS, TO THE STATE
3718      ; DICTATED BY THE CONTENTS OF THE TX/RX CONTROL BLOCK.
3719 020144 004767 005534      JSR   PC,TXRINI   ;INITIALISE DUT.
3720
3721      ;+
3722      ; SET UP CONTROL BLOCK FOR LINES IN GROUP 2.
3723 020150 012700 003122      MOV   #CBB,RO    ;GET START ADDRESS OF CONTROL BLOCK.
3724 020154 010120              MOV   R1,(R0)+   ;SET LPR PARAMETER FOR RX LINES.
3725 020156 062700 000010      ADD   #10,RO     ;SELECT THE ADDRESS OF THE LINE BIT MAP IN C.B.
3726 020162 016710 162004      MOV   ACTLNS,(R0) ;BIT MAP OF LINES TO INITIALISE.
3727 020166 005104              COM   R4         ;GENERATE A BIT MAP OF LINES IN GRP 2.
3728 020170 040410              BIC   R4,(R0)    ;CLEAR THE UNWANTED LINES.
3729
3730      ;+
3731      ; INITIALISE THE DUT AND THE ASSOCIATED POINTERS AND COUNTERS, TO THE STATE
3732      ; DICTATED BY THE CONTENTS OF THE TX/RX CONTROL BLOCK.
3733 020172 004767 005506      JSR   PC,TXRINI   ;INITIALISE DUT.
3734
3735      ;+
3736      ; SET-UP THE REQUIRED LPR PARAMETERS NEEDED FOR THE CORRECT RECEPTION OF DATA
3737      ; ON ASSOCIATED IN-ACTIVE LINES.
3738
3739      ;+
3740      ; INITIALISE LPR PARAMETERS FOR INACTIVE LINES IN GROUP 2.
3741
3742      ;+
3743 020176 012701 177777      MOV   #MAPLNS,R1 ;SET UP BIT MAP CORRESPONDING TO ALL LINES.
3744 020202 016702 161764      MOV   ACTLNS,R2  ;GET THE ACTIVE (TX) LINE BIT MAP.
3745 020206 005101              COM   R1         ;GENERATE A BIT MAP OF NONE EXISTANT LINES.
3746 020210 005102              COM   R2         ;GENERATE A BIT MAP OF INACTIVE LINES.
3747 020212 040102              BIC   R1,R2     ;CLEAR ANY "NONE EXISTANT" INACTIVE LINES.
3748 020214 040402              BIC   R4,R2     ;
3749 020216 010267 162712      MOV   R2,CBMAPA  ;SET UP BIT MAP IN CONTROL BLOCK.
3750 020222 005067 162704      CLR   CBOPNA     ;CLEAR REPEAT TX COUNT IN CONTROL BLOCK.
3751 020226 016767 000072 162666  MOV   72$,CBLPRA ;SET-UP COMPLEMENTARY LPR PARM.
3752 020234 004767 005444      JSR   PC,TXRINI  ;INITIALISE INACTIVE LINES.
3753
3754      ;+
3755      ; INITIALISE LPR PARAMETERS FOR INACTIVE LINES IN GROUP 1.
3756      ;+
3757 020240 016702 161726      MOV   ACTLNS,R2  ;GET THE ACTIVE (TX) LINE BIT MAP.
3758 020244 005102              COM   R2         ;GENERATE A BIT MAP OF INACTIVE LINES.
3759 020246 040102              BIC   R1,R2     ;CLEAR ANY NONE EXISTANT INACTIVE LINES.
3760 020250 005104              COM   R4         ;
3761 020252 040402              BIC   R4,R2     ; ONLY PASS LGRP2 ASSOCIATED LINE BIT MAP.
3762 020254 010267 162654      MOV   R2,CBMAPA  ;SET-UP BIT MAP IN CONTROL BLOCK.
3763 020260 016767 000036 162634  MOV   70$,CBLPRA ;SET UP COMPLEMENTARY LPR PARAM FOR LGRP1.
3764 020266 004767 005412      JSR   PC,TXRINI  ;INITIALISE INACTIVE LINES IN LGRP1.
3765
3766      ;+
3767      ; DISABLE RECEIVERS ON ALL LINES TO ENSURE THAT ONLY THE RECEIVERS OF THE
3768      ; ASSOCIATED ACTIVE (TX) LINES ARE ENABLED.(STAGGARED LOOPBACK)
3769      ; RE-ENABLE RECEPTION ON THE CORRECT ASSOCIATED LINES.
3770

```



```

3783 .SBTTL GLOBAL SUBROUTINE - GETBDR -
3784 ;* *****
3785 ;* - GET BAUDRATE SUBROUTINE -
3786 ;* THIS ROUTINE REQUESTS A BAUDRATE INPUT FROM THE OPERATOR. THIS
3787 ;* BAUDRATE IS LOOKED UP IN A TABLE TO GIVE THE LPR BAUDRATE FIELD
3788 ;* VALUE WHICH IS ASSOCIATED WITH THAT BAUDRATE.
3789 ;*
3790 ;* INPUTS: BDRMSG - LABEL AT THE BAUDRATE PROMPT MESSAGE.
3791 ;* BRTBLE - LABEL AFTER END OF THE BAUDRATE TABLE.
3792 ;* UBRFMT - LABEL AT THE UNSUPPORTED BAUDRATE MESSAGE.
3793 ;*
3794 ;* OUTPUTS: R1 - BAUDRATE CODE IN LS 4 BITS.
3795 ;*
3796 ;* CALLING SEQUENCE: JSR PC,GETBDR
3797 ;*
3798 ;* COMMENTS:
3799 ;*
3800 ;* SUBORDINATE ROUTINES CALLED: NONE.
3801 ;*
3802 ;* *****
3803 GETBDR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
3804 020326 004567 164772 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
3805 020332 016705 161672 MOV GMANWD,R5 ;SAVE THE GMAINIX VALUE.
3806 ;*
3807 ; PROMPT THE OPERATOR: "MODEM BAUDRATE IN BPS: (D) 1200 ?"
3808 020336 012767 002260 161664 2$: MOV #1200,GMANWD ;SET UP DEFAULT VALUE TO 1200 BAUD.
3809 020344 GMANID BDRMSG,GMANWD,D,177777,0,38400.,YES
3810 020344 104443 TRAP C$GMAN
3811 020346 000406 BR 10000$
3812 020350 002230 .WORD GMANWD
3813 020352 000052 .WORD T$CODE
3814 020354 010020 .WORD BDRMSG
3815 020356 177777 .WORD 177777
3816 020360 000000 .WORD T$L0LIM
3817 020362 113000 .WORD T$HILIM
3818 020364 10000$:
3819 MOV GMANWD,R2
3820 ;*
3821 ; ATTEMPT TO LOOK THE VALUE UP IN THE BAUDRATE TABLE.
3822 ;*
3823 MOV #15.,R1 ;INITIALIZE BAUDRATE CODE TO HIGHEST BAUDRATE.
3824 MOV #BRTBLE,R3 ;INITIALIZE BAUDRATE POINTER.
3825 4$: CMP R2,-(R3) ;COMPARE BAUDRATE WITH A TABLE ENTRY.
3826 BEQ 60$ ;BAUDRATES COMPARE? YES, EXIT WITH CODE.
3827 DEC R1 ;NO, SET BAUDRATE CODE TO NEXT LOWER BAUDRATE.
3828 BNE 4$ ;DONE? NO, LOOP.
3829 CMP R2,-(R3) ;CHECK IF LAST BAUDRATE MATCHES.
3830 BEQ 60$ ;BAUDRATES MATCH? YES, EXIT WITH CODE.
3831 ;*
3832 ; REPORT "NNNNN IS NOT A SUPPORTED BAUDRATE, ENTER ANOTHER OR CTRL C."
3833 ;*
3834 PRINT #UBRFMT,R2
3835 MOV R2,(SP)
3836 020414 010246

```

```

020416 012746 007722
020422 012746 000002
020426 010600
020430 104417
020432 062706 000006
3829 020436 000737 BR 2$ ;LOOP TO GET ANOTHER BAUDRATE.
3830
3831 020440 010567 161564 60$: MOV R5,GMANWD ;RESTORE THE GMANIX PARAMETER VALUE.
3832 020444 PASS R1 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
020444 010166 000004 MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
020450 004'36 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
3833 020452 000207 RTS PC ; R1 - BAUDRATE CODE.

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

```

3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880

020454
020454 004567 164644
020460 005000
020462 005002
020464 005767 162226
020470 001416
020472 016704 162214
020476 011402
020500 005024
020502 020427 003120
020506 103402
020510 012704 002720
020514 010467 162172
020520 005367 162172
020524 000261
020526
020526 010266 000006
020532 004736
020534 000207

```

.SBTTL GLOBAL SUBROUTINE - GETCHR -
;*****
; - GET A CHARACTER FROM THE RX BUFFER ROUTINE -
; THIS SUBROUTINE GETS A CHARACTER FROM THE RX BUFFER WHICH IS IN THE
; HOST SYSTEM MEMORY. IF THE BUFFER IS EMPTY UPON ENTRY OF THIS ROUTINE
; THIS ROUTINE RETURNS A NULL CHARACTER WITH DATA,VALID CLEAR AND A
; BUFFER EMPTY INDICATION.
;
; INPUTS: RXBCNT - RX BUFFER CHARACTER COUNT,
;          RXBEND - LABEL AFTER END OF THE RX BUFFER AREA IN MEMORY,
;          RXBETX - EQUATED TO RX BUFFER LEVEL AT WHICH TO ENABLE TX,
;          RXBOPT - POINTER TO NEXT AVAILABLE INPUT SLOT OF RX BUFFER,
;          RXBSTA - LABEL AT START OF RX BUFFER AREA IN MEMORY.
;
; OUTPUTS: R2 - CHARACTER WHICH IS READ FROM THE BUFFER,
;          RXBOPT - UPDATED TO POINT TO NEXT INPUT SLOT OF RX BUFFER,
;          RXBCNT - RX BUFFER CHARACTER COUNT (UPDATED),
;          CARRY - "SUCCESS" FLAG (SET IF BUFFER IS NOT EMPTY ON ENTRY).
;
; CALLING SEQUENCE: JSR PC,GETCHR
;
; COMMENTS:
;
; SUBORDINATE ROUTINES CALLED: NONE.
;*****
GETCHR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5,
;          JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;          CLR R0 ;CLEAR THE "RE-ENABLE" TX FLAG (SUBRTN OUTPUT).
;          CLR R2 ;GET NULL CHAR IN CASE BUFFER IS EMPTY.
;          TST RXBCNT ;CHECK FOR RX BUFFER EMPTY, CLEAR CARRY.
;          BEQ 60$ ;EXIT THE ROUTINE IF BUFFER IS EMPTY.
;          MOV RXBOPT,R4 ;GET THE BUFFER OUTPUT POINTER.
;          MOV (R4),R2 ;GET A CHARACTER FROM THE BUFFER.
;          CLR (R4)+ ;DELETE THE READ CHARACTER FROM THE BUFFER.
;          CMP R4,#RXBEND ;CHECK IF POINTER SHOULD WRAP AROUND.
;          BLO 2$ ;SKIP WRAPAROUND IF POINTER IS NOT AT END.
;          MOV #RXBSTA,R4 ;WRAP INPUT POINTER AROUND.
;          MOV R4,RXBOPT ;UPDATE THE OUTPUT POINTER STORAGE.
;
;          DEC RXBCNT ;REMOVE THIS CHAR FROM THE BUFFER COUNT.
;          SEC ;SET SUCCESS FLAG, BUFFER WAS NOT EMPTY.
;
;          60$: PASS R2 ;RESTORE GPRS, EXCEPT
;          MOV R2,R2SLOT(SP) ;PUT R2 IN STACK SLOT.
;          JSR PC,#(SP)+ ;RETURN TO PREG05 SUBRT.
;          ;R2 - CONTAINS THE CHARACTER READ FROM BUFFER.
;          ;CARRY-"SUCCESS" FLAG, SET IF BUFFER NOT EMPTY.
;          RTS PC

```

3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901 020536
020536 004567 164562
3902 020542 000301
3903 020544 042701 177400
3904 020550 010102
3905 020552 042701 000360
3906 020556 006202
3907 020560 006202
3908 020562 006202
3909 020564 006202
3910 020566 020102
3911 020570 101401
3912 020572 010201
3913 020574 116102 005214
3914 020600 042702 177400
3915 020604 010267 161436
3916
3917 020610
020610 004736
3918 020612 000207

```

.SBTTL GLOBAL SUBROUTINE - GETTIM -
; * *****
; * - GET TIME-OUT VALUE BASED ON MINIMUM BAUDRATE ROUTINE -
; * THIS SUBROUTINE GETS THE NECESSARY TIME-OUT VALUE TO VERIFY THAT ALL
; * CHARS HAVE BEEN RECEIVED AT THE COMPLETION OF THE TX/RX OF A DATA
; * PATTERN. THIS USES THE SLOWEST BAUDRATE WHICH IS SPECIFIED IN THE
; * PASSED IN DUT LPR CONTENTS TO CALCULATE THIS TIME-OUT VALUE.
; *
; * INPUTS: R1 - DUT LPR CONTENTS.
; *
; * OUTPUTS: RXTOUT - TIME-OUT VALUE FOR WAITING FOR LAST RX CHAR.
; *
; * CALLING SEQUENCE: JSR PC,GETTIM
; *
; * COMMENTS:
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
; * *****
GETTIM:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
SWAB R1 ;PUT THE BAUD RATE FIELDS IN THE LOW BYTE.
BIC #177400,R1 ;CLEAR STOP,PARITY,AND CHAR FIELDS.
MOV R1,R2 ;COPY BAUD RATE FIELDS.
BIC #360,R1 ;SELECT RX BAUD RATE FIELD ONLY.
ASR R2 ;SHIFT TX BAUD RATE FIELD
; TO OCCUPY THE LOW FOUR BYTES.
ASR R2
ASR R2
;
CMP R1,R2 ;CHECK IF SAME BAUD RATE IN EACH FIELD.
BLOS 2$ ;BRANCH IF RX BAUD RATE IS LOWER OR SAME.
MOV R2,R1 ;TX BAUD RATE IS THE SLOWER OF THE TWO.
2$: MOVB PROTB(R1),R2 ;GET PROPORTIONAL DELAY FROM TABLE.
BIC #177400,R2 ;CLEAR UPPER BYTE BECAUSE OF SIGN EXTENSION.
MOV R2,RXTOUT ;LOAD THE RX TIME-OUT VARIABLE.
;
60$: PASS ;RESTORE GPRS.
; PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC JSR

```



```

3976 020652 004767 177034      JSR    PC,DODMA
3977 020656 103403              BCS    6$          ;SKIP ERROR IF DODMA WAS SUCCESSFUL.
3978                          ;+
3979                          ; SET THE PROPER BIT OF THE TX INTERRUPT FLAGS TO INDICATE THE LINE ERROR.
3980                          ;-
3981 020660 050567 161400      BIS    R5,TXINTF   ;INDICATE THE ERROR.
3982 020664 000402              BR     10$         ;SKIP UPDATING POINTERS AND COUNTERS.
3983                          ;+
3984                          ; UPDATE THE TX CHARACTER COUNT FOR THIS LINE.
3985                          ;-
3986 020666 060364 003502      6$:   ADD    R3, TXCNTB(R4) ;ADD THE DATA PATTERN LENGTH TO TX CHAR COUNT.
3987                          ;+
3988                          ; INCREMENT LINE COUNTER,GOTO NEXT LINE IF NOT DONE.
3989                          ;-
3990 020672 005201              10$:   INC    R1
3991 020674 020127 00002C      CMP    R1,#NUMLNS ;INCREMENT THE LINE COUNTER.
3992 020700 002752              BLT    2$          ;COMPARE THE LINE COUNTER WITH NUMBER OF LINES.
3993                          ;LOOP TO SFND CHAR TO ANOTHER LINE IF NOT DONE.
3994 020702                      60$:   PASS
3995 020704 000207              RTS    PC          ;RESTORE GPRS.
                                           JSR    PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

```


3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021 020706
020706 004567 164412
4022 020712 042701 177760
4023 020716 006301
4024 020720 016100 002364
4025 020724
020724 010066 000002
020730 004736
4026 020732 000207

```

.SBTTL 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 DUT CSRS CAN BE PASSED TO THIS ROUTINE WITHOUT ERROR).
;*
;* INPUTS: R1 - LINE NUMBER (ONLY LS 4 BITS USED, OTHERS DISREGARDED).
;* BITTBL - BASE LABEL OF A 16 WORD BIT TABLE.
;*
;* OUTPUTS: R0 - BIT MAP, BIT CORRESPONDING TO LINE NUMBER IS SET;
;* IF LINE NUMBER IS 3, THEN BIT3 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 DUT (IE - LESS THAN NUMLNS).
;* NOTE: THE LINE NUMBER IS NOT DESTROYED OR ALTERED, SO THIS
;* ROUTINE CAN BE USED EASILY IN LOOPS.
;*
;* SUBORDINATE ROUTINES CALLED: NONE.
;*****
LINBIT:: SAVE
;SAVE CONTENTS OF GPRS R0 THRU R5.
R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;MASK OUT ALL BUT 4 LSBITS OF THE LINE #.
;MULTIPLY LINE # BY 2 TO GET WORD TABLE OFFSET.
;GET THE SINGLE BIT BIT MAP.
;RESTORE GPRS, EXCEPT THE FOLLOWING.
R0,ROSL0T(SP) ;PUT R0 IN STACK SLOT.
PC,@(SP); ;RETURN TO PREG05 SUBRT.
;R0 - BIT MAP WITH LINE # BIT SET.

```

4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058 020734
020734 004567 164364
4059 020740
020740 005067 161532
4060 020744
020744 005067 161314
4061 020750
020750 005067 161526
4062 020754
020754 005067 161524
4063
4064
4065
4066 020760
020760 010167 162136
4067 020764
020764 012701 003122
4068 020770
020770 005201
4069 020772
020772 005201
4070 020774
020774 012721 011004
4071 021000
021000 010221
4072 021002
021002 010321
4073 021004
021004 012721 000001
4074 021010
021010 016721 161156
4075 021014
021014 112721 000003
4076 021020
021020 005201
4077 021022
021022 012711 000002
4078
4079
4080
4081
4082 021026
021026 004767 004652
4083

```

.SBTTL GLOBAL SUBROUTINE - MODSUP -
;*****
; - MODEM LOOPBACK TX/RX SET-UP ROUTINE -
;
; THIS ROUTINE IS USED TO INITIALISE BOTH THE DUT AND THE
; TRANSMISSION/RECEPTION CONTROL PARAMETERS TO THE CORRECT
; STATE, PRIOR TO A MODEM LOOPBACK TEST DATA PATTERN TX/RX.
;
; INPUTS: R1 - TX, RX LPR CONTENTS.
; R2 - START ADDRESS OF DATA PATTERN TO TRANSMIT.
; R3 - LENGTH OF DATA PATTERN.
; ACTLNS - CONTAINS A BIT MAP OF ALL CURRENTLY ACTIVE LINES.
; CBB - LABEL AT BASE OF TX/RX CONTROL BLOCK.
;
; OUTPUTS: THE CONTENTS OF THE TX/RX CONTROL BLOCK (CCB) ARE DESTROYED.
; THE INDIRECT ADDRESS FIELD OF THE DUT CSR MAY BE DESTROYED.
; THE DUT'S LPR'S AND LNC'S MAY BE MODIFIED.
; THE FOLLOWING POINTERS AND COUNTERS ARE INITIALISED;
; CHCNT,CHRTOT,DPEND,DPLEN,EXCNT,RXCNT,RXPTR,TXCNT,
; TXPTR,TXRXL.
; CHRTOT, RXDNF, TXDNF AND TXINTF ARE CLEARED.
;
; CALLING SEQUENCE: JSR PC,MODSUP
;
; COMMENTS: DUT IS SET UP WITH DSR AND DTR SET. ONE DATA PATTERN IS
; SENT AND RECEIVED FROM EACH LINE.
;
; SUBORDINATE ROUTINES CALLED: CONMAP,RXENBL,TXRINI.
;*****
MODSUP:; SAVE
;SAVE CONTENTS OF THE GPR'S R0 THRU R5.
;R5,PREGOS ;CALL REGISTER SAVE SUBRT.
;CLEAR TOTAL RECEIVED CHAR COUNTER.
;CLEAR FLAGS USED TO LOG DMA H.OVER ERRORS.
;CLEAR THE TX DONE FLAGS.
;CLEAR THE RX DONE FLAGS.
;
; SET UP THE TRANSMISSION/RECEPTION CONTROL BLOCK TO THE DESIRED STATE.
;
; MOV R1,CBB ;SET CONTENTS OF LPR PARAMS IN TX/RX C.BLK.
;MOV @CBB,R1 ;GET BASE ADDRESS OF CONTROL BLOCK.
;INC R1 ;INCREMENT ADDRESS FOR NEXT WORD
;INC R1 ;INITIALISE THE FOLLOWING IN THE CNTRL.BLK:
;MOV @11004,(R1); LNCtrl: RTS, DTR, ENABLE RECEIVERS.
;MOV R2,(R1); ; START ADDRESS OF DATA PATTERN.
;MOV R3,(R1); ; DATA PATTERN LENGTH.
;MOV @1,(R1); ; NUMBER OF DATA PATTERNS TO TRANSMIT.
;MOV ACTLNS,(R1); ; BIT MAP OF LINES TO INITIALISE.
;MOVB @3,(R1); ;SET LOOPBACK MODE TO H325.
;INC R1 ;INCREMENT ADDRESS FOR THE NEXT WORD.
;MOV @2,(R1) ;SET AMOUNT OF OFFSET EACH TX STARTS AT TO .
;
; INITIALISE THE DUT AND THE ASSOCIATED POINTERS AND COUNTERS, TO THE STATE
; DICTATED BY THE CONTENTS OF THE TX/RX CONTROL BLOCK.
;
; JSR PC,TXRINI ;INITIALISE DUT.
;

```

D9

DMU 11 FUNC ISI PART4
GLOBAL SUBROUTINE

MACRO M1200 15 MAR 84 09:48 PAGE 75-1
= NUDSUP

SEQ 107

```

4084                                     ; INITIALISE POINTERS AND COUNTERS FOR INACTIVE LINES TO ZERO.
4085                                     ;
4086 021032 012701 177777                MOV  #MAPLNS,R1          ;GET THE LINE BIT MAP FOR ALL LINES.
4087 021036 016702 161130                MOV  ACTLNS,R2          ;GET THE ACTIVE LINE BIT MAP.
4088 021042 005101                        COM  R1
4089 021044 005102                        COM  R2
4090 021046 040102                        BIC  R1,R2             ;GENERATE AN IN-ACTIVE LINE BIT MAP.
4091 021050 010267 162060                MOV  R2,CBMAPA         ;MOVE BIT MAP TO THE CONTROL BLOCK.
4092 021054 005067 162044                CLR  CBLNCA           ;CLEAR THE LNCTRL SET UP PARAMETERS.
4093 021060 005067 162046                CLR  CBDPNA           ;CLEAR THE REPEAT TX COUNT IN CNTRL BLCK.
4094 021064 004767 004614                JSR  FC,TXRINI        ;SET UP PARAMETERS FOR INACTIVE LINES.
4095
4096 021070                                60$: PASS              ;RESTORE GPR'S.
      021070 004736                                JSR  PC,(SP)+         ;RETURN TO PREG05 SUBRT.
4097 021072 000207                RTS  PC

```

4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137 021074
021074 004567 164224
4138
4139
4140
4141
4142 021100 005102
4143 021102 040203
4144
4145
4146
4147 021101 005701
4148 021101 001011
4149 021110 011400
4150 021111 010067 000070
4151 021116 040200
4152 021120 020003
4153 021122 000261
4154 021124 001420

```

.SBTTL GLOBAL SUBROUTINE - MSLGET -
*****
; * - MILLI SECONDS LOOP WHICH RETURNS READ WORD AND REMAINING TIME -
; * THIS SUBROUTINE IS A GENERAL PURPOSE TEST LOOP SUBROUTINE. IT IS USED
; * TO VERIFY THAT A CERTAIN ACTION OCCURS BEFORE A TIME-OUT PERIOD. THE
; * CALLING ROUTINE PASSES IN WHICH BITS SHOULD BE SET AND CLEARED FOR THE
; * DESIRED CONDITION AND THE TIME-OUT VALUE IN MILLI-SECONDS.
; * THIS ROUTINE CHECKS FOR THE DESIRED CONDITION UPON ENTRANCE INTO THE
; * ROUTINE AND THEN ONCE EACH MILLI-SECOND 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 FIELDS IN R2.
; * R4 - ADDRESS OF THE WORD TO TEST.
; * MSLCNT - MILLI SECOND SOFTWARE LOOP COUNT.
; *
; * OUTPUTS: R0 - THE LAST WORD WHICH WAS READ TO CHECK FOR THE CONDITION.
; * R1 - REMAINING NUMBER OF MS IN TIME-OUT TIME.
; * CARRY - SUCCESS FLAG (SET IF CONDITION IS MET BEFORE TIME-OUT).
; *
; * CALLING SEQUENCE: JSR PC,MSLGET
; *
; * COMMENTS: THIS ROUTINE WORKS WITH OR WITHOUT A HARDWARE CLOCK, BUT THE
; * CALIBRATION IS ONLY GUARENTEED WHEN A LINE CLOCK IS AVAILABLE
; * ON THE SYSTEM.
; * THIS ROUTINE CAN BE USED AS A DELAY ROUTINE, BY SPECIFYING THE
; * DESIRED DELAY AS THE TIME-OUT AND SPECIFYING A CONDITION TO
; * LOOK FOR WHICH WILL NOT BE MET DURING THE DELAY.
; * IF A TIME-OUT VALUE OF 0 IS SPECIFIED, THIS ROUTINE CHECKS FOR
; * THE DESIRED CONDITION BEFORE RETURNING. IT INDICATES SUCCESS
; * IF THE CONDITION IS MET, FAILURE OTHERWISE.
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
*****
MSLGET:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
; *
; * SET UP MASK FOR REMOVING UNUSED BITS IN THE TEST WORD, AND CLEAR UNUSED
; * BITS IN THE DESIRED STATE WORD TO ALLOW DIRECT COMPARISON.
; *
; * COM R2 ;GET MASK OF UNUSED BITS.
; * BIC R2,R3 ;MASK OUT UNUSED BITS IN DESIRED STATE WORD.
; *
; * HANDLE THE TEST AND EXIT IF WE HAVE A 0 TIME-OUT VALUE.
; *
; * TST R1 ;TEST THE TIME-OUT VALUE FOR ZERO.
; * BNE 2$ ;IF NON-ZERO TIME-OUT, GO LOOP AND TEST.
; * MOV (R4),R0 ;GET THE WORD TO TEST BEFORE EXITING.
; * MOV R0,62$ ;SAVE VALUE SO WE CAN RETURN IT.
; * BIC R2,R0 ;MASK OUT UNTESTED BITS OF WORD.
; * CMP R0,R3 ;COMPARE AGAINST DESIRED STATE WORD.
; * SEC ;INDICATE SUCCESS IN CASE WORDS ARE EQUAL.
; * BEQ 6$ ;EXIT WITH SUCCESS IF WORDS ARE EQUAL.

```

```

4155 021126 000241          CLC          ;INDICATE FAILURE (TIME-OUT).
4156 021130 000416          BR          6$          ;EXIT WITH FAILURE, WORDS AREN'T EQUAL.
4157                          ;+
4158                          ; NON-ZERO TIME-OUT VALUE. LOOP, WAITING FOR CONDITION OR TIME-OUT.
4159                          ;-
4160 021132 016705 161154   2$:      MOV      MSLCNT,R5      ;LOAD MS LOOP COUNT.
4161 021136 011400          4$:      MOV      (R4),R0      ;GET THE WORD TO TEST.
4162 021140 010067 000042   4$:      MOV      R0,62$      ;SAVE WORD IN CASE THIS IS THE LAST.
4163 021144 040200          BIC      R2,R0      ;MASK OUT UNTESTED BITS OF WORD.
4164 021146 020003          CMP      R0,R3      ;COMPARE AGAINST DESIRED STATE WORD.
4165 021150 000261          SEC          ;SET CARRY IN CASE OF SUCCESS.
4166 021152 001405          BEQ      6$          ;EXIT WITH SUCCESS IF WORDS ARE EQUAL.
4167 021154 005305          DEC      R5          ;COUNT DOWN THE INSIDE MS LOOP COUNT.
4168 021156 001367          BNE      4$          ;LOOP IF MS NOT UP.
4169 021160 005301          DEC      R1          ;DECREMENT THE MS TIME COUNT.
4170 021162 001363          BNE      2$          ;IF TIME NOT UP, LOOP TO COUNT ANOTHER MS.
4171 021164 000241          CLC          ;CLEAR CARRY, WE TIMED-OUT.
4172                          ;+
4173                          ; HAVE EITHER FOUND CONDITION, OR TIMED-OUT (POSSIBLY FROM 0 TIME-OUT VALUE).
4174                          ; RESTORE THE LAST CONTENTS READ FROM THE TEST WORD. EXIT ROUTINE.
4175                          ;-
4176 021166 016700 000014   6$:      MOV      62$,R0      ;PASS OUT THE LAST READ WORD.
4177 021172          60$:     PASS      R0,R1      ;RESTORE GPRS, EXCEPT THE FOLLOWING:
                                MOV      RO,R0SLOT(SP) ;PUT R0 IN STACK SLOT.
                                MOV      R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
                                JSR      PC,@(SP)+    ;RETURN TO PREG05 SUBRT.
4178                          ;R0 - LAST READ WORD CHECKED FOR CONDITION.
4179                          ;R1 - REMAINING TIME (0 IF TIME-OUT OCCURED).
4180 021204 000207          RTS      PC      ;CARRY - SET IF SUCCESS, CLEAR IF TIME-OUT.
4181                          ;+
4182                          ; LOCAL STORAGE.
4183                          ;-
4184 021206 000000          62$:     .WORD    0          ;STORAGE FOR THE LAST READ WORD.

```

```

4186 .SBTTL GLOBAL SUBROUTINE - MSLOOP -
4187 ;*****
4188 ;* - TEST LOOP SUBROUTINE -
4189 ;* THIS SUBROUTINE IS A GENERAL PURPOSE TEST LOOP SUBROUTINE. IT IS USED
4190 ;* TO VERIFY THAT A CERTAIN ACTION OCCURS BEFORE A TIME-OUT PERIOD. THE
4191 ;* CALLING ROUTINE PASSES IN WHICH BITS SHOULD BE SET AND CLEARED FOR THE
4192 ;* DESIRED CONDITION AND THE TIME-OUT VALUE IN MILLI-SECONDS.
4193 ;* THIS ROUTINE CHECKS FOR THE DESIRED CONDITION UPON ENTRANCE INTO THE
4194 ;* ROUTINE AND THEN ONCE EACH MILLI-SECOND THEREAFTER.
4195 ;*
4196 ;* INPUTS: R1 - TIME-OUT VALUE IN MILLI-SECONDS (UP TO 64K MS).
4197 ;* R2 - BIT MAP OF BITS TO TEST (1 INDICATES TO TEST THE BIT).
4198 ;* R3 - DESIRED STATES OF THE INDICATED FIELDS IN R2.
4199 ;* R4 - ADDRESS OF THE WORD TO TEST.
4200 ;* MSLCNT - MILLI SECOND SOFTWARE LOOP COUNT.
4201 ;*
4202 ;* OUTPUTS: CARRY - SUCCESS FLAG (SET IF CONDITION IS MET BEFORE TIME-OUT).
4203 ;*
4204 ;* CALLING SEQUENCE: JSR PC,MSLOOP
4205 ;*
4206 ;* COMMENTS: THIS ROUTINE WORKS WITH OR WITHOUT A HARDWARE CLOCK, BUT THE
4207 ;* CALIBRATION IS ONLY GUARENTEED WHEN A LINE CLOCK IS AVAILABLE
4208 ;* ON THE SYSTEM.
4209 ;* THIS ROUTINE CAN BE USED AS A DELAY ROUTINE, BY SPECIFYING THE
4210 ;* DESIRED DELAY AS THE TIME-OUT AND SPECIFYING A CONDITION TO
4211 ;* LOOK FOR WHICH WILL NOT BE MET DURING THE DELAY.
4212 ;* IF A TIME-OUT VALUE OF 0 IS SPECIFIED, THIS ROUTINE CHECKS FOR
4213 ;* THE DESIRED CONDITION BEFORE RETURNING. IT INDICATES SUCCESS
4214 ;* IF THE CONDITION IS MET, FAILURE OTHERWISE.
4215 ;*
4216 ;* SUBORDINATE ROUTINES CALLED: MSLGET.
4217 ;*****
4218
4219 021210 MSLOOP:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
021210 004567 164110 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4220
4221 ;*
4222 ;* CALLING THE MSLGET ROUTINE FROM THE MSLOOP ROUTINE ISOLATES THE CALLER OF
4223 ;* MSLOOP FROM THE RETURNED TEST WORD AND REMAINING TIME-OUT VALUES.
4224 ;*
4225 021214 004767 177654 JSR PC,MSLGET ;CALL THE MULTI-PURPOSE MS LOOP AND SEARCH RTN.
4226
4227 021220 60$: PASS ;RESTORE GPRS,
021220 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4228 021222 000207 RTS PC ;CARRY - SET IF SUCCESS, CLEAR IF TIME-OUT,

```

```

4230 .SBTTL GLOBAL SUBROUTINE - MSSRPT -
4231 ;* *****
4232 ;* - MODEM STATUS SIGNAL REPORT ROUTINE -
4233 ;* THIS SUBROUTINE IS USED TO REPORT THE STATES OF THE MODEM STATUS
4234 ;* SIGNALS FOR ALL ACTIVE LINES.
4235 ;*
4236 ;* INPUTS: ACTLNS - BIT MAP OF ACTIVE LINES.
4237 ;* CSRA - CONTAINS ADDRESS OF THE DUT CSR.
4238 ;* EF9101 - LABEL AT FORMAT STATEMENT FOR BLANK LINE.
4239 ;* IESTAT - CONTAINS STATES OF THE DUT INTERRUPT ENABLE BITS.
4240 ;* STATA - CONTAINS ADDRESS OF THE DUT STAT REGISTER.
4241 ;* NUMLNS - EQUATED TO THE NUMBER OF LINES ON THE DEVICE.
4242 ;*
4243 ;* OUTPUTS: DUT CSR IND.ADR.REG FIELD - CONTENTS DESTROYED.
4244 ;* REPORT MESSAGES ARE PRINTED ON THE OPERATOR'S CONSOLE.
4245 ;*
4246 ;* CALLING SEQUENCE: JSR PL,MSSRPT
4247 ;*
4248 ;* COMMENTS:
4249 ;*
4250 ;* SUBORDINATE ROUTINES CALLED: NONE.
4251 ;* - *****
4252
4253 021224 MSSRPT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
021224 004567 164074 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4254
4255 ; PRINT THE BASIC MODEM STATUS MESSAGE.
4256 ; "MODEM STATUS SIGNAL REPORT:"
4257 ; -
4258 021230 PRINTF #MSFMT1
021230 012746 007566 MOV #MSFMT1,-(SP)
021234 012746 000001 MOV #1,-(SP)
021240 010600 MOV SP,R0
021242 104417 TRAP C$PNTF
021244 062706 000004 ADD #4,SP
4259
4260 021250 CLR R1 ;START WITH LINE 0.
4261 021252 MOV #1,R2
4262 021256 MOV CSRA,R3 ;GET THE CSR ADDRESS.
4263 021262 MOV IESTAT,R4 ;GET THE STATES OF THE INTERRUPT ENABLE BITS.
4264 021266 MOV ACTLNS,R5 ;GET THE ACTIVE LINES BIT MAP.
4265
4266 021272 2$: BIT R2,R5 ;TEST LINE BIT IN ACTIVE LINES BIT MAP.
4267 021274 BEQ 4$ ;LINE ACTIVE? NO, SKIP REPORT FOR LINE.
4268
4269 021276 MOV R4,R0 ;SET UP DUT CSR IND.ADR.REG FIELD
4270 021300 BIS R1,R0 ; LEAVING THE INTERRUPT ENABLE
4271 021302 MOV R0,(R3) ; BITS IN THE SPECIFIED STATE.
4272 021304 MOV #FSLSA,R0 ;READ THE DUT STATUS REG FOR THIS LINE.
4273 021310 SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
021310 004567 164010 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4274 021314 CLR R2 ;CLEAR THE SIGNAL STATUS INDICATORS.
4275 021316 CLR R3
4276 021320 CLR R4
4277 021322 CLR R5
4278 021324 ASL R0 ;SHIFT DSR INTO CARRY.
4279 021326 ROL R2 ; THEN ROTATE INTO INDICATOR.

```

```

4280 021330 006300      ASL    R0      ;SHIFT BLANK SLOT INTO CARRY,
4281 021332 006300      ASL    R0      ;  SHIFT RI INTO CARRY,
4282 021334 006103      ROL    R3      ;  THEN ROTATE INTO INDICATOR.
4283 021336 006300      ASL    R0      ;SHIFT DCD INTO CARRY,
4284 021340 006104      ROL    R4      ;  THEN ROTATE INTO INDICATOR.
4285 021342 006300      ASL    R0      ;SHIFT CTS INTO CARRY,
4286 021344 006105      ROL    R5      ;  THEN ROTATE INTO INDICATOR.
4287
4288                    ; PRINT THE STATUS FOR THIS LINE.
4289                    ; "LINE #N: DSR=N, RI=N, DCD=N, CTS=N"
4290                    ;
4291                    ; PRINTF #MSFMT2,R1,R2,R3,R4,R5
021346                    MOV    R5,-(SP)
021346 010546                    MOV    R4,-(SP)
021350 010446                    MOV    R3,-(SP)
021352 010346                    MOV    R2,-(SP)
021354 010246                    MOV    R1,-(SP)
021356 010146                    MOV    #MSFMT2,-(SP)
021360 012746 007626                    MOV    #6,-(SP)
021364 012746 000006                    MOV    SP,R0
021370 010600                    TRAP  C$PNTF
021372 104417                    ADD    #16,SP
021374 062706 000016
4292 021400      PASS                    ;RESTORE ALL THE GPRS.
021400 004736      JSR    PC,@(SP),      ;RETURN TO PREGOS SUBRT.
4293
4294 021402 006302      4$:  ASL    R2      ;SHIFT LINE BIT MAP TO NEXT LINE.
4295 021404 005201      INC    R1      ;INCREMENT THE LINE COUNTER.
4296 021406 020127 000020      CMP    R1,#NUMLNS ;CMP LINE COUNTER WITH # OF LINES ON DEVICE.
4297 021412 002727      BLT    2$      ;ALL LINES DONE? NO, LOOP TO DO NEXT LINE.
4298
4299 021414      PRINTF #EF9101      ;PRINT A BLANK LINE.
021414 012746 007306                    MOV    #EF9101,-(SP)
021420 012746 000001                    MOV    #1,-(SP)
021424 010600                    MOV    SP,R0
021426 104417                    TRAP  C$PNTF
021430 062706 000004                    ADD    #4,SP
4300
4301 021434      60$:  PASS                    ;RESTORE GPRS.
021434 004736      JSR    PC,@(SP),      ;RETURN TO PREGOS SUBRT.
4302 021436 000207      RTS    PC

```



```

4304 .SBTTL GLOBAL SUBROUTINE - MUL16U -
4305 ;* *****
4306 ;* - 16 BIT UNSIGNED MULTIPLY ROUTINE -
4307 ;* THIS ROUTINE MULTIPLIES 2 16 BIT UNSIGNED NUMBERS AND RETURNS A 16 BIT
4308 ;* UNSIGNED RESULT. THE MULTIPLICATION IS PERFORMED BY ITERATIVE
4309 ;* ADDITION OF ONE NUMBER TO A SUM WHILE DECREMENTING THE OTHER NUMBER
4310 ;* TO ZERO. IF OVERFLOW OCCURS (177777 TO 0) THE PRODUCT IS INVALID.
4311 ;*
4312 ;* INPUTS: R1 - MULTIPLICAND (16 BIT UNSIGNED).
4313 ;* R2 - MULTIPLIER (16 BIT UNSIGNED).
4314 ;*
4315 ;* OUTPUTS: R1 - PRODUCT (16 BIT UNSIGNED), -1 IF OVERFLOW.
4316 ;* CARRY - SET IF SUCCESS (NO OVERFLOW), CLEAR OTHERWISE.
4317 ;*
4318 ;* CALLING SEQUENCE: JSR PC,MUL16U
4319 ;*
4320 ;* COMMENTS: NOTE: FOR MINIMUM EXECUTION TIME R2 SHOULD CONTAIN THE
4321 ;* SMALLER OF THE 2 ARGUMENTS.
4322 ;*
4323 ;* SUBORDINATE ROUTINES CALLED: NONE.
4324 ;* - *****
4325
4326 021440 MUL16U:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
021440 004567 163660 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4327 021444 C05003 ;CLEAR THE PRODUCT.
4328 021446 005702 ;CHECK THE MULTIPLIER.
4329 021450 001003 BNE 2$ ;GO TO DO MULTIPLICATION IF NOT ZERO.
4330 021452 005001 CLR R1 ;RETURN A PRODUCT OF ZERO.
4331 021454 000261 SEC ;INDICATE SUCCESS.
4332 021456 000412 BR 60$ ;EXIT THE ROUTINE.
4333
4334 021460 060103 2$: ADD R1,R3 ;ADD THE MULTIPLICAND TO THE PRODUCT.
4335 021462 103405 BCS 50$ ;EXIT WITH OVERFLOW IF ONE OCCURRED.
4336 021464 005302 DEC R2 ;DECREMENT THE MULTIPLIER.
4337 021466 001374 BNE 2$ ;LOOP IF MULTIPLIER NOT ZERO.
4338 021470 010301 MOV R3,R1 ;PREPARE TO PASS OUT THE PRODUCT.
4339 021472 000261 SEC ;INDICATE SUCCESS.
4340 021474 000403 BR 60$ ;EXIT WITH SUCCESS.
4341
4342 021476 012701 177777 50$: MOV 0-1,R1 ;FORCE PRODUCT TO MAX VALUE, WE OVERFLOWED.
4343 021502 000241 CLC ;INDICATE FAILURE.
4344
4345 021504 60$: PASS R1 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
021504 010166 000004 MOV R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
021510 004736 JSR PC,0(SP) ;RETURN TO PREG05 SUBRT.
4346 ; R1 - PRODUCT (16 BIT UNSIGNED),
4347 021512 000207 RTS PC ; CARRY - SET IF SUCCESS (NO OVERFLOW).

```

4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389 021514
021514 004567 163604
4390 021520 010305
4391 021522 052705 177400
4392 021526 005067 000270
4393
4394
4395
4396
4397 021532 004767 175412
4398 021536 103052
4399
4400
4401
4402 021540 010304
4403 021542 006304
4404 021544 006304

```

.SBTTL GLOBAL SUBROUTINE - NEWCHR -
*****
; * - NEW CHARACTER HANDLING ROUTINE -
; * THIS SUBROUTINE HANDLES A NEW CHARACTER WHICH HAS BEEN READ FROM
; * THE OUT. THE COUNTERS AND POINTERS WHICH ARE INVOLVED WITH THE
; * CHARACTER ARE UPDATED. THE CHARACTER IS CHECKED FOR ERRORS AND
; * ANY ERRORS WHICH ARE FOUND ARE REPORTED.
; *
; * INPUTS: R2 - THE READ CHARACTER INCLUDING ERROR FLAGS AND LINE NUMBER.
; * R3 - MASK OF THE INACTIVES BITS IN A TX OR RX CHAR BYTE.
; * ACTLNS - BIT MAP OF ACTIVE OUT LINES.
; * DPRSQB - LABEL AT DATA PATTERN RESYNC QUEUES TABLE BASE.
; * TXRXLB - BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
; * BITTBL - TABLE OF WORDS WITH BITS SET FOR USE IN FORMING MAPS.
; * ERSMRF - "PRINT ERROR SUMMARY FOR LINE" FLAGS.
; * ERRTBL - ERROR INFORMATION (ERRNBR, ERRMSG, ERRTP).
; * ERCNTB - BASE OF THE RX CHARACTER ERROR COUNTERS TABLE.
; * NDERPT - CONTAINS NUMBER OF CHAR ERRORS TO REPORT ON A LINE.
; * INPUTS TO SUBROUTINES: CHCNTB, DPENDB, DPLEN, DPRSQE, EXCNTB, RXCNTB,
; * RXPTRB, ERRNBR, ERRMSG, ERRTP.
; *
; * OUTPUTS: ERRLK - CONTENTS DESTROYED.
; * FOLLOWING VARIABLES UPDATED FOR LINE ON WHICH CHAR WAS RECEIVED:
; * DPRSQ - DATA PATTERN RESYNC QUE OF RECEIVED CHARACTERS.
; * ERCNT - COUNT OF THE NUMBER OF CHARACTER ERRORS ON LINE.
; * ERSMRF - UPDATED "PRINT ERROR SUMMARY FOR LINE" FLAGS.
; * EXCNT - COUNT OF THE NUMBER OF EXTRA CHARS RECEIVED ON LINE.
; * RXCNT - COUNT OF THE NUMBER OF CHARACTERS RECEIVED ON LINE.
; * RXPTR - UPDATED TO POINT TO THE NEXT EXPECTED CHAR ON LINE.
; *
; * CALLING SEQUENCE: JSR PC,NEWCHR
; *
; * COMMENTS: THIS ROUTINE CAN REPORT ERRORS WITH NUMBERS INITIAL ERRNBR
; * AND INITIAL ERRNBR + 1. ERRNBR IS RESTORED TO ITS INITIAL
; * VALUE BEFORE THIS ROUTINE RETURNS.
; *
; * SUBROUTINES CALLED: CKCHR,CKINAC,TXROFF,TXRON.
; * INDIRECT SUBROUTINES: CHKEXT,CHKLOS,ER9002,ER9003,UPDCHR.
; * *****
NEWCHR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R3,R5 ;GET THE BIT MAP OF INACTIVE DATA BYTE BITS.
BIS #177400,R5 ;ALL UPPER BITS OF EXPECTED DATA ARE INACTIVE.
CLR 70$ ;CLEAR THE "ERROR FOUND" FLAG.
; *
; * IF THE NEW CHARACTER IS VALID ON AN INACTIVE LINE, GO REPORT ERROR.
; * ROUTINE USED ALSO EXTRACTS LINE NUMBER FROM THE NEW CHARACTER.
; *
; * JSR PC,CKINAC ;CHECK FOR CHAR ON INACTIVE LINE.
; * BCC 4$ ;GO REPORT ERROR IF ON INACTIVE LINE.
; *
; * PUSH THE NEW CHARACTER ON THE RESYNC QUE FOR THIS LINE.
; *
; * MOV R3,R4 ;CALCULATE BASE ADDRESS OF THE
; * ASL R4 ; DATA PATTERN RESYNCH QUEUE
; * ASL R4 ; (QUEUE IS 4 WORDS LONG) FOR

```

```

4405 021546 062704 004642      ADD    #DPRSQB,R4      ; THIS LINE.
4406 021552 010401      MOV    R4,R1          ;GET THE BASE OF THE QUEUE.
4407 021554 016121 000002      MOV    2(R1),(R1)+    ;MOVE FROM CHR1 SLOT TO CHR0 SLOT.
4408 021560 016121 000002      MOV    2(R1),(R1)+    ;MOVE FROM CHR2 SLOT TO CHR1 SLOT.
4409 021564 010211      MOV    R2,(R1)        ;PUT NEW CHAR INTO CHR2 SLOT.
4410
4411      ;+
4412      ; CHECK THE DATA VALID FOR THE CHARACTER AT THE BOTTON OF THE QUEUE.
4413      ; IF DATA.VALID IS CLEAR, EXIT THE ROUTINE --NOTHING TO ANALYZE.
4414 021566 011402      MOV    (R4),R2        ;GET CHR0 VALUE, SET FLAGS.
4415 021570 100112      BPL   50$            ;EXIT ROUTINE IF DATA.VALID IS CLEAR.
4416
4417      ;+
4418      ; TEST FOR ANY OF THE ERROR BITS SET IN CHR0.
4419 021572 032702 070000      BIT    #70000,R2     ;TEST FOR ANY CHR0 ERROR BITS SET.
4420 021576 001427      BEQ   2$            ;SKIP THIS ERROR IF NO ERROR BITS SET.
4421
4422      ;+
4423      ; WE HAVE AT LEAST ONE ERROR FLAG SET ON THE RECEIVED CHAR.
4424      ; REPORT DATA ERROR FLAG ERROR IF NOT IN SUMMARY MODE.
4425 021600 005367 000216      DEC    70$          ;SET THE "ERROR FOUND" FLAG.
4426 021604 016300 005234      MOV    TXRXLB(R3),R0 ;GET THE TX LINE OFFSET FOR THIS RX LINE.
4427 021610 036067 002364 160662      BIT    BITTBL(R0),ERSMRF ;CHECK THE ERROR SUMMARY FLAG FOR TX LINE.
4428 021616 001017      BNE   2$            ;IF ERROR SUMMARY FLAG SET, SKIP NEXT REPORT.
4429 021620 012767 014676 163474      MOV    #ER9003,ERRBLK ;SELECT THE ER9003 ERROR REPORT ROUTINE.
4430 021626 004767 004326      JSR   PC,TXROFF     ;TURN OFF TX AND RX DURING ERROR REPORTING.
4431 021632      ERROR              ;
4432 021634 012767 000001 160362      MOV    #1,FERROR    ;INDICATE AN ERROR HAS BEEN FOUND.
4433 021642 032767 000100 160312      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
4434 021650 001462      BEQ   60$          ;EXIT IF IT HASN'T.
4435
4436 021652 004767 004342      JSR   PC,TXRON      ;TURN TX AND RX BACK ON.
4437
4438      ;+
4439      ; CHECK THE CHARACTER AT THE BOTTOM OF THE RESYNC QUE FOR DATA ERRORS.
4440 021656 004767 174636      2$: JSR   PC,CKCHR     ;CHECK THE CHR0 CHAR FOR ERRORS.
4441 021662 103433      BCS   6$            ;SKIP ERROR REPORT IF CHR0 IS CORRECT.
4442
4443      ;+
4444      ; WE HAVE SOME SORT OF DATA ERROR SO REPORT IT (UNLESS IN SUMMARY REPORT MODE).
4445 021664 005367 000132      4$: DEC    70$          ;SET THE "ERROR FOUND" FLAG.
4446 021670 016300 005234      MOV    TXRXLB(R3),R0 ;GET THE TX LINE OFFSET FOR THIS RX LINE.
4447 021674 036067 002364 160576      BIT    BITTBL(R0),ERSMRF ;CHECK THE ERROR SUMMARY FLAG FOR THIS LINE.
4448 021702 001023      BNE   6$            ;SKIP ERROR REPORT IF ERROR SUMMARY FLAG SET.
4449 021704 012767 014520 163410      MOV    #ER9002,ERRBLK ;SELECT THE ER9002 ERROR REPORT ROUTINE.
4450 021712 005267 163400      INC    ERRNBR        ;SELECT INITIAL ERRNBR + 1.
4451 021716 004767 004236      JSR   PC,TXROFF     ;TURN OFF TX AND RX DURING ERROR REPORTING.
4452 021722      ERROR              ;
4453 021724 012767 000001 160272      MOV    #1,FERROR    ;INDICATE AN ERROR HAS BEEN FOUND.
4454 021732 032767 000100 160222      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
4455 021740 001426      BEQ   60$          ;EXIT IF IT HASN'T.
4456
4457 021742 004767 004252      JSR   PC,TXRON      ;TURN TX AND RX BACK ON.
4458 021746 005367 163344      DEC    ERRNBR        ;RESTORE INITIAL ERRNBR.
4459

```

```

4460 ; COUNT A CHARACTER ERROR IF ONE OCCURRED.
4461 ; UPDATE THE "REPORT ERROR SUMMARY" FLAG FOR LINE BASED ON ERROR COUNT.
4462 ;
4463 021752 005767 000044 6$: TST 70$ ;CHECK THE "ERROR FOUND" FLAG.
4464 021756 001417 BEQ 60$ ;SKIP COUNTING AN ERROR IF FLAG IS CLEAR.
4465 021760 005263 003302 INC ERCNTB(R3) ;INCREMENT THE ERROR COUNTER FOR THIS LINE.
4466 021764 001002 BNE 8$ ;SKIP SETTING COUNTER TO MAX IF NO OVERFLOW.
4467 021766 005363 003302 DEC ERCNTB(R3) ;RESET THE ERROR COUNTER TO -1 (MAX VALUE).
4468 021772 005767 160166 8$: TST NDERPT ;DISABLE ERROR SUMMARY FUNCTION IF
4469 021776 001407 BEQ 60$ ; NUMBER OF DATA ERRORS TO REPORT IS 0.
4470 022000 026367 003302 160156 CMP ERCNTB(R3),NDERPT ;COMPARE ERROR COUNT WITH # OF ERR'S TO RPT.
4471 022006 103403 BLO 60$ ;SKIP SETTING OF SUMMARY FLAG IF NOT TOO MANY.
4472 022010 056367 002364 160462 BIS BITTBL(R3),ERSMRF ;SET "PRINT ERROR SUMMARY" FLAG FOR LINE.
4473
4474 022016 60$: PASS ;RESTORE GPRS.
022016 004736 JSR PC,0(SP)+ ;RETURN TO PREG05 SUBRT.
4475 022020 000207 RTS PC
4476
4477 022022 000000 70$: .WORD 0 ;LOCAL STORAGE FOR ERROR OCCURRED FLAG.

```

```

4479 .SBTTL GLOBAL SUBROUTINE - OOPS -
4480 ;* *****
4481 ;* - PROGRAM ABORT SUBROUTINE -
4482 ;* THIS SUBROUTINE IS USED TO ABORT THE PROGRAM WHEN A FATAL ERROR IS
4483 ;* DETECTED IN THE PROGRAM OR THE HOST SYSTEM HARDWARE. AN ERROR MESSAGE
4484 ;* IS PRINTED GIVING SOME INFORMATION ABOUT THE NATURE OF THE ABORT.
4485 ;*
4486 ;* INPUTS: R1 - ERROR CODE GIVING REASON FOR ABORT.
4487 ;*
4488 ;* OUTPUTS: AN ERROR MESSAGE IS PRINTED.
4489 ;* A LIST OF RETURN PC VALUES FOR ALL SUBROUTINE CALLS IS PRINTED.
4490 ;*
4491 ;* CALLING SEQUENCE: JSR PC,OOPS
4492 ;*
4493 ;* COMMENTS:
4494 ;*
4495 ;* SUBORDINATE ROUTINES CALLED: NONE.
4496 ;* - *****
4497
4498 022024 OOPS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
022024 004567 163274 ; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4499 ; REPORT "HOST COMPUTER HARDWARE OR SOFTWARE BUG ENCOUNTERED." ERROR.
4500 022030 ERRSF 101,EM0101
022030 104454 TRAP C$ERSF
022032 000145 .WORD 101
022034 022070 .WORD EM0101
022036 000000 .WORD 0
4501 ; REPORT "PROGRAM HUNG, WAITING FOR A CONTROL-C."
4502 022040 PRINTF #EM0102
022040 012746 022154 MOV #EM0102,-(SP)
022044 012746 000001 MOV #1,-(SP)
022050 010600 MOV SP,R0
022052 104417 TRAP C$PNTF
022054 062706 000004 ADD #4,SP
4503 022060 2$: BREAK ;LOOK FOR OPERATOR CONTROL -C INPUT.
022060 104422 TRAP C$BRK
4504 022062 000776 BR 2$ ;INFINITE LOOP.
4505 022064 60$: PASS ;DON'T NEED THIS, BUT SOMEBODY MAY CHANGE THIS
022064 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4506 022066 000207 RTS PC ; ROUTINE IN THE FUTURE, SO BE CONSISTANT.
4507
4508 022070 110 117 123 EM0101:: .ASCIZ /HOST COMPUTER HARDWARE OR SOFTWARE BUG ENCOUNTERED./
022073 124 040 103
022076 117 115 120
022101 125 124 105
022104 122 040 110
022107 101 122 104
022112 127 101 122
022115 105 040 117
022120 122 040 123
022123 117 106 124
022126 127 101 122
022131 105 040 102
022134 125 107 040
022137 105 116 103
022142 117 125 116
022145 124 105 122

```

B10

	022150	105	104	056	
	022153	000			
4509	022154	045	116	045	EM0102:: ,ASCIZ /N*APROGRAM HUNG, WAITING FOR A CONTROL -C, <*****N*/
	022157	101	120	122	
	022162	117	107	122	
	022165	101	115	040	
	022170	110	125	116	
	022173	107	054	040	
	022176	127	101	111	
	022201	124	111	116	
	022204	107	040	106	
	022207	117	122	040	
	022212	101	040	103	
	022215	117	116	124	
	022220	122	117	114	
	022223	055	103	056	
	022226	040	074	052	
	022231	052	052	052	
	022234	052	052	052	
	022237	052	052	052	
	022242	052	052	052	
	022245	045	116	045	
4510	022250	116	000		

.EVEN

4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565

022252
022252 004567 163046
022256 016704 163034
022262 005005

022264 012767 014162 163030
022272 032702 020000
022276 001002
022300 052705 000002

022304 032702 010000
022310 001402
022312 052705 000014
022316 005705
022320 001412
022322 036367 002364 160150
022330 001004

022332
022332 104460

022334 012767 000001 157662

022342 005263 003302
022346 010467 162744
022352
022352 004736

```
.SBTTL GLOBAL SUBROUTINE - PRFRME -
;*****
; - PROCESS FRAMING ERRORS -
; THIS SUBROUTINE IS USED IN THE FRAMING ERROR BIT TEST, TO VERIFY THAT
; ALL RECEIVED CHARACTERS HAVE THEIR FRAMING ERROR BIT SET AND PARITY
; ERROR BIT CLEAR.
;
; INPUTS: R2 - CONTAINS THE CHARACTER READ FROM THE FIFO.
; ERRNBR - ERROR NUMBER OF ERRORS IN THIS ROUTINE.
; ERSMAF - "REPORT ERROR SUMMARY FOR LINE" FLAGS
;
; OUTPUTS: ERBLK - THE CONTENTS OF THIS WORD ARE DESTROYED.
; ERCNTB - THE ERROR COUNT FOR THIS LINE IS UPDATED.
; MESSAGES MAY BE PRINTED AT THE OPERATORS CONSOLE.
;
; CALLING SEQUENCE: JSR PC,PRFRME
;
; COMMENTS: THIS ROUTINE REPORTS ERRORS WITH INITIAL NUMBER.
; ERRNBR IS RESTORED TO ITS INITIAL VALUE BEFORE THIS SUBROUTINE
; RETURNS.
;
; SUBORDINATE ROUTINES CALLED: ER6201.
;*****
PRFRME::SAVE
;SAVE CONTENTS OF GPRS R0 THRU R5.
;CALL REGISTER SAVE SUBRT.
;SAVE THE CONTENTS OF THE INITIAL ERROR NUMBER.
;CLEAR ERROR/MESSAGE FLAGS.
;
; TEST FRAMING AND PARITY ERROR BITS IN TURN. REPORT ANY ERRORS FOUND, IE.
; FRAMING ERROR BIT CLEAR, OR PARITY ERROR BIT SET.
;
; SET UP THE ADDRESS OF THE ERROR ROUTINE.
; CHECK ON STATE OF THE FRAMING ERROR BIT.
; BRANCH IF FRAMING ERROR BIT SET.
; SET REPORT FRAMING ERROR FLAG.
;
; CHECK ON THE STATE OF THE PARITY ERROR BIT.
; BRANCH IF PARITY ERROR BIT CLEAR.
; SET REPORT "PARITY ERROR SET" FLAGS.
; CHECK IF ANY ERROR FLAGS SET.
; EXIT IF ALL FLAGS CLEAR.
; CHECK THE ERROR SUMMARY FLAG FOR THIS LINE.
; SKIP ERROR REPORT IF ERROR SUMMARY FLAG SET.
;
; REPORT ERROR "CHARACTER RECEIVED WITH PARITY/FRAMING ERROR BIT SET".
; >>>> ERROR <<<<<.
; TRAP C$ERROR
;
; INDICATE AN ERROR HAS BEEN DETECTED.
;
; INCREMENT ERROR COUNT FOR THIS LINE.
; RESTORE ERROR NUMBER.
; RESTORE GPRS.
; RETURN TO PREG05 SUBRT.
;
; MOV ERRNBR,R4
; CLR R5
; MOV #ER6201,ERRBLK
; BIT #BIT13,R2
; BNE 6$
; BIS #BIT1,R5
; 6$: BIT #BIT12,R2
; BEQ 8$
; BIS #14,R5
; 8$: TST R5
; BEQ 60$
; BIT BITTBL(R3),ERSMAF
; BNE 10$
; INC ERCNTB(R3)
; MOV R4,ERRNBR
; PASS
; JSR PC,(SP)+
```

DHU-11 FUNC TST PART4
GLOBAL SUBROUTINE

MACRO M1200 15-MAR-84 09:48 PAGE 82-1
- PRFRME -

D10

SEQ 120

4566 022354 000207

RTS PC


```

4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596 022356
      022356 004567 162742
4597 022362 016746 162730
4598 022366 005005
4599
4600
4601
4602
4603
4604 022370 012767 014162 162724
4605 022376 032702 010000
4606 022402 001002
4607 022404 052705 000010
4608 022410 032702 020000
4609 022414 001402
4610 022416 052705 000003
4611 022422 005705
4612 022424 001414
4613 022426 036367 002364 160044
4614 022434 001035
4615
4616 022436
      022436 104460
4617
4618 022440 012767 000001 157556
4619 022446 032767 000100 157506
4620 022454 001440
4621
4622

```

```

,SBTTL GLOBAL SUBROUTINE - PRPARE -
;+ *****
;+ - PROCESS PARITY ERRORS -
;+ THIS SUBROUTINE IS USED IN THE PARITY ERROR TEST, TO VERIFY THAT
;+ ALL RECEIVED CHARACTERS HAVE THEIR PARITY ERROR BIT SET AND FRAMMING
;+ ERROR BIT CLEAR.
;+
;+ INPUTS: R2 - CONTAINS THE CHARACTER READ FROM THE FIFO.
;+ R3 - CONTAINS 2 * LINE NUMBER OF THE READ CHAR.
;+ ERRNBR - ERROR NUMBER OF ERRORS IN THIS ROUTINE.
;+ ERSMRF - "REPORT ERROR SUMMARY FOR LINE" FLAGS.
;+ FERROR - "AT LEAST ONE ERROR FOUND" FLAG.
;+
;+ OUTPUTS: ERRBLK - THE CONTENTS OF THIS WORD ARE DESTROYED.
;+ ERCNTB - THE ERROR COUNT FOR THIS LINE IS UPDATED.
;+ MESSAGES MAY BE PRINTED AT THE OPERATORS CONSOLE.
;+
;+ CALLING SEQUENCE: JSR PC,PRPARE
;+
;+ COMMENTS: THIS ROUTINE REPORTS ERRORS WITH INITIAL ERRNBR THRU ERRNBR+1.
;+ ERRNBR IS RESTORED TO ITS INITIAL VALUE BEFORE THIS SUBROUTINE
;+ RETURNS.
;+ THE CONTENTS OF THE ERRBLK ARE DESTROYED.
;+
;+ SUBORDINATE ROUTINES CALLED: ER9002,ER6201.
;+ *****
PRPARE::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
      JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
      MOV ERRNBR,-(SP) ;SAVE THE CONTENTS OF THE INITIAL ERROR NUMBER.
      CLR R5 ;CLEAR ERROR/MESSAGE FLAGS.
;+
;+ TEST FRAMMING AND PARITY ERROR BITS IN TURN. REPORT ANY ERRORS FOUND, IE.
;+ PARITY ERROR BIT CLEAR, OR FRAMMING ERROR BIT SET.
;+
;+ MOV #ER6201,ERRBLK ;SET UP THE ADDRESS OF THE ERROR ROUTINE.
;+ BIT #BIT12,R2 ;CHECK ON STATE OF THE PARITY ERROR BIT.
;+ BNE 6$ ;BRANCH IF PARITY ERROR BIT SET.
;+ BIS #BIT3,R5 ;SET REPORT PARITY ERROR FLAG.
;+ BIT #BIT13,R2 ;CHECK ON THE STATE OF THE FRAMMING ERROR BIT.
;+ BEQ 8$ ;BRANCH IF FRAMMING ERROR BIT CLEAR.
;+ BIS #3,R5 ;SET REPORT "FRAMMING ERROR SET" FLAGS.
;+ TST R5 ;CHECK IF ANY ERROR FLAGS SET.
;+ BEQ 12$ ;BRANCH TO MAKE DATA CHECK IF ALL FLAGS CLEAR.
;+ BIT BITTBL(R3),ERSMRF ;CHECK THE ERROR SUMMARY FLAG FOR THIS LINE.
;+ BNE 14$ ;SKIP ALL ERROR REP IF IN ERROR SUMMARY MODE.
;+ REPORT ERROR "CHAR RECEIVED WITH PARITY/FRAMMING ERROR BIT SET/CLEAR".
;+ ERROR >>>> ERROR <<<<<.
;+ TRAP C$ERROR
;+
;+ MOV #1,FERROR ;INDICATE AN ERROR HAS BEEN FOUND.
;+ BIT #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
;+ BEQ 18$ ;EXIT IF IT HASN'T.
;+
;+

```

```

4623                                     ; COMPARE ACTUAL DATA WITH EXPECTED DATA TO CHECK FOR MULTIPLE ERRORS.
4624                                     ;
4625 022456 005267 162634                12$: INC   ERRNBR           ; INCREMENT ERROR NUMBER.
4626 022462 016304 003402                MOV   RXPTRB(R3),R4 ; GET THE POINTER TO THE EXPECTED DATA.
4627 022466 111404                       MOVB  (R4),R4        ; GET THE EXPECTED DATA.
4628 022470 042704 177400                BIC   #177400,R4    ; CLEAR THE UPPER BYTE.
4629 022474 120204                       CMPB  R2,R4         ; COMPARE ACTUAL AND EXPECTED DATA.
4630 022476 001427                       BEQ   18$          ; SKIP ERROR REPORT IF DATA CORRECT.
4631 022500 042704 100000                BIC   #BIT15,R4    ; CLEAR "NONE" EXPECTED MESSAGE FLAG.
4632 022504 036367 002364 157766        BIT   BITT0L(R3),ERSMRF ; CHECK THE ERROR SUMMARY FLAG FOR THIS LINE.
4633 022512 001017                       BNE   16$          ; SKIP ERROR REPORT IF ERROR SUMMARY FLAG SET.
4634 022514 036367 002364 157762        BIT   BITT0L(R3),RXDNF ; CHECK FOR RECEPTION COMPLETE ON THIS LINE.
4635 022522 001402                       BEQ   14$          ; SKIP SETTING NONE EXPECTED FLAG.
4636 022524 052704 100000                BIS   #BIT15,R4    ; SET "NONE" EXPECTED MESSAGE FLAG.
4637 022530 012701 011371                14$: MOV   #EM9008,R1 ; SELECT ERROR MESSAGE TO BE REPORTED.
4638 022534 012767 014520 162560        MOV   #ER9002,ERRBLK ; SELECT ERROR REPORTING ROUTINE.
4639                                     ; REPORT ERROR "RECEIVE CHARACTER MISCOMPARE"
4640 022542                                ERROR
4641 022542 104460                                TRAP   C$ERROR
4641 022544 012767 000001 157452        MOV   #1,FERROR   ; INDICATE AN ERROR HAS BEEN FOUND.
4642
4643
4644 022552 005263 003302                16$: INC   ERCNTB(R3) ; INCREMENT ERROR COUNT FOR THIS LINE.
4645 022556 012667 162534                18$: MOV   (SP)+,ERRNBR ; RESTORE ERROR NUMBER.
4646
4647 022562                                60$: PASS
4647 022562 004736                                ; RESTORE GPRS.
4648 022564 000207                                RTS   PC          JSR   PC,@(SP)+ ; RETURN TO PREG05 SUBRT.

```

4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683

022566		
022566	004567	162532
022572	016701	157402
022576	016702	157402
022602	042703	177760
022606	056703	157422
022612	010311	
022614	011204	
022616		
022616	010446	
022620	012746	012075
022624	012746	007206
022630	012746	000003
022634	010600	
022636	104415	
022640	062706	000010
022644		
022644	004736	
022646	000207	

```

.SBTTL GLOBAL SUBROUTINE - PRTLPR -
; * *****
; * -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: R3 - CONTAINS THE NUMBER OF THE LINE YOU WISH TO EXAMINE.
; * CSRA - CONTAINS THE ADDRESS OF THE DUT'S CSR.
; * IESTAT - CONTAINS THE CURRENT STATUS OF THE TX AND RX INTERRUPT
; * ENABLE BITS IN THE DUT'S CSR.
; * LPRA - CONTAINS THE ADDRESS OF THE DUT'S LPR REGISTER.
; *
; * OUTPUTS: AN EXTENDED INFORMATION MESSAGE IS PRINTED ON THE OPERATORS
; * CONSOLE.
; *
; * CALLING SEQUENCE: JSR PC,PRTLPR
; *
; * COMMENTS: THIS ROUTINE CHANGES THE INDIRECT ADDRESS FIELD OF THE DEVICE
; * UNDER TEST'S CSR.
; *
; * SUBORDINATE ROUTINES CALLED: NONE.
; *
; * - - - *****
PRTLPR:;SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV CSRA,R1 ;GET THE CSR ADDRESS.
MOV LPRA,R2 ;GET THE LPR ADDRESS.
BIC #177760,R3 ;CLEAR ANY UNWANTED BITS.
BIS IESTAT,R3 ;SET STATE OF TX AND RX INTERRUPT ENABLE BITS.
MOV R3,(R1) ;SELECT LINE.
MOV (R2),R4 ;GET CONTENTS OF THE LPR.
;PRINT MESSAGE "CONTENTS OF THE LPR:NNNNNN"
PRINTX #EF9019,#EM9026,R4;PRINT OUT MESSAGE ON OPERATORS CONSOLE.
MOV R4,-(SP)
MOV #EM9026,-(SP)
MOV #EF9019,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C#PNTX
ADD #10,SP
60$: PASS ;RESTORE GPRS.
;PC,#(SP). ;RETURN TO PREG05 SUBRT.
RTS PC JSR

```

```

4685 .SBTTL GLOBAL SUBROUTINE - PUFIFO -
4686 ;*****
4687 ;* - PURGE THE FIFO
4688 ;* THIS ROUTINE TRIES TO REMOVE ALL THE CHARACTERS FROM THE FIFO,
4689 ;* ANY BMP CODES THAT ARE FOUND ARE SAVED ON THE BMP CODE QUEUE.
4690 ;*
4691 ;* INPUTS: RBUFA- CONTAINS THE ADDRESS OF THE RECEIVER.
4692 ;*
4693 ;*
4694 ;* OUTPUTS: CARRY BIT - INDICATES THE STATE OF THE FIFO, SET: = PURGED.
4695 ;* BMPCQ - THE CONTENTS OF THE BMP CODE QUEUE MAY BE UPDATED.
4696 ;*
4697 ;* CALLING SEQUENCE: JSR PC,PUFIFO
4698 ;*
4699 ;* COMMENTS:
4700 ;*
4701 ;* SUBORDINATE ROUTINES CALLED: SAVBMP.
4702 ;*****
4703
4704 022650 PUFIFO::SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
022650 004567 162450 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4705 022654 012701 001000 MOV #512,R1 ;SET MAXIMUM TRY COUNT OF 512.
4706 022660 016704 157316 MOV RBUFA,R4 ;GET ADDRESS OF THE RECEIVER BUFFER REGISTER.
4707
4708 022664 011402 2$: MOV (R4),R2 ;GET THE CONTENTS OF THE RECEIVER BUFFER REG.
4709 022666 100016 BPL 6$ ;EXIT IF THE FIFO IS EMPTY, DATA_VALID CLR.
4710 ;+
4711 ; CHECK IF THE READ CHARACTER IS ACTUALLY A BMP CODE.
4712 ; IF IT IS, THEN SAVE IT ON THE BMP CODE QUEUE TO BE REPORTED LATER.
4713 ;-
4714 022670 012700 070000 MOV #70000,R0 ;GENERATE A BIT MAP OF CHAR ERROR BITS
4715 022674 040200 BIC R2,R0 ; WHICH ARE NOT SET FOR CHAR.
4716 022676 001006 BNE 4$ ;THROW CHAR AWAY IF NOT BMP OR SELFTEST CODE.
4717 ;+
4718 ; CHECK IF THE READ DATA IS MODEM STATUS , BMP OR SELFTEST?.
4719 ;-
4720 022700 012700 000301 MOV #301,R0 ; CHECK IF BMP.
4721 022704 040200 BIC R2,R0 ;TRY TO CLEAR BMP FLAGS IN THE READ DATA.
4722 022706 001002 BNE 4$ ;IF IT IS MODEM OR SELFTEST CODE THROW IT AWAY.
4723 022710 004767 001644 JSR PC,SAVBMP ;SAVE BMP CODE ON THE QUEUE.
4724
4725 022714 005301 4$: DEC R1 ;DECREMENT THE TRY COUNT.
4726 022716 001362 BNE 2$ ;LOOP TO TRY AGAIN.
4727 022720 000241 CLC ;CLEAR CARRY,TO INDICATE FIFO NOT PURGED.
4728 022722 000401 BR 60$ ;EXIT WITH CARRY CLEAR.
4729 022724 000261 6$: SEC ;SET CARRY, TO INDICATE FIFO PURGED.
4730
4731 022726 60$: PASS ;RESTORE GPRS,
022726 004736 JSR PC,@(SP); ;RETURN TO PREG05 SUBRT.
4732 ;CARRY BIT, SET INDICATES FIFO PURGED.
4733 022730 000207 RTS PC

```

4735
 4736
 4737
 4738
 4739
 4740
 4741
 4742
 4743
 4744
 4745
 4746
 4747
 4748
 4749
 4750
 4751
 4752
 4753
 4754
 4755
 4756
 4757
 4758
 4759
 4760
 4761
 4762
 4763 022732
 022732 004567 152366
 4764 022736 016746 162354
 4765 022742 012705 001000
 4766
 4767
 4768
 4769
 4770 022746 017702 157230
 4771 022752 100063
 4772
 4773
 4774
 4775 022754 012700 070000
 4776 022760 040200
 4777 022762 001012
 4778
 4779
 4780
 4781
 4782 022764 012767 014420 162330
 4783 022772 012700 000300
 4784 022776 040200
 4785 023000 001003
 4786 023002 004767 001552
 4787 023006 000430
 4788
 4789
 4790

```

.SBTTL GLOBAL SUBROUTINE - PUFIFR -
*****
; - PURGE FIFO REPORT ANY ERRORS FOUND.
; THIS ROUTINE REMOVES ALL DATA FROM THE FIFO. ANY BMP CODES THAT ARE
; FOUND ARE SAVE ON THE QUEUE TO BE REPORTED LATER IN THE BMP REPORT TEST.
; ANY UNEXPECTED DATA (IE ANY NON-STATUS INFORAMTION) THAT ARE FOUND,
; ARE REPORTED AS AN ERROR.
; IF THE FIFO WILL NOT PURGE AFTER 512 ATTEMPTS, THEN THE CURRENT TEST
; THAT CALLED THIS ROUTINE RECEIVES A FAILURE FLAG THAT SHOULD BE USED
; TO ABORT THE TEST.
;
; INPUTS: ERRIBL - ERRTYPE, ERRMSG, ERRNBR ARE SET UP CORRECTLY.
; RBUFA - CONTAINS THE ADDRESS OF THE RECEIVER.
;
; OUTPUTS: CARRY BIT - ABORT TEST FLAG, CLR = ABORT TEST, SET = OK.
; ERRBLK - VALUE WILL BE DASTROYED.
; BMPQOP - THE BMP CODE QUEUE POINTER MAY BE UPDATED.
; THE CONTENTS OF THE BMP CODE QUEUE MAY BE UDATED.
;
; CALLING SEQUENCE: JSR PC,PUFIFR
;
; COMMENTS: THIS ROUTINE REPORTS ERRORS WITH NUMBERS INITIAL ERRNBR
; THRU TO ERRNBR+2.
; THE ERRNBR IS RESTORED TO ITS INITIAL VALUE BEFORE RETURNING.
;
; SUBORDINATE ROUTINES CALLED: ER1603,ER9001,ER9002,SAVBMP.
*****
PUFIFR::SAVE
;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV ERRNBR, -(SP) ;SAVE THE CONTENTS OF THE ERROR NUMBER.
MOV #512, R5 ;SET MAXIMUM READ COUNTER TO 2*FIFO SIZE.
;
; READ DATA FROM THE FIFO UNTIL DATA VALID IS CLEAR OF READ COUNTER IS ZERO.
; REPORT ANY BMP OR UNEXPECTED DATA AS ERRORS.
;
2$: MOV RBUFA, R2 ;GET THE CONTENTS OF THE RECEIVER BUFFER REG.
BPL 8$ ;EXIT IF DATA VALID CLEAR, IE. FIFO PURGED.
;
; CHECK IF READ DATA IS STATUS OR UNEXPECTED CHARACTER.
;
MOV #70000, R0 ;GENERATE A BIT MAP OF CHAR ERROR BITS
BIC R2, R0 ; WHICH ARE NOT SET FOR CHAR.
BNE 4$ ;SKIP BMP CHECK IF IT IS UNEXPECTED DATA.
;
; CHECK IF THE READ DATA IS MODEM STATUS , BMP OR SELFTEST?.
; IF IT IS A BMP CODE THEN SAVE IT ON THE QUEUE.
;
MOV #ER9001, ERRBLK ;SET UP THE CORRECT ERROR REPORTING ROUTINE.
MOV #300, R0 ; CHECK IF BMP OR SELFTEST?.
BIC R2, R0 ;TRY TO CLEAR BMP FLAGS IN THE READ DATA.
BNE 4$ ;SKIP BMP ERROR REPORT IF MODEM OR SELFTEST?.
JSR PC, SAVBMP ;SAVE THE BMP CODE ON THE QUEUE.
BR 6$ ;BRANCH TO CHECK READ COUNT.
;
; CHECK IF THE READ DATA IS MODEM, SELFTEST OR UNEXPECTED DATA.
;

```

```

4791 023010 032702 000001      4$:   BIT    #BIT0,R2      ;TEST THE MODEM STATUS INDICATION BIT.
4792 023014 001425              BEQ    6$              ;DO NOT REPORT ANY ERROR IF MODEM STATUS.
4793 023016 012701 012466      MOV    #EM9104,R1     ;PASS THE CORRECT ERROR MESSAGE TO REPORT.
4794 023022 010203              MOV    R2,R3         ;EXTRACT THE LINE NUMBER FROM
4795 023024 000303              SWAB   R3             ; THE READ DATA.
4796 023026 042703 177760      BIC    #177760,R3    ;
4797 023032 006303              ASL    R3             ;FORM LINE NUMBER TIMES 2 FOR ER9002 ROUTINE.
4798 023034 052704 100000      BIS    #BIT15,R4     ;SET THE "NONE" EXPECTED MESSAGE FLAG.
4799 023040 005267 162252      INC    ERRNBR        ;SET ERROR NUMBER TO INTIAL ERRBR+1.
4800 023044 012767 014520 162250  MOV    #ER9002,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
4801                                ;REPORT ERROR "UNEXPECTED DATA FOUND IN FIFO".
4802 023052                                ;ERROR                                ;
4803 023052 104460                                ;                                ; >>>>> ERROR <<<<<.
4804                                ;                                ; TRAP    C$ERROR
4805                                ;
;+
; EXIT WITH FAILURE IF EXTENDED ERROR REPORTING HAS NOT BEEN ENABLED
;-
4806 023054 032767 000100 157100  BIT    #BIT06,OPTION ;EXIT WITH TEST FAILURE MESSAGE IF
4807 023062 001415              BEQ    7$              ;NO EXTENDED ERROR REPORTING HAS BEEN REQUESTED
4808                                ; DURING THE SOFTWARE QUESTIONS.
4809
4810 023064 005367 162226      DEC    ERRNBR        ;RESTORE ERROR NUMBER TO INTIAL ERRNBR.
4811
4812 023070 005305      6$:   DEC    R5              ;DECREMENT READ COUNTER.
4813 023072 001325              BNE    2$              ;LOOP TO READ NEXT CHAR FROM FIFO IF COUNT > 0.
4814
;+
; THE FIFO WILL NOT CLEAR, REPORT THE ERROR AND INDICATE THAT THE TEST IS TO
; BE ABORTED.
;-
4815
4816
4817
4818 023074 062767 000002 162214  ADD    #2,ERRNBR     ;SET ERROR NUMBER TO INTIAL ERRNBR+2.
4819 023102 012767 014070 162212  MOV    #ER1603,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE.
4820 023110 012701 011670      MOV    #EM9017,R1     ;PASS THE MESSAGE TO BE REPORTED.
4821                                ;REPORT THE ERROR "FIFO WILL NOT PURGE, (DATA VALID STUCK SET)"
4822                                ; "?????? TEST ABORTED".
4823 023114                                ;                                ;
4824 023114 104460                                ;                                ; >>>>> ERROR <<<<<.
4825 023116 000241      7$:   CLC                                ;INDICATE THE TEST IS TO BE ABORTED.
4826 023120 000401              BR    10$             ;EXIT THIS ROUTINE AND ABORT THE CURRENT TEST.
4827 023122 000261      8$:   SEC                                ;SET THE CARRY, DO NOT ABORT THE TEST.
4828
4829 023124 012667 162166      10$:  MOV    (SP)+,ERRNBR  ;RESTORE INITIAL ERROR NUMBER.
4830 023130 004736      60$:  PASS                                ;RESTORE GPRS.
4831                                ; JSR    PC,(SP)+      ;RETURN TO PREG05 SUBRT.
4832                                ; CARRY BIT, SET INDICATES FIFO PURGED, DO NOT
4833 023132 000207              RTS    PC             ; ABORT THE TEST.

```

```

4835 .SBTTL GLOBAL SUBROUTINE - PURRXB -
4836 ;+ *****
4837 ;* - PURGE THE RX BUFFER IN MEMORY ROUTINE -
4838 ;* THIS SUBROUTINE IS USED BEFORE THE BEGINNING OF A TX/RX OF DATA
4839 ;* PATTERNS TO CLEAR OUT THE RX BUFFER AND TO INITIALIZE THE VARIOUS
4840 ;* COUNTERS AND POINTERS RELATED TO THAT BUFFER.
4841 ;*
4842 ;* INPUTS: RXBSTA - LABEL AT THE BEGINNING OF THE RX BUFFER.
4843 ;*
4844 ;* OUTPUTS: RXBCNT - COUNT OF # OF CHARS IN RX BUFFER (CLEARED).
4845 ;* RXBIPT - INPUT POINTER TO RX BUFFER (INITIALIZED).
4846 ;* RXBOPT - OUTPUT POINTER TO RX BUFFER (INITIALIZED).
4847 ;* THE CONTENTS OF THE RX BUFFER ARE CLEARED.
4848 ;*
4849 ;* CALLING SEQUENCE: JSR PC,PURRXB
4850 ;*
4851 ;* COMMENTS:
4852 ;*
4853 ;* SUBORDINATE ROUTINES CALLED: NONE.
4854 ;-- *****
4855
4856 023134 PURRXB:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
023134 004567 162164 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
4857
4858 023140 MOV #RXBOPT,R1 ;GET THE ADDRESS OF THE RX OUTPUT POINTER.
4859 023144 012721 002720 MOV #RXBSTA,(R1)+ ;INITIALIZE THE RX BUFFER OUTPUT POINTER.
4860 023150 012721 002720 MOV #RXBSTA,(R1)+ ;INITIALIZE THE RX BUFFER INPUT POINTER.
4861 023154 005021 2$: CLR (R1)+ ;CLEAR CHAR COUNT AND THE BUFFER AREA.
4862 023156 020127 003120 CMP R1,#RXBEND ;CHECK IF LAST LOCATION HAS BEEN CLEARED.
4863 023162 101774 BLOS 2$ ;LOOP IF NOT DONE.
4864
4865 023164 60$: PASS ;RESTORE GPRS.
023164 004736 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
4866 023166 000207 RTS PC

```

```

4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901 023170
      023170 004567 162130
4902 023174 016704 162116
4903 023200 016703 157026
4904 023204 005067 157276
4905 023210 004700 001320
4906 023214 004767 002440
4907
4908
4909
4910
4911 023220 012701 004642
4912 023224 012702 005042
4913 023230 005021
4914 023232 020102
4915 023234 103775
4916
4917
4918
4919
4920 023236 016701 157004
4921 023242 026767 157234 156722
4922 023250 001402
4923 023252 062701 000062

```

```

.SBTTL GLOBAL SUBROUTINE - RDCHRS -
; * *****
; * - READ AND COMPARE INPUT CHARACTERS ROUTINE -
; * THIS SUBROUTINE READS THE CHARACTERS FROM THE RX BUFFER IN MEMORY.
; * IF CHARACTERS STOP APPEARING IN THE BUFFER WITH DATA VALID SET
; * OR IF MORE THAN THE ALLOWABLE NUMBER OF CHARACTERS HAS BEEN READ FROM
; * THE BUFFER THIS ROUTINE EXITS WITH AN RX COMPLETE INDICATION.
; * EACH READ CHAR IS ANALYZED AND ANY NECESSARY ERRORS ARE REPORTED.
; *
; * INPUTS: ACTLNS - BIT MAP OF THE ACTIVE OUT LINES.
; *          ERRNBR - SET TO ERROR NUMBER OF FIRST ERROR IN THIS ROUTINE.
; *          IBM - MASK OF THE INACTIVE BITS IN A TX OR RX CHAR BYTE.
; *          OSTENC - ADDRESS OF THE END OF THE OUTPUT STORAGE FIFO BUFFER.
; *          OSTPTR - POINTER TO THE NEXT BYTE TO READ FROM OSTORE.
; *          RXBOPT - POINTER INTO THE RX CHAR BUFFER IN MEMORY.
; *          RXTOUT - TIME-OUT VALUE FOR RX OF LAST CHAR.
; *
; * OUTPUTS: ERROR MESSAGES MAY BE PRINTED AT THE OPERATOR'S CONSOLE.
; *          TXDBLF - TX/RX DISABLED FLAG (CLEARED).
; *          TXENBM - TX ENABLE STATE MASK (DESTROYED).
; *          SAVPRI - STORAGE FOR PROCESSOR PRIORITY (DESTROYED).
; *          SAVTEN - STORAGE FOR TX ENABLE STATES (DESTROYED).
; *
; * CALLING SEQUENCE: JSR PC,RDCHRS
; *
; * COMMENTS: THIS ROUTINE REPORTS ERRORS WITH NUMBERS INITIAL ERRNBR
; *           THRU INITIAL ERRNBR + 4.
; *           ERRNBR IS RESTORED BEFORE THIS ROUTINE RETURNS.
; *
; * SUBROUTINES CALLED: CKCHR,NEWCHR,REPCOD,RXIE0,RXIE1,TXENBL,TXIE0,TXIE1,
; *                   WAIBIS.
; * *****
RDCHRS:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
          JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
          MOV ERRNBR,R4 ;PRESERVE THE INITIAL ERROR NUMBER.
          MOV IBM,R3 ;GET THE INACTIVE BIT MASK.
          CLR TXDBLF ;CLEAR THE TX DISABLED FLAG.
          JSR PC,RXIE1 ;TURN ON OUT RECEPTION INTERRUPTS.
          JSR PC,TXIE1 ;TURN ON OUT TRANSMISSION INTERRUPTS.
; *
; * CLEAR ALL RESYNC QUEUES FOR ALL LINES.
; *
; *
; * MOV #DPRSQB,R1 ;GET BASE ADDRESS OF RESYNC QUEUES TABLE.
; * MOV #DPRSQE,R2 ;GET END ADDRESS OF RESYNC QUEUES TABLE.
2$: CLR (R1)+ ;CLEAR A WORD OF THE TABLE.
   CMP R1,R2 ;CHECK IF POINTER AT END OF TABLE.
   BLO 2$ ;LOOP UNTIL TABLE IS CLEAR.
; *
; * WAIT FOR A CHARACTER TO APPEAR IN THE FIFO.
; * IF NO CHARACTER APPEARS WITHIN TIME-OUT PERIOD: EXIT ROUTINE, WE'RE DONE.
; *
4$: MOV RXTOUT,R1 ;GET TIME-OUT FOR SLOWEST BAUD RATE IN USE.
   CMP TXDONF,ACTLNS ;CHECK FOR TRANSMISSION DONE ON ACTIVE LINES.
   BEQ 6$ ;SKIP ADDING 50 MS DELAY IF TX DONE ALL LINES.
   ADD #50.,R1 ;ADD 50 MILLI SEC TO DELAY IF NOT LAST CHAR.

```



```

4924 023256 052701 170000      6$:   BIS      #170000,R1      ;INDICATE TO TEST DATA.VALID BIT.
4925 023262 016702 157424      MOV      RXBOPT,R2      ;INDICATE TO CHECK MEMORY RECEIVE BUFFER.
4926 023266 004767 003520      JSR      PC,WAIBIS     ;WAIT FOR RECEIVED CHAR OR TIME-OUT.
4927 023272 103117              BCC      18$           ;EXIT ROUTINE IF TIME-OUT, WE'RE DONE.
4928
4929 023274 004767 175154      JSR      PC,GETCHR     ;READ A CHARACTER FROM THE MEMORY BUFFER.
4930
4931      ;+
4932      ; CHECK IF THE TX ISR IS DISABLED.
4933      ; RE-ENABLE RX ISR IF THE SPACE FOR NEW CHARS IS LOW ENOUGH.
4934      ; IF THE BUFFER CAN ACCOMODATE MORE CHARS THEN RE-ENABLE TRANSMISSION.
4935 023300 005767 157202      8$:   TST      TXDBLF     ;CHECK IF TX IS DISABLED.
4936 023304 100027              BPL      10$           ;SKIP RX/TX CHECK IF TX NOT DISABLED.
4937 023306 026727 157404 000020  CMP      RXBCNT,#RXBETX ;COMPARE BUFFER COUNT WITH LEVEL TO ENABLE RX.
4938 023314 101023              BHI      10$           ;SKIP ENABLE RX IF BUFFER TOO FULL.
4939 023316 004767 001212      JSR      PC,RXIE1     ;ENABLE RECEPTION INTERRUPTS.
4940 023322 016705 156734      MOV      TXENBM,R5    ;GET THE PRESERVED TX.ENABLE STATES.
4941 023326 026727 157364 000020  CMP      RXBCNT,#RXBETX ;COMPARE BUFFER COUNT WITH LEVEL TO ENABLE TX.
4942 023334 101013              BHI      10$           ;SKIP ENABLING TX IF BUFFER TOO FULL.
4943 023336              GETPRI   R1           ;SAVE THE CURRENT PROCESSOR PRIORITY.
4944 023342              SETPRI   #PRI07        ;DISABLE INTERRUPTS.
4945 023350 004767 002054      JSR      PC,TXENBL    ;ENABLE TRANSMISSION.
4946 023354 005067 157126      CLR      TXDBLF     ;CLEAR THE TX DISABLE FLAG.
4947 023360              SETPRI   R1           ;RE-ENABLE INTERUPTS.
4948 023364              10$:
4949
4950 023364 005367 157106      DEC      CHRTOT     ;DECREMENT THE TOTAL CHAR COUNTER.
4951 023370 001014              BNE      12$           ;SKIP ERROR IF NOT TOO MANY RECEIVED.
4952 023372 010467 161720      MOV      R4,ERRNBR   ;SET ERROR NUMBER TO INITIAL ERRNBR.
4953 023376 012701 012001      MOV      #EM9025,R1  ;SELECT THE PROPER ERROR MESSAGE.
4954 023402 012767 014032 161712  MOV      #ER0503,ERRBLK ;SELECT THE PROPER ERROR REPORT ROUTINE.
4955
4956      ;+
4957      ; REPORT ERROR AT INITIAL ERRNBR.
4958      ; "MORE THAN TWICE THE EXPECTED NUMBER OF CHARACTERS RECEIVED."
4959 023410              ERROR
4960 023412 012767 000001 156604  MOV      #1,FERROR   ;INDICATE THAT AN ERROR HAS BEEN FOUND.
4961
4962 023420 000477              BR       60$         ;EXIT THE ROUTINE, WE'RE GIVING UP.
4963
4964      ;+
4965      ; DETERMINE IF THE CHARACTER IS DATA OR A STATUS CODE.
4966 023422 012700 070000      12$:  MOV      #70000,R0   ;GENERATE A BIT MAP OF CHARACTER ERROR BITS
4967 023426 040200              BIC      R2,R0       ; WHICH ARE NOT SET FOR THE CHARACTER.
4968 023430 001016              BNE      14$         ;SKIP REPORTING OF ERROR CODE IF WE HAVE CHAR.
4969
4970      ;+
4971      ; THE DATA IS EITHER A BMP CODE OR A MODEM STATUS CODE.
4972      ; REPORT THAT THE CODE WAS FOUND.
4973      ; ERRORS REPORTED WITH ERROR NUMBERS >>>> ERRNBR+1 AND ERRNBR+2 <<<<<,

```

```

4974 023432 010467 161660      MOV    R4,ERRNBR      ;GET THE ERROR NUMBER PASSED INTO THIS ROUTINE.
4975 023436 005267 161654      INC    ERRNBR        ;SET ERROR NUMBER TO INITIAL ERRNBR+1.
4976 023442 004767 000222      JSR    PC,REPCOD     ;REPORT THE BMP OR MODEM STATUS CHANGE CODE.
4977
4978 023446 005767 156552      TST    FERROR        ;HAS AN ERROR BEEN DETECTED ?
4979 023452 001423              BEQ    16$           ;NO, THEN BRANCH.
4980 023454 032767 000100 156500  BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED.
4981 023462 001456              BEQ    60$           ;YES, THEN EXIT WITH TEST FAIL MESSAGE.
4982
4983 023464 000416              BR     16$           ;BRANCH TO GET THE NEXT CHARACTER
4984
4985      ;+
4986      ; THE DATA IS A VALID CHARACTER:
4987      ; COMPARE THE READ DATA WITH THE EXPECTED DATA.
4988      ; UPDATE EXPECTED DATA POINTER.
4989      ; ERRORS REPORTED WITH ERROR NUMBERS >>>> ERRNBR+3 AND ERRNBR+4 <<<<<.
4990 023466 010467 161624      14$:  MOV    R4,ERRNBR      ;CALCULATE THE STARTING ERROR NUMBER FOR THE
4991 023472 062767 000003 161616  ADD    #3,ERRNBR     ; NEXT ROUTINE CALL (INITIAL ERRNBR+3).
4992 023500 004767 176010      JSR    PC,NEWCHR     ;HANDLE THE NEW DATA CHARACTER.
4993 023504 005767 156514      TST    FERROR        ;HAS AN ERROR BEEN DETECTED ?
4994 023510 001404              BEQ    16$           ;NO, THEN BRANCH.
4995 023512 032767 000100 156442  BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED.
4996 023520 001437              BEQ    60$           ;YES, THEN EXIT WITH TEST FAIL MESSAGE.
4997
4998      ;+
4999      ; DONE PROCESSING THIS CHARACTER.
5000      ; READ ANOTHER CHAR FROM THE DUT FIFO.
5001      ; IF DATA.VALID IS SET, LOOP TO CHECK THE RECEIVED CHARACTER.
5002      ; IF DATA.VALID IS CLEAR LOOP TO WAIT FOR IT SET OR TIME-OUT.
5003 023522 004767 174726      16$:  JSR    PC,GETCHR     ;READ A CHARACTER FROM THE RX BUFFER.
5004 023526 103664              BCS   8$             ;IF DATA.VALID SET, GO TO CHECK THE RX CHAR.
5005 023530 000642              BR    4$             ;LOOP TO WAIT CHAR OR TIME-OUT IF BUFFER EMPTY.
5006
5007      ;+
5008      ; USE DUMMY CHARACTERS TO FORCE ANALYSIS OF CHARACTERS IN RESYNC QUEUES.
5009 023532 004767 000766      18$:  JSR    PC,RXIE0     ;TURN OFF DUT RX INTERRUPTS.
5010 023536 004767 001462      JSR    PC,TXDONE     ;CHECK IF TX DONE, TURN OFF DUT TX INTERRUPTS.
5011 023542 005002              CLR    R2            ;CLEAR THE DUMMY CHARACTER.
5012 023544 005001              CLR    R1            ;CLEAR THE LOOP COUNTER.
5013 023546 004767 175742      20$:  JSR    PC,NEWCHR     ;FORCE ONE RESYNC QUE CHAR TO BE ANALYZED.
5014
5015 023552 005767 156446      TST    FERROR        ;HAS AN ERROR BEEN DETECTED ?
5016 023556 001404              BEQ    22$           ;NO, THEN BRANCH.
5017 023560 032767 000100 156374  BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED.
5018 023566 001414              BEQ    60$           ;YES, THEN EXIT WITH TEST FAIL MESSAGE.
5019
5020 023570 062702 000400      22$:  ADD    #400,R2       ;INCREMENT THE LINE NUMBER IN THE DUMMY CHAR.
5021 023574 005201              INC    R1            ;INCREMENT THE LOOP COUNTER.
5022 023576 120127 000020      CMPB   R1,#NUMLNS    ;TEST FOR LOOP COUNTER EQUAL TO # OF DUT LINES.
5023 023602 002761              BIT    20$           ;LOOP IF LOOP COUNT IS NOT ALL LINES DONE.
5024 023604 005701              TST    R1            ;CHECK FOR SECOND TIME AROUND OUTER LOOP.
5025 023606 100404              BMI   60$           ;EXIT IF OUTER LOOP DONE TWICE.
5026 023610 005002              CLR    R2            ;CLEAR THE DUMMY CHAR FOR 2ND TIME AROUND LOOP.
5027 023612 012701 100000      MOV    #100000,R1    ;CLEAR LOOP COUNT, SET OUTER LOOP FLAG.
5028 023616 000753              BR    20$           ;LOOP THE SECOND TIME AROUND OUTER LOOP.
5029
5030 023620 010467 161472      60$:  MOV    R4,ERRNBR     ;RESTORE THE ERROR NUMBER TO ITS INITIAL VALUE.
    
```

B11

DHU-11 FUNC TST PART4 MACRO M1200 15 MAR-84 09:48 PAGE 88-3
GLOBAL SUBROUTINE - RDCHRS -

SEQ 131

5031	023624		PASS			RESTORE GPRS.	
	023624	004736			JSR	PC, @ (SP)+	RETURN TO PREG05 SUBRT.
5032	023626	000207	RTS	PC			

```

5034 .SBTTL GLOBAL SUBROUTINE - RDMAST -
5035 ;* *****
5036 ;* - REPORT DMA_START BIT ERRORS ROUTINE -
5037 ;* THIS SUBROUTINE CHECKS FOR LINES WHICH HAVE DMA_START BIT ERRORS
5038 ;* DURING THE JUST COMPLETED DMA TRANSMISSION. IF ANY ARE FOUND,
5039 ;* THEY ARE REPORTED.
5040 ;*
5041 ;* INPUTS: ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE FOR THIS ROUTINE.
5042 ;* ERRNBR - ERROR NUMBER OF ERROR REPORTED IN THIS ROUTINE.
5043 ;* TXINTF - CONTAINS BIT MAP OF LINES WITH DMA_START BIT ERRORS.
5044 ;*
5045 ;* OUTPUTS: ERRBLK - ADDRESS OF THE ERROR REPORTING ROUTINE (DESTROYED).
5046 ;* MESSAGES MAY BE PRINTED AT THE OPERATOR CONSOLE.
5047 ;*
5048 ;* CALLING SEQUENCE: JSR PC,RDMAST
5049 ;*
5050 ;* COMMENTS: IF NO LINES HAVE DMA_START BIT ERRORS, NO MESSAGES ARE PRINTED.
5051 ;*
5052 ;* SUBORDINATE ROUTINES CALLED: ER9102.
5053 ;* *****
5054
5055 RDMAST:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
5056 023630 004567 161470 MOV TXINTF,R2 ;GET COPY OF THE DMA_START ERRORS BIT MAP.
5057 023634 016702 156424 BEQ 60$ ;EXIT IF NO DMA_START ERROR BITS ARE SET.
5058 ;*
5059 ;* WE HAVE SOME DMA_START BIT ERRORS TO REPORT.
5060 ;*
5061 023642 012767 015504 161452 MOV @ER9102,ERRBLK ;SELECT THE ERROR REPORTING ROUTINE.
5062 023650 012701 012377 MOV @EM9102,R1 ;INDICATE THAT WE HAVE DMA_START BIT ERROR.
5063 ;*
5064 ;* REPORT "DMA_START BIT SET AFTER RESET OR TX ACTION ... ON LINE(S):"
5065 ;*
5066 023654 104460 ERROR ; >>>> ERROR <<<<<.
; TRAP C:ERROR
5067 023656 012767 000001 156340 MOV @1,FERROR ;INDICATE AN ERROR HAS BEEN DETECTED.
5068 ;*
5069 023664 004736 60$: PASS ;RESTORE GPRS.
; PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
5070 023666 000207 RTS PC

```

```

5072
5073
5074
5075
5076
5077
5078
5079
5080
5081
5082
5083
5084
5085
5086
5087
5088
5089
5090
5091
5092
5093
5094 023670
      023670 004567 161430
5095 023674 012767 014420 161420
5096 023702 016703 161410
5097 023706 010204
5098 023710 000304
5099 023712 042704 177760
5100
5101
5102
5103 023716 012701 011173
5104 023722 032702 000001
5105 023726 001422
5106 023730 005267 161362
5107 023734 012701 011215
5108 023740 012700 000300
5109 023744 040200
5110 023746 001003
5111 023750 004767 000604
5112 023754 000423
5113 023756 122702 000201
5114 023762 001416
5115 023764 122702 000203
5116 023770 001413
5117 023772 000400
5118
5119
5120
5121 023774 042702 177400
5122 024000 004767 002154
5123 024004
      024004 104460
5124 024006 012767 000001 156210
5125
5126 024014 004767 002200

```

```

.SBTTL GLOBAL SUBROUTINE - REPCOD -
*****
- ROUTINE TO REPORT ERROR CODE FROM DUT -
THIS ROUTINE REPORTS AN ERROR CODE WHICH HAS BEEN READ FROM THE DUT
FIFO. THE CODE IS CHECKED TO DETERMINE WHETHER IT IS A SELFTEST CODE
AN MODEM STATUS CHANGE CODE OR A BMP CODE. THIS ROUTINE ASSUMES THAT
THE CODE INDICATES AN ERROR. IF A BMP CODE IS FOUND IT IS NOT REPORTED
IMMEDIATELY, BUT IS SAVED ON THE BMP CODE QUEUE TO BE REPORTED LATER.
INPUTS: R2 - CONTAINS THE ERROR CODE COMPLETE WITH FLAGS AND LINE #.
ERRTBL - ERRTP,ERRNBR,AND ERRMSG SET UP CORRECTLY.
OUTPUTS: ERRBLK - VALUE MAY BE DESTROYED.
BMPCPQ - MAYBE UPDATED IF A BMP CODE IS ADDED TO THE QUEUE.
CALLING SEQUENCE: JSR PC,REPCOD
COMMENTS: ERRNBR IS RESTORED TO ITS ENTERING VALUE BY THIS ROUTINE.
THIS ROUTINE REPORTS ERRORS WITH NUMBERS ERRNBR THRU ERRNBR+1.
SUBORDINATE ROUTINES CALLED: ER9001,SAVBMP.
*****
REPCOD:; SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV #ER9001,ERRBLK ;SELECT THE ERROR REPORTING ROUTINE.
MOV ERRNBR,R3 ;PRESERVE THE ERROR NUMBER.
MOV R2,R4 ;EXTRACT THE LINE NUMBER FIELD
SWAB R4 ; FROM THE ERROR CODE WHICH WAS
BIC #177760,R4 ; PASSED INTO THIS ROUTINE.
;
; DETERMINE THE TYPE OF CODE WHICH IS TO BE REPORTED.
;
MOV #EM9003,R1 ;SELECT MODEM STATUS CODE MESSAGE.
BIT #BIT0,R2 ;TEST THE MODEM STATUS INDICATION BIT.
BEQ 4$ ;GOTO REPORT ERROR IF MODEM STATUS CODE.
INC ERRNBR ;SELECT THE SELFTEST CODE ERROR NUMBER.
MOV #EM9004,R1 ;SELECT SELFTEST CODE MESSAGE.
MOV #300,R0 ;CHECK IF SELF-TEST OR BMP CODE.
BIC R2,R0 ;TRY TO CLEAR BMP BITS.
BNE 2$ ;GO CHECK FOR SELFTEST CODE IF NOT BMP.
JSR PC,SAVBMP ;SAVE THE BMP CODE ON THE QUEUE.
BR 60$ ;EXIT THIS ROUTINE.
2$: CMPB #201,R2 ;CHECK FOR SELF TEST NULL CODE.
BEQ 6$ ;EXIT ROUTINE IF NULL CODE FOUND.
CMPB #203,R2 ;CHECK FOR SKIP SELF TEST CODE.
BEQ 6$ ;EXIT ROUTINE IF SKIP SELF TEST CODE FOUND.
BR 4$ ;GO REPORT SELF TEST ERROR.
;
; REPORT "UNEXPECTED XXXXX CODE FOUND IN RECEIVE CHAR FIFO."
4$: BIC #177400,R2 ;REMOVE UPPER BYTE OF CODE TO BE REPORTED.
JSR PC,TXROFF ;TURN OFF TX AND RX DURING ERROR REPORTING.
ERROR ; >>>> ERROR <<<<<. TRAP C$ERROR
MOV #1,FERROR ;INDICATE THAT AN ERROR HAS BEEN FOUND.
JSR PC,TXRON ;TURN TX AND RX BACK ON.

```

E11

DHU-11 FUNC ISI PART4 MACRO M1200 15 MAR 84 09:48 PAGE 90-1
GLOBAL SUBROUTINE - REPCOD -

SEQ 134

```
5127          ;  
5128          ; RESTORE THE INITIAL ERROR NUMBER.  
5129          ;  
5130 024020 010367 161272 6$:      MOV      R3,ERRNBR  
5131          ;  
5132 024024          60$:      PASS          ;RESTORE GPRS.  
          024024 004736          JSR      PC,0(SP)+  
5133 024026 000207          RTS      PC          ;RETURN TO PREG05 SUBRT.
```

5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175

024030
024030 004567 161270
024034 005767 156440
024040 001404
024042 012767 015070 161252
024050
024050 104460
024052
024052 004736
024054 000207

```

.SBTTL GLOBAL SUBROUTINE - REPSMR -
;+ *****
;+ - REPORT ERROR SUMMARY ROUTINE -
;+ 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: ERCNTB - LABEL AT BASE OF LINE ERROR COUNTERS TABLE.
;+ ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE.
;+ ERRNBR - ERROR NUMBER OF ERRORS IN THIS ROUTINE.
;+ ERSMRF - "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 ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
TST ERSMRF ;CHECK THE "PRINT LINE ERROR SUMMARY" FLAGS.
BEQ 60$ ;EXIT WITHOUT ACTION IF NO SUMMARY FLAGS SET.
;+
;+ WE HAVE SOME ERROR SUMMARIES TO REPORT.
;+
;+ MOV #ER9004,ERRBLK ;SELECT ERROR REPORTING ROUTINE.
;+
;+ REPORT
;+ "ERROR SUMMARY REPORT FOR LINES WITH EXCESSIVE NUMBERS OF ERRORS:"
;+
;+ ERROR TRAP C$ERRRJR
60$: PASS ;RESTORE GPRS.
; PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC

```

```

5177 .SBTTL GLOBAL SUBROUTINE - RESETT -
5178 ;*****
5179 ;* - RESET DEVICE UNDER TEST -
5180 ;* THIS SUBROUTINE IS USED TO RESET THE DUT TO A KNOWN STATE.
5181 ;* IF RESET DOES NOT SUCCESSFULLY COMPLETE, IE. TIME-OUT OCCURS, THEN
5182 ;* AN ABORT TEST ERROR MESSAGE IS REPORTED.
5183 ;*
5184 ;* INPUTS: CSRA - CONTAINS THE ADDRESS OF THE CSR
5185 ;* TXBFCA - CONTAINS ADDRESS OF DUT DMA BUFFER COUNT REGISTER.
5186 ;* ERRBTBL - ERRRTYP,ERNBR,AND ERRMSG SET UP CORRECTLY.
5187 ;*
5188 ;* OUTPUTS: THE DUT PERFORMS ITS RESET FUNCTION INTO A KNOWN STATE.
5189 ;* CARRY - CLEAR INDICATES THE TEST IS TO BE ABORTED.
5190 ;* ERRBLK - VALUE MAY BE DESTROYED.
5191 ;* IESTAT - TX AND RX INTERRUPT FLAGS ARE CLEARED.
5192 ;* TX AND RX INTERRUPT ENABLE BITS IN THE DUT'S CSR ARE CLEARED.
5193 ;*
5194 ;* CALLING SEQUENCE: JSR PC,RESETT
5195 ;*
5196 ;* COMMENTS: THIS SUBROUTINE CAN REPORT ERRORS WITH NUMBERS INITIAL ERRNBR
5197 ;* THIS ROUTINE DOES NOT DESTROY THE VALUE OF ERRNBR.
5198 ;*
5199 ;* SUBORDINATE ROUTINES CALLED: DELAY,MSLGET.
5200 ;*****
5201
5202 024056 RESETT:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU P5.
024056 004567 161242 ;RS,PREG05 ;CALL REGISTER SAVE SUBRT.
5203 024062 012702 000040 MOV #BIT05,R2 ;SET BIT MASK OF MASTER RESET BIT.
5204
5205 ;*
5206 ;* TEST THE STATE OF THE MASTER RESET BIT IN THE CSR.
5207 ;* IF MR IS SET THEN WAIT FOR SELF-TEST TO COMPLETE.
5208 ;* IF TIME-OUT OCCURS, REPORT THE ERROR AND PASS-OUT ABORT TEST INDICATOR.
5209 024066 016704 156106 MOV CSRA,R4 ;GET THE ADDRESS OF THE DUT'S CSR.
5210 024072 030214 BIT R2,(R4) ;CHECK STATE OF MASTER RESET BIT.
5211 024074 001406 BEQ 2$ ;DON'T DELAY IF MR IS ALREADY CLEAR.
5212 024076 005003 CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
5213 024100 012701 011610 MOV #5000.,R1 ;PASS TIME-OUT VALUE OF 5 SECONDS.
5214 024104 004767 174764 JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
5215 024110 103012 BCC 4$ ;GO REPORT ERROR IF TIMEOUT OCCURRED.
5216
5217 ;*
5218 ;* SET MASTER RESET BIT IN CSR. CLEAR TX AND RX ENABLE BITS, ETC.
5219 ;* SKIP THE SELFTEST.
5220 ;* TIME-OUT OF 5 SECS, JUST IN CASE THE SELF-TEST EXECUTES.
5221 ;*
5222 024112 010277 156062 2$: MOV R2,@CSRA ;SET MASTER RESET BIT, DISABLE TX AND RX INTS.
5223 024116 004767 000504 JSR PC,SKPSTS ;TRY TO SKIP THE SELFTEST.
5224
5225 ;*
5226 ;* SET SELF-TEST TIME-OUT OF 5 SECONDS, AND WAIT FOR M,R TO CLEAR.
5227 ;* IF TIME-OUT OCCURS, THEN REPORT THE FATAL ERROR AND PASS-OUT THE ABORT
5228 ;* TEST INDICATOR.
5229 024122 005003 CLR R3 ;SET UP DESIRED STATE OF MASTER RESET BIT.
5230 024124 012701 011610 MOV #5000.,R1 ;PASS TIME-OUT VALUE OF 5 SECONDS.
5231 024130 004767 174740 JSR PC,MSLGET ;WAIT FOR SELF-TEST TO COMPLETE, MR CLEAR.
5232 024134 103410 BCS 6$ ;SKIP ERROR REPORT IF MR CLEARED IN TIME.

```



```

5233
5234
5235
5236
5237 024136 012701 010111
5238 024142 012767 014070 161152
5239
5240
5241 024150
      024150 104460
5242 024152 000241
5243 024154 000403
5244
5245
5246
5247
5248 024156 005067 156052
5249 024162 000261
5250
5251 024164
      024164 004736
5252
5253 024166 000207
5254
;+
; SET UP ERROR MESSAGE TO REPORT "FATAL ERROR FOUND DURING RESET, TEST ABORTED".
; INDICATE TEST IS TO BE ABORTED BY CLEARING THE CARRY BIT.
;-
4$:  MOV    #EM1601,R1      ;PASS ERROR MESSAGE TO REPORT.
      MOV    #ER1603,ERRBLK ;PASS ADDRESS OF ERROR HANDLING ROUTINE.
      ;REPORT ERROR "TIME-OUT OCCURRED WAITING FOR MASTER RESET TO CLEAR"
      ; "TEST ABORTED"
      ERROR                                ;          >>>> ERROR <<<<<
                                           TRAP    C$ERROR
      CLC                                ;INDICATE TEST IS TO BE ABORTED.
      BR     60$                       ;EXIT THIS SUBROUTINE, ABORT TEST INDICATOR.
;+
; CLEAR TX AND RX INTERRUPT ENABLE STATUS FLAGS IN IESTAT.
; EXIT WITH CONTINUE TEST INDICATOR SET (IE,CARRY SET).
;-
6$:  CLR    IESTAT          ;CLEAR TX AND RX INTERRUPT STATUS FLAGS.
      SEC                                ;INDICATE SUCCESS, CONTINUE TEST.
;+
60$: PASS
      JSR                                ;RESTORE GPRS, PASS THE FOLLOWING INTACT:
      PC,@(SP)+                ;RETURN TO PREG05 SUBRT.
      ;CARRY BIT;IF CLEAR,INDICATES ABORT TEST.
      RTS    PC

```

5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299

024170
024170 004567 161130
024174 010502
024176 046702 156302
024202 001413

024204 012767 015204 161110
024212 012701 011661
024216 012704 003542

024222
024222 104460
024224 012767 000001 155772

024232
024232 004736
024234 000207

```
.SBTTL GLOBAL SUBROUTINE - RRXNDN -
;*****
; - REPORT RECEPTION NOT COMPLETED ROUTINE -
; THIS SUBROUTINE CHECKS FOR LINES WHICH DID NOT RECEIVE THE COMPLETE
; DATA PATTERN. IF ANY ARE FOUND, THEY ARE REPORTED.
;
; INPUTS: R5 - LOCAL ACTIVE LINES BIT MAP.
; DPLENB - BASE OF TABLE OF DATA PATTERN LENGTHS.
; ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE FOR THIS ROUTINE.
; ERRNR - ERROR NUMBER OF ERROR REPORTED IN THIS ROUTINE.
; RXCNTB - LABEL AT BASE OF THE RX CHARACTER COUNTERS TABLE.
; RXDNFB - RECEPTION DONE FLAGS.
;
; OUTPUTS: ERRBLK - ADDRESS OF THE ERROR REPORTING ROUTINE (DESTROYED).
; MESSAGES MAY BE PRINTED AT THE OPERATOR CONSOLE.
;
; CALLING SEQUENCE: JSR PC,RRXNDN
;
; COMMENTS: IF NO LINES FAILED TO COMPLETE THEIR RECEPTION, NO MESSAGES
; ARE PRINTED.
;
; SUBORDINATE ROUTINES CALLED: ER9005.
; - *****
RRXNDN:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R5,R2 ;GET COPY OF THE LOCAL ACTIVE LINES BIT MAP.
BIC RXDNFB,R2 ;GET MAP OF ACTIVE LINES WITH RX DONE FLAG CLR.
BEQ 60$ ;EXIT IF NO ACTIVE LINES HAVE RX DONE FLAG CLR.
;
; WE HAVE SOME "RX NOT COMPLETED" ERRORS TO REPORT.
;
; MOV #ER9005,ERRBLK ;SELECT THE ERROR REPORTING ROUTINE.
; MOV #EM9016,R1 ;INDICATE THAT WE ARE DEALING WITH RECEPTION.
; MOV #RXCNTB,R4 ;PASS BASE OF RX CHAR COUNTERS TABLE TO ER9005.
;
; REPORT "SINGLE CHARACTER MODE TEST ERROR:"
; "DATA PATTERN NOT COMPLETELY RECEIVED ON ALL LINES:"
; ...
;
; ERROR
;
; MOV #1,FERROR ;INDICATE AN ERROR HAS BEEN FOUND. TRAP C$ERROR
;
60$: PASS ;RESTORE GPRS.
RTS PC,JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
```

5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325 024236
024236 004567 161062
5326 024242 010502
5327 024244 046702 156232
5328 024250 001413
5329
5330
5331
5332 024252 012767 015204 161042
5333 024260 012701 011645
5334 024264 012704 003502
5335
5336
5337
5338
5339
5340
5341 024270
024270 104460
5342 024272 012767 000001 155724
5343
5344 024300
024300 004736
5345 024302 000207

```

.SBTTL GLOBAL SUBROUTINE - RTXNDN -
;+ *****
;+ - REPORT TRANSMISSION NOT COMPLETED ROUTINE -
;+ THIS SUBROUTINE CHECKS FOR LINES WHICH DID NOT TRANSMIT THE COMPLETE
;+ DATA PATTERN. IF ANY ARE FOUND, THEY ARE REPORTED.
;+
;+ INPUTS: R5 - LOCAL ACTIVE LINES BIT MAP.
;+ DPLENB - LABEL AT BASE OF DATA PATTERN LENGTH TABLE.
;+ ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE FOR THIS ROUTINE.
;+ ERRNR - ERROR NUMBER OF ERROR REPORTED IN THIS ROUTINE.
;+ TXCNTB - LABEL AT BASE OF THE TX CHARACTER COUNTERS TABLE.
;+ TXDONF - TRANSMISSION DONE FLAGS.
;+
;+ OUTPUTS: ERRBLK - ADDRESS OF THE ERROR REPORTING ROUTINE (DESTROYED).
;+ MESSAGES MAY BE PRINTED AT THE OPERATOR CONSOLE.
;+
;+ CALLING SEQUENCE: JSR PC,RTXNDN
;+
;+ COMMENTS: IF NO LINES FAILED TO COMPLETE THEIR TRANSMISSION, NO MESSAGES
;+ ARE PRINTED.
;+
;+ SUBORDINATE ROUTINES CALLED: ER9005.
;+ - *****
RTXNDN:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R5,R2 ;GET COPY OF THE LOCAL ACTIVE LINES BIT MAP.
BIC TXDONF,R2 ;GET MAP OF ACTIVE LINES WITH TX DONE FLAG CLR.
BEQ 60$ ;EXIT IF NO ACTIVE LINES HAVE TX DONE FLAG CLR.
;+
;+ WE HAVE SOME "TX NOT COMPLETED" ERRORS TO REPORT.
;+
;+ MOV @ER9005,ERRBLK ;SELECT THE ERROR REPORTING ROUTINE.
;+ MOV @EM9015,R1 ;INDICATE WE ARE DEALING WITH TRANSMISSION.
;+ MOV @TXCNTB,R4 ;PASS BASE OF TX CHAR COUNTERS TO TABLE ER0805.
;+
;+ REPORT "SINGLE CHARACTER MODE TEST ERROR:"
;+ "DATA PATTERN NOT COMPLETELY TRANSMITTED ON ALL LINES:"
;+ ...
;+
;+ ERROR ; >>>> ERROR <<<<<. TRAP C$ERROR
MOV #1,FERROR ;INDICATE THAT AN ERROR HAS BEEN FOUND.
60$: PASS ;RESTORE GPRS.
; PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC JSR

```

```

5347 .SBTTL GLOBAL SUBROUTINE - RXDSBL -
5348 ;+ *****
5349 ;* - DISABLE RECEIVERS -
5350 ;* THIS SUBROUTINE IS USED TO DISABLE RECEPTION ON SELECTED LINES BY,
5351 ;* CLEARING THE ASSOCIATED RX_ENABLE BIT ON THE DUT.
5352 ;*
5353 ;* INPUTS: R5 - BIT'S SET CORRESPOND TO LINES ON WHICH TO CLEAR RX_ENABLE.
5354 ;* CSRA - CONTAINS THE ADDRESS OF THE DUT CSR.
5355 ;* IESTAT - CONTAINS THE STATE OF TXIE AND RXIE BITS IN THE CSR.
5356 ;* NUMLNS - EQUATED TO BE THE MAXIMUM NUMBER OF LINES AVAILABLE.
5357 ;* LNCTRA - CONTAINS THE ADDRESS OF THE LNCTRL REGISTER.
5358 ;*
5359 ;* OUTPUTS: R5 - BIT'S SET INDICATE INITIAL STATES OF ALL RX_ENABLE BITS.
5360 ;* LNCTRA - THE STATE OF THE RX_ENABLE BIT MAY BE ALTERED.
5361 ;* THE CONTENTS OF THE IND_ADD_REG FIELD IN THE CSR ARE DESTROYED.
5362 ;*
5363 ;* CALLING SEQUENCE: JSR PC,RXDSBL
5364 ;*
5365 ;* COMMENTS:
5366 ;*
5367 ;* SUBORDINATE ROUTINES CALLED: NONE.
5368 ;- *****
5369
5370 RXDSBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
5371 024304 004567 161014 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
5372 024310 010500 MOV R5,R0 ;COPY BIT MAP OF LINES TO DISABLE RECEPTION.
5373 024312 012701 000001 MOV #BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
5374 024316 016702 155666 MOV LNCTRA,R2 ;GET THE ADDRESS OF THE LNCTRL REGISTER.
5375 024322 012703 000020 MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER PLUS ONE.
5376 024326 016704 15570C MOV IESTAT,R4 ;GET THE STATES OF THE INT ENABLE BITS.
5377 024332 005005 CLR R5 ;LOG POSSIBLE RX DISABLED ON ALL LINES.
5378 ;+
5379 ; SELECT EVERY LINE IN TURN, AND LOG THE STATE OF EACH RX_ENABLE BIT.
5380 024334 010477 155640 2$: MOV R4,@CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
5381 024340 032712 000004 BIT #BIT2,(R2) ;CHECK STATE OF RX_ENABLE BIT ON SELECTED LINE.
5382 024344 001401 BEQ 4$ ;SKIP NEXT INSTRUCTION IF RX_ENABLE CLEAR.
5383 024346 050105 BIS R1,R5 ;LOG RX_ENABLE BIT SET FOR SELECTED LINE.
5384 ;+
5385 ; CLEAR RX_ENABLE ON LINES THAT HAVE A CORRESPONDING BIT SET IN THE RX_DISABLE
5386 ; LINE BIT MAP.
5387 ;-
5388 024350 030100 4$: BIT R1,R0 ;CHECK STATE OF DISABLE LINE BIT MAP.
5389 024352 001402 BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
5390 024354 042712 000004 BIC #BIT2,(R2) ;CLEAR RX_ENABLE BIT ON SELECTED LINE.
5391 024360 005204 6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
5392 024362 006301 ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
5393 024364 005303 DEC R3 ;DECREMENT LINE NUMBER.
5394 024366 001362 BNE 2$ ;LOOP TO CHECK NEXT LINE.
5395 ;-
5396 024370 010566 000014 60$: PASS R5 ;RESTORE GPRS,EXCEPT
5397 024374 004736 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
5398 024376 000207 JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
;R5 - PREVIOUS STATES OF ALL RX_ENABLE BITS.
RTS PC

```

```

5400 .SBTTL GLOBAL SUBROUTINE - RXENBL -
5401 ; ** *****
5402 ; * - ENABLE RECEIVER -
5403 ; * THIS SUBROUTINE IS USED TO ENABLE RECEPTION ON SELECTED LINES BY
5404 ; * SETTING THE ASSOCIATED RX.ENABLE BIT ON THE DUT.
5405 ; *
5406 ; * INPUTS: R5 - BIT'S SET CORRESPOND TO LINES ON WHICH TO SET RX.ENABLE.
5407 ; * CSRA - CONTAINS THE ADDRESS OF THE DUT CSR.
5408 ; * IESTAT - CONTAINS THE STATE OF IXIE AND RXIE BITS IN THE CSR.
5409 ; * NUMLNS - EQUATED TO BE THE MAXIMUM NUMBER OF LINES AVAILABLE.
5410 ; * LNCTRA - CONTAINS THE ADDRESS OF THE LNCTRL REGISTER.
5411 ; *
5412 ; * OUTPUTS: R5 - BIT'S SET INDICATE PREVIOUSLY DISABLED LINES.
5413 ; * LNCTRA - THE STATE OF THE RX.ENABLE BIT MAY BE ALTERED.
5414 ; * THE CONTENTS OF THE IND.ADD.REG FIELD IN THE CSR ARE DESTROYED.
5415 ; *
5416 ; * CALLING SEQUENCE: JSR PC,RXENBL
5417 ; *
5418 ; * COMMENTS:
5419 ; *
5420 ; * SUBORDINATE ROUTINES CALLED: NONE.
5421 ; - * *****
5422
5423 024400 RXENBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
024400 004567 160720 JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
5424 024404 010500 MOV R5,R0 ;COPY BIT MAP OF LINES TO ENABLE.
5425 024406 012701 000001 MOV #BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
5426 024412 016702 155572 MOV LNCTRA,R2 ;GET THE ADDRESS OF THE LNCTRL REGISTER.
5427 024416 012703 000020 MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER.
5428 024422 016704 155606 MOV IESTAT,R4 ;GET THE STATES OF THE INT ENABLE BITS.
5429 024426 005005 CLR R5 ;CLEAR RX.ENABLE BIT LOG OF DISABLED LINES.
5430 ; *
5431 ; * SELECT EVERY LINE IN TURN,AND LOG ANY RX.ENABLE BIT THAT IS CLEAR.
5432 ; -
5433 024430 010477 155544 2$: MOV R4,#CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
5434 024434 032712 000004 BIT #BIT2,(R2) ;CHECK STATE OF RX.ENABLE BIT ON SELECTED LINE.
5435 024440 001001 BNE 4$ ;SKIP NEXT INSTRUCTION IF RX.ENABLE SET.
5436 024442 050105 BIS R1,R5 ;LOG RX ENABLE BIT CLEAR FOR SELECTED LINE.
5437 ; *
5438 ; * SET RX.ENABLE ON LINES THAT HAVE A CORRESPONDING BIT SET IN THE RX ENABLE
5439 ; * LINE BIT MAP.
5440 ; -
5441 024444 030100 4$: BIT R1,P0 ;CHECK STATE OF RX.ENABLE LINE BIT MAP.
5442 024446 001402 BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
5443 024450 052712 000004 BIS #BIT2,(R2) ;ENABLE RECEPTION ON SELECTED LINE.
5444 024454 005204 6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
5445 024456 006301 ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
5446 024460 005303 DEC R3 ;DECREMENT LINE NUMBER.
5447 024462 001362 BNE 2$ ;LOOP TO CHECK NEXT LINE.
5448
5449 024464 60$: PASS R5 ;RESTORE GPRS,EXCEPT
024464 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
024470 004736 JSR PC,#(SP)+ ;RETURN TO PREG05 SUBRT.
5450 ;R5 - LINE BIT MAP CORRESPONDING TO THE
5451 ; PREVIOUS LINES THAT WERE DISABLED.
5452 024472 000207 RTS PC

```

```

5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472 024474 010046
5473 024476 104440
      024476 104440
      024500 010046
5474 024502
      024502 012700 000340
      024506 104441
5475 024510 042767 137777 155516
5476 024516 016777 155512 155454
5477 024524
      024524 012600
      024526 104441
5478 024530 012600
5479 024532 000207

```

```

.SBTL GLOBAL SUBROUTINE - RXIE0
; *****
; - RECEIVER INTERRUPT DISABLE -
; THIS ROUTINE IS USED TO DISABLE RECEIVER INTERRUPTS IN THE DHU11.
;
; INPUTS: NONE.
;
; OUTPUTS: THE RX.INT.ENBL BIT IS CLEARED IN THE DUT CSR.
;          IESTST -CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT
;          ENABLE BITS.
;
; CALLING SEQUENCE: JSR PC,RXIE0
;
; COMMENTS: THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
;           THE DUT CSR ARE DESTROYED.
;
; SUBORDINATE ROUTINES CALLED: NONE.
; *****
RXIE0:: MOV RO,-(SP) ;SAVE CONTENTS OF RO ON THE STACK.
        GETPRI -(SP) ;SAVE PROCESSOR PRIORITY ON STACK.
;
; TRAP C$GPRI
; MOV RO,-(SP)
5474 SETPRI #PRI07 ;IGNORE ANY INTERRUPT THAT MAY BE GENERATED.
; MOV #PRI07,RO
; TRAP C$SPRI
5475 BIC #137777,IESTAT ;CLEAR RX.INT.ENBL BIT IN IESTAT.
5476 MOV IESTAT,@CSRA ;DISABLE RX INTERRUPTS.
5477 SETPRI (SP)+ ;ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
; MOV (SP)+,RO
; TRAP C$SPRI
; MOV (SP)+,RO
; TRAP C$SPRI
; RESTORE RO.
RTS PC

```

```

5481 .SBTTL GLOBAL SUBROUTINE - RXIE1 -
5482 ;* *****
5483 ;* - RECEIVER INTERRUPT ENABLE -
5484 ;* THIS ROUTINE IS USED TO ENABLE RECEIVER INTERRUPTS IN THE DHU11.
5485 ;*
5486 ;* INPUTS: NONE.
5487 ;*
5488 ;* OUTPUTS: THE RX.INT.ENBL BIT IS SET IN THE DUT CSR.
5489 ;* IESTAT -CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT
5490 ;* ENABLE BITS.
5491 ;*
5492 ;* CALLING SEQUENCE: JSR PC,RXIE1
5493 ;*
5494 ;* COMMENTS: THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
5495 ;* THE DUT CSR ARE DESTROYED.
5496 ;*
5497 ;* SUBORDINATE ROUTINES CALLED: NONE.
5498 ;* - - *****
5499
5500 024534 052767 000100 155472 RXIE1:: BIS #BIT06,IESTAT ;SET RX.INT.ENBL BIT IN IESTAT.
5501 024542 042767 137677 155464 BIC #137677,IESTAT ;CLEAR ALL OTHER BITS, EXCEPT TX AND RX I.E.
5502 024550 016777 155460 155422 MOV IESTAT,@CSRA ;ENABLE RX INTERRUPTS.
5503 024556 000207 RTS PC

```

5505
5506
5507
5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540

024560
024560 004567 160540
024564 016704 155720
024570 116724 155464
024574 005204
024576 042702 177400
024602 010224
024604 020427 002712
024610 103402
024612 162704 000004
024616 010467 155666
024622
024622 004736
024624 000207

```

.SBTTL GLOBAL SUBROUTINE - SAVBMP
*****
; * - SAVE BMP CODES ROUTINE
; * 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.
; * BMPCQP - CONTAINS ADDRESS OF NEXT LOCATION IN THE BMP QUEUE.
; * BMPCQB - LABEL AT BASE OF THE BMP CODE QUEUE.
; * BMPCQE - LABEL OF NEXT LOCATION AFTER THE END OF THE BMP QUEUE.
; * TSTNUM - CONTAINS THE NUMBER OF THE CURRENT TEST.
; *
; * OUTPUTS: BMPCQP - 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 ;SAVE CONTENTS OF GPRS R0 THRU R5.
;CALL REGISTER SAVE SUBRT.
JSR R5,PREG05 ;GET THE POINTER TO THE NEXT LOCATION IN QUEUE.
;SAVE THE CURRENT TEST NUMBER ON THE QUEUE.
MOV BMPCQP,R4 ;INCREMENT THE POINTER TO GIVE AN EVEN ADDRESS.
MOVB TSTNUM,(R4) ;CLEAR THE UNWANTED BITS FROM THE BMP CODE.
INC R4 ;SAVE THE BMP CODE ON THE QUEUE.
BIC #177400,R2 ;CHECK IF OVERFLOW WILL OCCUR THE NEXT TIME.
MOV R2,(R4) ;GO SAVE THE POINTER IF WE WILL NOT OVERFLOW.
CMP R4,#BMPCQE ;RESET THE POINTER TO THE LAST LOCATION IN QUE.
HLO 2$ ;SAVE THE POINTER.
SUB #4,R4
2$: MOV R4,BMPCQP
60$: PASS
RTS PC JSR ;RESTORE GPRS.
PC,#(SP) ;RETURN TO PREG05 SUBRT.

```


5542
5543
5544
5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557
5558
5559
5560
5561
5562 024626
024626 004567 160472
5563 024632 012704 000012
5564 024636 004767 172614
5565
5566
5567
5568 024642 012701 000060
5569
5570
5571 024646 012703 052525
5572 024652 005301
5573 024654 016704 155320
5574 024660 010124
5575 024662 010324
5576 024664 020467 155326
5577 024670 103774
5578 024672 032701 000017
5579 024676 001365
5580
5581 024700
024700 004736
5582 024702 000207

```

.SBTTL GLOBAL SUBROUTINE SKPSTS
;*****
;* - SKIP SELFTEST ROUTINE -
;* THIS 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 DUT 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 ;SAVE CONTENTS OF GPRS R0 THRU R5.
;*****
;***** JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV #10,,R4 ;PASS DELAY VALUE OF 10 MILLI-SECONDS.
JSR PC,DELAY ;DELAY FOR 10 MILLI-SECONDS.
;*****
;* WRITE SKIP SELF-TEST CODE (52525) TO ALL THE INDEXED DUT REGISTERS.
;*****
MOV #NUMLNS:BIT05,R1 ;FORM IND.ADR.REG FIELD (PLUS M.R. BIT) WORD.
;THE ABOVE INCLUSION OF THE M.R. BIT IS NECESSARY BECAUSE OF THE
;LACK OF A M.R. BIT WRITE LOCK-OUT ON THE DMU-11.
MOV #52525,R3 ;INITIALISE THE SKIP SELF-TEST CODE.
4$: DEC R1 ;SELECT THE NEXT SET OF DEVICE REGISTERS.
MOV CSRA,R4 ;GET THE ADDRESS OF THE CSR OF THE DUT.
MOV R1,(R4). ;SELECT A BANK OF DUT REGISTERS.
6$: MOV R3,(R4). ;WRITE THE CODE TO A DUT REGISTER.
CMP R4,TXBFCA ;COMPARE POINTER WITH LAST REGISTER ADDRESS.
BLO 6$ ;LOOP IF NOT ALL REGS DONE IN THIS BANK.
BIT #17,R1 ;TEST FOR IND.ADR.REG FIELD DECREMENTED TO 0.
BNE 4$ ;LOOP UNTIL ALL REGISTERS CONTAIN THE CODE.
60$: PASS ;RESTORE GPRS.
;*****
;***** JSR PC,@(SP). ;RETURN TO PREG05 SUBRT.
RTS PC

```

5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629
5630
5631
5632
5633 024704
024704 004567 160414
5634 024710 010067 000264
5635 024714 010167 000262
5636 024720 005067 155556
5637 024724 005067 155554
5638
5639

```

.SBTTL GLOBAL SUBROUTINE - SPLSUP -
*****
- SPLIT SPEED TRANSMISSION/RECEPTION SET-UP -
*****
THIS ROUTINE IS USED TO INITIALISE BOTH THE DUT AND THE
TRANSMISSION/RECEPTION CONTROL PARAMETERS TO THE CORRECT
STATE, PRIOR TO SPLIT SPEED TRANSMISSION/RECEPTION.

INPUTS:  R0 - TX,RX LPR CONTENTS FOR LINES IN GROUP II.
         R1 - TX,RX LPR CONTENTS FOR LINES IN GROUP I.
         R2 - START ADDRESS OF DATA PATTERN TO TRANSMIT.
         R3 - NUMBER OF TIME DATA PATTERN TO BE TX ON LINES IN LINGRP1.
         R4 - NUMBER OF TIME DATA PATTERN TO BE TX ON LINES IN LINGRP2.
         ACTLNS - CONTAINS A BIT MAP OF ALL CURRENTLY ACTIVE LINES.
         LGRP1M - CONTAINS THE BIT MAP OF LINE GROUP I LINES.
         LOPBCK - CONTAINS THE TYPE OF LOOPBACK MODE SELECTED.
         CCB - LABEL AT BASE OF TX/RX CONTROL BLOCK.

OUTPUTS: THE CONTENTS OF THE CONTROL BLOCK ARE DESTROYED.
         THE INDIRECT ADDRESS FIELD OF THE DUT CSR MAY BE DESTROYED.
         THE DUT'S LPR'S AND LNC'S MAY BE MODIFIED.
         THE FOLLOWING POINTERS AND COUNTERS ARE INITIALISED;
         CHCNT,CHRTOT,DPEND,DPLEN,EXCNT,RXCNT,RXDONF,RXPTR,TXCNT,
         TXDONF,TXPTR,TXRXL.

CALLING SEQUENCE: JSR PC,SPLSUP

COMMENTS: THIS ROUTINE SHOULD BE CALLED TWICE DURING THE TESTING OF
          THE SPLIT SPEED CAPABILITIES OF THE DUT.
          SO THAT BOTH LINE GROUPS ARE TESTED ON TRANSMISSION AND
          RECEPTION.
          EG,  R1 - LPR CONTENTS FOR LINES IN LGRP2M, TX=Y, RX=Z BAUD.
              R2 - LPR CONTENTS FOR LINES IN LGRP1M, TX=Z, RX=Y BAUD.
              R3 - REPEAT TX ON LINES IN LINE GROUP 1 = X TIMES.
              R4 - REPEAT TX ON LINES IN LINE GROUP 2 = W TIMES.
              JSR PC,SPLSUP ; DO SET-UP.
              EXECUTE TEST FOR THE ABOVE SET-UP.
              SWAP THE CONTENTS OF R1 AND R2.
              SWAP THE CONTENTS OF R3 AND R4.
              R1 - LPR CONTENTS FOR LINES IN LGRP2M, TX=Z, RX=Y BAUD.
              R2 - LPR CONTENTS FOR LINES IN LGRP1M, TX=Y, RX=Z BAUD.
              R3 - REPEAT TX ON LINES IN LINE GROUP 1 = W TIMES.
              R4 - REPEAT TX ON LINES IN LINE GROUP 2 = X TIMES.
              JSR PC,SPLSUP ; DO SET UP AGAIN.
              EXECUTE TEST AGAIN.

SUBORDINATE ROUTINES CALLED: CONMAP,RXDSBL,RXENBL,TXRINI.
*****
SPLSUP:: SAVE
          JSR ;SAVE CONTENTS OF THE GPR'S R0 THRU R5.
          R5,PREGOS ;CALL REGISTER SAVE SUBRT.
          MOV R0,70H ;SAVE LPR PARAMETER FOR LINE GRP2.
          MOV R1,72H ;SAVE LPR PARAMETER FOR LINE GRP1.
          CLR TXDONF ;CLEAR THE TX DONE FLAGS FOR ALL LINES.
          CLR RXDONF ;CLEAR THE RX DONE FLAGS FOR ALL LINES.

; SET UP THE TRANSMISSION/RECEPTION CONTROL BLOCK TO INITIALISE THE LINES

```

```

5640 ; IN GROUP II.
5641 ;
5642 024730 010067 156166      MOV     R0,CBB           ;SET CONTENTS OF LPR PARAMS IN TX/RX C.BLK.
5643 024734 012700 003124      MOV     #CBB+2,R0       ;GET BASE ADDRESS OF CONTROL BLOCK.
5644 024740 012720 000004      MOV     #4,(R0)         ;LNCTRL PARAMETER, ENABLE RECEIVERS.
5645 024744 010220              MOV     R2,(R0)+        ;START ADDRESS OF DATA PATTERN.
5646 024746 012720 000020      MOV     #16,(R0)+      ;DATA PATTERN LENGTH SET TO 16.
5647 024752 010420              MOV     R4,(R0)+        ;NUMBER OF DATA PATTNS TO TRANSMIT ON LINGRP2.
5648 024754 016710 155212      MOV     ACTLNS,(R0)     ;BIT MAP OF LINES TO INITIALISE.
5649 024760 046720 155252      BIC     LGRP1M,(R0)+    ;CLEAR THE UNWANTED LINES FROM BIT MAP.
5650 024764 116720 155204      MOV     LOPBCK,(R0)+   ;SET LOOPBACK MODE.
5651 024770 005200              INC     R0              ;INCREMENT ADDRESS TO ACCESS NEXT WORD.
5652 024772 012710 000002      MOV     #2,(R0)        ;SET OFFSET FOR EACH TRANSMISSION START TO 2.
5653 ;
5654 ; INITIALISE THE DUT AND THE ASSOCIATED POINTERS AND COUNTERS, TO THE STATE
5655 ; DICTATED BY THE CONTENTS OF THE TX/RX CONTROL BLOCK.
5656 ;
5657 024776 004767 000702      JSR     PC,TXRINI       ;INITIALISE DUT.
5658 ;
5659 ; SET UP CONTROL BLOCK FOR LINES IN GROUP I.
5660 ;
5661 025002 012700 003122      MOV     #CBB,R0         ;GET START ADDRESS OF CONTROL BLOCK.
5662 025006 010120              MOV     R1,(R0)+        ;SET LPR PARAMETER FOR LINES TO RECEIVE DATA.
5663 025010 012720 000004      MOV     #4,(R0)+        ;LNCTRL PARAMETER, ENABLE RECEIVERS.
5664 025014 010220              MOV     R2,(R0)+        ;START ADDRESS OF DATA PATTERN.
5665 025016 012720 000020      MOV     #16,(R0)+      ;DATA PATTERN LENGTH SET TO 16.
5666 025022 010320              MOV     R3,(R0)+        ;NUMBER OF DATA PATTNS TO TRANSMIT ON LINGRP1.
5667 025024 016710 155142      MOV     ACTLNS,(R0)     ;BIT MAP OF LINES TO INITIALISE.
5668 025030 046720 155204      BIC     LGRP2M,(R0)+    ;CLEAR THE UNWANTED LINES FROM BIT MAP.
5669 025034 116720 155134      MOV     LOPBCK,(R0)+   ;SET LOOPBACK MODE.
5670 025040 005200              INC     R0              ;INCREMENT ADDRESS TO ACCESS NEXT WORD.
5671 025042 012710 000002      MOV     #2,(R0)        ;SET OFFSET FOR EACH TRANSMISSION START TO 2.
5672 ;
5673 ; INITIALISE THE DUT AND THE ASSOCIATED POINTERS AND COUNTERS, TO THE STATE
5674 ; DICTATED BY THE CONTENTS OF THE TX/RX CONTROL BLOCK.
5675 ;
5676 025046 004767 000632      JSR     PC,TXRINI       ;INITIALISE DUT.
5677 ;
5678 ;
5679 ; SET-UP THE REQUIRED LPR PARAMETERS NEEDED FOR THE CORRECT RECEPTION OF DATA
5680 ; ON ASSOCIATED IN-ACTIVE LINES.
5681 ;
5682 ;
5683 ; INITIALISE LPR PARAMETERS FOR LINE GROUP 1.
5684 ;
5685 025052 012701 177777      MOV     #MAPLNS,R1      ;SET UP BIT MAP CORRESPONDING TO ALL LINES.
5686 025056 016702 155110      MOV     ACTLNS,R2       ;GET THE ACTIVE (TX) LINE BIT MAP.
5687 025062 005101              COM     R1              ;GENERATE A BIT MAP OF NONE EXISTANT LINES.
5688 025064 005102              COM     R2              ;GENERATE A BIT MAP OF INACTIVE LINES.
5689 025066 040102              BIC     R1,R2           ;CLEAR ANY "NONE EXISTANT" INACTIVE LINES.
5690 025070 046702 155144      BIC     LGRP1M,R2       ;ONLY PASS LGRP1 ASSOCIATED LINE BIT MAP.
5691 025074 010267 156034      MOV     R2,CBMAPA       ;SET UP BIT MAP IN CONTROL BLOCK.
5692 025100 005067 156026      CLR     CBDPNA          ;CLEAR REPEAT TX COUNT IN CONTROL BLOCK.
5693 025104 016767 000072 156010 MOV     #72,CBLPRA      ;SET-UP COMPLEMENTARY LPR PARM FOR LGRP2.
5694 025112 004767 000566      JSR     PC,TXRINI       ;INITIALISE INACTIVE LINES IN LGRP2.
5695 ;
5696 ; INITIALISE LPR PARAMETERS FOR LINE GROUP 2.

```

```

5697
5698 025116 016702 155050      ;-      MOV      ACTLNS,R2      ;GET THE ACTIVE (TX) LINE BIT MAP.
5699 025122 005102              COM      R2              ;GENERATE A BIT MAP OF INACTIVE LINES.
5700 025124 040102              BIC      R1,R2          ;CLEAR ANY NONE EXISTANT INACTIVE LINES.
5701 025126 046702 155104      BIC      LGRP1M,R2      ;ONLY PASS LGRP2 ASSOCIATED LINE BIT MAP.
5702 025132 010267 155776      MOV      R2,CBMAPA      ;SET-UP BIT MAP IN CONTROL BLOCK.
5703 025136 016767 000036 155756  MOV      70$,CBLPRA     ;SET-UP COMPLAMENTARY LPR PARAM FOR LGRP1.
5704 025144 004767 000534      JSR      PC,IXRINI      ;INITIALISE INACTIVE LINES IN LGRP1.
5705
5706      ;+
5707      ; DISABLE RECEIVERS ON ALL LINES TO ENSURE THAT ONLY THE RECEIVERS OF THE
5708      ; ASSOCIATED ACTIVE (TX) LINES ARE ENABLED.(STAGGARED LOOPBACK)
5709      ; RE-ENABLE RECEPTION ON THE CORRECT ASSOCIATED LINES.
5710 025150 012705 177777      ;-      MOV      @MAPLNS,R5      ;SET-UP BIT MAP FOR ALL LINES.
5711 025154 004767 177124      JSR      PC,RXDSBL     ;DISABLE RX ON ALL LINES.
5712
5713      ;+
5714      ; ENABLE RECEIVERS ON ASSOCIATED (RX) LINES.
5715 025160 016705 155006      ;-      MOV      ACTLNS,R5      ;GET ACTIVE (TX) LINE BIT MAP.
5716 025164 004767 172212      JSR      PC,CONMAP     ;GENERATE AN ASSOCIATED (RX) LINE BIT MAP.
5717 025170 004767 177204      JSR      PC,RXENBL     ;ENABLE RECEIVERS ON ASSOCIATED LINES.
5718
5719 025174 004736 60$: PASS      ;RESTORE GRP'S.
                    JSR      PC,@(SP)+      ;RETURN TO PREG05 SUBRT.
5720 025176 000207      RTS      PC
5721 025200 000000      70$:   .WORD 0          ;LOCAL STORAGE OF LPR PARAMETER LGRP2.
5722 025202 000000      72$:   .WORD 0          ;LOCAL STORAGE OF LPR PARAMETER LGRP1.

```

5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741 025204
025204 004567 160114
5742 025210 010146
5743 025212 012746 025220
5744 025216 000002
5745 025220
025220 004736
5746 025222 000207
5747
5748
5749
5750

```

.SBTTL GLOBAL SUBROUTINE - STPSW -
;*****
;* - SET PROCESSOR STATUS WORD -
;* THIS ROUTINE SETS THE PSW TO THE CONTENTS OF R1.
;*
;* INPUTS: R1 - CONTAINS THE NEW PSW SETTINGS
;*
;* OUTPUTS: PSW - SET TO THE CONTENTS OF R1
;*
;* CALLING SEQUENCE: JSR PC,STPSW
;*
;* COMMENTS: USED IN THE DMA ADDRESS TEST TO SET THE PROCESSOR
;* PRIORITY WITHOUT MAKING A CALL TO THE DRS.
;*
;* SUBROUTINES CALLED: NONE.
;*****
STPSW:: SAVE
; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R1,-(SP) ;PUSH THE NEW PSW CONTENTS ONTO THE STACK
MOV ADDR,-(SP) ;PUSH THE NEW PC VALUE ONTO THE STACK
RTI ;LOAD THE NEW PC AND PSW
ADDR: PASS
; JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC ;RETURN

```

5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773 025224
025224 004567 160074
5774
5775
5776
5777
5778
5779
5780 025230 016703 154736
5781 025234 016702 155242
5782 025240 040203
5783 025242 005703
5784 025244 001427
5785
5786
5787
5788
5789
5790
5791 025246 005004
5792 025250 012702 000001
5793 025254 030203
5794 025256 001003
5795 025260 006102
5796 025262 005204
5797 025264 000773
5798 025266 006304
5799 025270 016401 003442
5800 025274 016702 154746
5801 025300 004767 174134
5802 025304 006301
5803
5804
5805
5806
5807 025306 016702 154660

```
.SBTTL GLOBAL SUBROUTINE - TXDONE -
;+ *****
;+ - TRANSMISSION DONE -
;+ THIS SUBROUTINE IS USED IN THE TRANSMISSION/RECEPTION TESTS TO ALLOW
;+ TIME FOR TRANSMISSION TO COMPLETE ON OUTSTANDING LINES.
;+
;+ INPUTS: ACTLNS - CONTAINS BIT MAP OF ALL ACTIVE LINES.
;+ TXDNF - TX DONE FLAGS, SET FOR LINES THAT HAVE SENT ALL CHARS.
;+ CHCNT - TABLE CONTAINING THE NUMBER OF CHARS TO BE TX'D.
;+
;+ OUTPUTS: TRANSMISSION INTERRUPTS ARE DISABLED.
;+
;+ CALLING SEQUENCE: JSR PC, TXDONE
;+
;+ COMMENTS:
;+
;+ SUBORDINATE ROUTINES CALLED: MSLOOP, MUL16U.
;+ *****
TXDONE:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSR R5, PREG05 ;CALL REGISTER SAVE SUBRT.
;+
;+ CHECK IF ALL ACTIVE LINES HAVE COMPLETED TRANSMISSION.
;+ IF ANY HAVE NOT YET COMPLETED, DETERMINE THE TX CHAR COUNT FOR A
;+ LINE THAT HAS OUTSTANDING CHARACTERS TO TRANSMIT. USING THIS VALUE,
;+ CALCULATE THE TIME-OUT VALUE NEEDED AT THE CURRENTLY SELECTED BAUD RATE.
;+
;+
;+ MOV ACTLNS, R3 ;GET THE ACTIVE LINE BIT MAP.
;+ MOV TXDNF, R2 ;GET THE BIT MAP OF LINES THAT HAVE COMPLETED.
;+ BIC R2, R3 ;GENERATE A BIT MAP OF LINES THAT ARE STILL TX.
;+ TST R3 ;CHECK IF ALL LINES HAVE COMPLETED TX.
;+ BEQ 6$ ;GO DISABLE TX INTERRUPTS IF ALL DONE.
;+
;+ FIND A LINE THAT HAS NOT COMPLETED TRANSMISSION.
;+ OBTAIN THE EXPECTED CHARACTER COUNT FOR THAT LINE (WHICH IS THE SAME FOR
;+ ALL OTHER LINES WITH OUTSTANDING TX'S).
;+ CALCULATE TIME-OUT VALUE.
;+
;+
;+ CLR R4 ;CLEAR LINE NUMBER COUNTER.
;+ MOV #1, R2 ;SELECT BIT MAP FOR THE FIRST LINE.
2$: BIT R2, R3 ;SEE IF THIS LINE HAS COMPLETED.
; BNE 4$ ;BRANCH IF THIS LINE HAS NOT COMPLETED TX.
; ROL R2 ;SHIFT THE LINE BIT MAP FOR THE NEXT LINE.
; INC R4 ;INCREMENT THE LINE NUMBER COUNTER.
; BR 2$ ;LOOP TO CHECK THE NEXT LINE.
4$: ASL R4 ;LINE NUMBER X 2 TO OBTAIN OFFSET INTO TABLE.
; MOV CHCNTB(R4), R1 ;GET THE EXPECTED NUMBER OF CHARS TO BE TX'D.
; MOV RXTOUT, R2 ;GET THE CURRENT TIME-OUT VALUE FOR ONE CHAR.
; JSR PC, MUL16U ;(NUMBER OF CHARS TO TX) X (TIME-OUT OF 1 CHAR)
; ASL R1 ;MULTIPLY DELAY TIME BY 2 TO GIVE A SAFE VALUE.
;+
;+ WAIT FOR ALL OUSTANDING TRANSMISSIONS TO COMPLETE OR TIME-OUT.
;+ DISABLE ALL TRANSMISSION INTERRUPTS.
;+
;+ MOV ACTLNS, R2 ;PASS A BIT MAP OF THE BITS TO TEST.
```

5808	025312	010203		MOV	R2,R3		;PASS THE EXPECTED STATE OF THE TXDONF.
5809	025314	012704	002502	MOV	4TXDONF,R4		;PASS THE ADDRESS OF THE WORD TO TEST.
5810	025320	004767	173664	JSR	PC,MSLOOP		;WAIT FOR TIME-OUT OF TX COMPLETION.
5811	025324	004767	000270	6\$: JSR	PC,TXIEO		;DISABLE ALL TX INTERRUPTS.
5812							
5813	025330			60\$: PASS			;RESTORE GPRS.
	025330	004736				JSR	PC,0(SP)+
5814	025332	000207		RTS	PC		;RETURN TO PREG05 SUBRT.

5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869

025334
025334 004567 157764
025340 010500
025342 012701 000001
025346 016702 154642
025352 005202
025354 012703 000020
025360 016704 154650
025364 005005
025366 010477 154606
025372 105712
025374 100001
025376 050105
025400 030100
025402 001402
025404 142712 000200
025410 005204
025412 006301
025414 005303
025416 001363
025420
025420 010566 000014
025424 004736
025426 000207

```
.SBTTL GLOBAL SUBROUTINE - TXDSBL -
;+ *****
;* - TRANSMITTER DISABLE -
;* 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.
;* CSR.A - CONTAINS THE ADDRESS OF THE DUT CSR.
;* IES'AT - 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 IND.ADD.REG FIELD IN THE CSR ARE DESTROYED.
;*
;* CALLING SEQUENCE: JSR PC,TXDSBL
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: NONE.
;-- *****
TXDSBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV R5,R0 ;COPY BIT MAP OF LINES TO DISABLE TRANSMISSION.
MOV #BITC,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
MOV TXAD2A,R2 ;GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
INC R2 ;GET THE ADDRESS OF THE MSBYTE OF TBUFFAD2 REG.
MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER PLUS ONE.
MOV IESTAT,R4 ;GET THE STATES OF THE INT ENABLE BITS.
CLR R5 ;LOG POSSIBLE TX DISABLED ON ALL LINES.
;+
; SELECT EVERY LINE IN TURN, AND LOG THE STATE OF EACH TX.ENABLE BIT.
;-
2$: MOV R4,@CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
TSTB (R2) ;CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
BPL 4$ ;SKIP NEXT INSTRUCTION IF TX.ENABLE CLEAR.
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.
;-
4$: BIT R1,R0 ;CHECK STATE OF DISABLE LINE BIT MAP.
BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
BICB #BIT7,(R2) ;CLEAR TX.ENABLE BIT ON SELECTED LINE.
6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
DEC R3 ;DECREMENT LINE NUMBER.
BNE 2$ ;LOOP TO CHECK NEXT LINE.
60$: PASS R5 ;RESTORE GPRS,EXCEPT
MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
JSR PC,@(SP)+ ;RETURN TO PREG05 SUBRT.
;R5 - PREVIOUS STATES OF ALL TX.ENABLE BITS.
RTS PC
```



```

5871 .SBTTL GLOBAL SUBROUTINE - TXENBL -
5872 ;* *****
5873 ;* - TRANSMITTER ENABLE -
5874 ;* THIS SUBROUTINE IS USED TO ENABLE TRANSMISSION ON SELECTED LINES BY
5875 ;* SETTING THE ASSOCIATED TX.ENABLE BIT ON THE DUT.
5876 ;*
5877 ;* INPUTS: R5 - BIT'S SET CORRESPOND TO LINES ON WHICH TO SET TX.ENABLE.
5878 ;* CSRA - CONTAINS THE ADDRESS OF THE DUT CSR.
5879 ;* IESTAT - CONTAINS THE STATE OF TXIE AND RXIE BITS IN THE CSR.
5880 ;* NUMLNS - EQUATED TO BE THE MAXIMUM NUMBER OF LINES AVAILABLE.
5881 ;* TXAD2A - CONTAINS THE ADDRESS OF THE TBUFFAD2 REGISTER.
5882 ;*
5883 ;* OUTPUTS: R5 - BIT'S SET INDICATE PREVIOUSLY DISABLED LINES.
5884 ;* TBUFFAD2 - THE STATE OF THE TX.ENABLE BIT MAY BE ALTERED.
5885 ;* THE CONTENTS OF THE IND.ADD.REG FIELD IN THE CSR ARE DESTROYED.
5886 ;*
5887 ;* CALLING SEQUENCE: JSR PC,TXENBL
5888 ;*
5889 ;* COMMENTS:
5890 ;*
5891 ;* SUBORDINATE ROUTINES CALLED: NONE.
5892 ;*
5893 ;* *****
5894 TXENBL:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
025430 ; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
025430 004567 157670 JSR
5895 025434 010500 MOV R5,R0 ;COPY BIT MAP OF LINES TO ENABLE.
5896 025436 012701 000001 MOV #BIT0,R1 ;INITIALIZE THE SELECTED LINE BIT MASK.
5897 025442 016702 154546 MOV TXAD2A,R2 ;GET THE ADDRESS OF THE TBUFFAD2 REGISTER.
5898 025446 005202 INC R2 ;GET THE ADDRESS OF THE MSBYTE OF TBUFFAD2 REG.
5899 025450 012703 000020 MOV #NUMLNS,R3 ;GET MAXIMUM LINE NUMBER.
5900 025454 016704 154554 MOV IESTAT,R4 ;GET THE STATES OF THE JNT ENABLE BITS.
5901 025460 005005 CLR R5 ;CLEAR TX.ENABLE BIT LOG OF DISABLED LINES.
5902 ;*
5903 ; SELECT EVERY LINE IN TURN,AND LOG ANY TX.ENABLE BIT THAT IS CLEAR.
5904 ;*
5905 025462 010477 154512 2$: MOV R4,@CSRA ;WRITE TO DUT CSR TO SELECT LINE REGISTERS.
5906 025466 105712 TSTB (R2) ;CHECK STATE OF TX.ENABLE BIT ON SELECTED LINE.
5907 025470 100401 BMI 4$ ;SKIP NEXT INSTRUCTION IF TX.ENABLE SET.
5908 025472 050105 BIS R1,R5 ;LOG TX ENABLE BIT CLEAR FOR SELECTED LINE.
5909 ;*
5910 ; SET TX.ENABLE ON LINES THAT HAVE A CORRESPONDING BIT SET IN THE TX ENABLE
5911 ; LINE BIT MAP.
5912 ;*
5913 025474 030100 4$: BIT R1,R0 ;CHECK STATE OF TX.ENABLE LINE BIT MAP.
5914 025476 001402 BEQ 6$ ;BRANCH IF THIS LINE TO REMAIN UNALTERED.
5915 025500 152712 000200 BISB #BIT7,(R2) ;ENABLE TRANSMISSION ON SELECTED LINE.
5916 025504 005204 6$: INC R4 ;PREPARE TO SELECT REGISTERS FOR NEXT LINE.
5917 025506 006301 ASL R1 ;SHIFT BIT MAP FOR NEXT LINE.
5918 025510 005303 DEC R3 ;DECREMENT LINE NUMBER.
5919 025512 001363 BNE 2$ ;LOOP TO CHECK NEXT LINE.
5920 ;*
5921 025514 010566 000014 60$: PASS R5 ;RESTORE GPRS,EXCEPT
025514 010566 000014 MOV R5,R5SLOT(SP) ;PUT R5 IN STACK SLOT.
025520 004736 JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
5922 ; R5 - LINE BIT MAP CORRESPONDING TO THE
5923 ; PREVIOUS LINES THAT WERE DISABLED.
5924 025522 000207 RTS PC

```

5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
5951
5952
5953
5954
5955
5956
5957 025524
025524 004567 157574
5958 025530 016705 154436
5959 025534 005104
5960 025536 040405
5961
5962
5963
5964 025540 005001
5965
5966
5967
5968 025542 000241
5969 025544 006005
5970 025546 103017
5971
5972
5973
5974
5975
5976 025550 010104
5977 025552 006304
5978 025554 016403 003202
5979 025560 016402 003342
5980
5981

```

.SBTTL GLOBAL SUBROUTINE - TXFRPR -
;+ *****
;+ - TRANSMIT FRAMMING ERROR DATA ROUTINE -
;+ THIS ROUTINE IS USED TO INITIATE DMA MODE TRANSMISSION
;+ IN THE FRAMMING ERROR TEST. IT SENDS A SINGLE CHARACTER DMA BUFFER ON
;+ EACH ACTIVE LINE IN THE BIT MAP, TO CAUSE FUTURE TX INTERRUPTS WHICH
;+ WILL CONTINUE THE TRANSMISSION IF MORE THAN ONE BUFFER IS TO BE SENT.
;+
;+ INPUTS: R4 - CONTAINS THE LINES ON WHICH TX IS TO TAKE PLACE.
;+ ACTLNS - ACTIVE LINES BIT MAP.
;+ BITTBL - LABEL OF TABLE OF WORDS EACH WITH A BIT SET.
;+ CSRA - CONTAINS THE ADDRESS OF THE DUT CSR.
;+ DPENDB - BASE OF THE DATA PATTERN END TABLE (ENTRY PER LINE).
;+ DPLENB - BASE OF THE DATA PATTERN LENGTH TABLE.
;+ IESTAT - PRESERVED STATES OF THE DUT INTERRUPT ENABLE BITS.
;+ NUMLNS - EQUATED TO NUMBER OF LINES ON A DUT.
;+ TXCNTB - LABEL AT BASE OF THE TX CHARACTER COUNTER TABLE.
;+ TXPTRB - LABEL AT BASE OF THE TX DATA PATTERN POINTERS TABLE.
;+
;+ OUTPUTS: CSR - DUT CSR IND.ADR.REG FIELD IS DESTROYED.
;+ TXCNTX - COUNTERS INCREMENTED FOR LINES ON WHICH CHARS SENT.
;+ TXINTF - TX INT FLAGS (BIT SET IF DMA.HO FOUND SET ON LINE).
;+
;+ CALLING SEQUENCE: JSR PC,TXFRPR
;+
;+ COMMENTS: THIS ROUTINE ASSUMES THAT AT LEAST ONE DATA PATTERN SHOULD BE
;+ TRANSMITTED ON EACH ACTIVE LINE.
;+ INTERRUPTS MUST BE DISABLED WHEN CALLING THIS ROUTINE.
;+
;+ SUBORDINATE ROUTINES CALLED: DODMA.
;+ *****
TXFRPR:; SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;+ ;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;+ MOV ACTLNS,R5 ;GET THE ACTIVE LINE BIT MAP.
;+ COM R4 ;GET BIT MAP OF LINES THAT WILL RECEIVE DATA.
;+ BIC R4,R5 ;CLEAR LINES THAT WILL RX FROM TX LINE BIT MAP.
;+
;+ SET UP LOOP WHICH HANDLES ONE LINE PER ITERATION.
;+
;+ CLR R1 ;CLEAR THE LINE NUMBER COUNTER.
;+
;+ ; IF THE LINE IS INACTIVE SKIP TO SELECT THE NEXT LINE.
;+
;+ 2$: CLC ;CLEAR BOOLEAN REGISTER.
;+ ROR R5 ;SHIFT BIT MAP OF LINES TO TX ON INTO BOOL.REG.
;+ BCC 6$ ;DON'T TX ON THIS LINE IF IT IS NOT ACTIVE.
;+
;+ ; LINE IS ACTIVE.
;+ ; INITIATE DMA ON THIS LINE.
;+ ; GET THE DATA PATTERN LENGTH FOR THIS LINE.
;+
;+ MOV R1,R4 ;COPY LINE NUMBER.
;+ ASL R4 ;CALCULATE WORD OFFSET FOR THIS LINE.
;+ MOV DPLENB(R4),R3 ;GET DATA PATTERN LENGTH FOR THIS LINE.
;+ MOV TXPTRB(R4),R2 ;PREPARE TO PASS DATA PATTERN ADR TO DODMA RTN.
;+
;+ ; WRITE DMA PARAMETERS TO THE DUT.

```

```

5982
5983 025564 004767 172122      ; JSR    PC,DODMA
5984 025570 103404              ; BCS    4$          ;SKIP ERROR IF DODMA WAS SUCCESSFUL.
5985
5986                          ; ;+
5987                          ; ; SET THE PROPER BIT OF THE TX INTERRUPT FLAGS TO INDICATE THE LINE ERROR.
5988 025572 056467 002364 154464 ; ; -
5989 025600 000402              ; BIS    BITTBL(R4),TXINTF ;INDICATE THE ERROR.
5990                          ; BR     6$          ;SKIP UPDATING POINTERS AND COUNTERS.
5991                          ; ;+
5992                          ; ; UPDATE THE TX CHARACTER COUNT FOR THIS LINE.
5993 025602 060364 003502      4$:  ADD    R3,TXCNTB(R4) ;ADD THE DATA PATTERN LENGTH TO TX CHAR COUNT.
5994                          ; ;+
5995                          ; ; INCREMENT LINE COUNTER,GOTO NEXT LINE IF NOT DONE.
5996                          ; ; -
5997 025606 005201              6$:  INC    R1          ;INCREMENT THE LINE COUNTER.
5998 025610 005705              ; TST    R5          ;TEST THE TX LINE BIT MAP.
5999 025612 001353              ; BNE    2$          ;LOOP TO SEND CHAR TO ANOTHER LINE IF NOT DONE.
6000
6001 025614 004736              60$: PASS          ;RESTORE GPRS.
6002 025616 000207              ; RTS    PC          JSR    PC,@(SP)+ ;RETURN TO PREG05 SUBRT.

```

```

6004 .SBTTL GLOBAL SUBROUTINE - TXIEO -
6005 ;* *****
6006 ;* - TRANSMITTER INTERRUPT DISABLE -
6007 ;* THIS ROUTINE IS USED TO DISABLE TRANSMITTER INTERRUPTS IN THE DHU11.
6008 ;*
6009 ;* INPUTS: NONE.
6010 ;*
6011 ;* OUTPUTS: THE TX.INT.ENBL BIT IS CLEARED IN THE DUT CSR,
6012 ;* IESTST -CONTAINS THE UPDATED STATUS OF THE TX AND RX INTERRUPT
6013 ;* ENABLE BITS.
6014 ;*
6015 ;* CALLING SEQUENCE: JSR PC,TXIEO
6016 ;*
6017 ;* COMMENTS: THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
6018 ;* THE DUT CSR ARE DESTROYED.
6019 ;*
6020 ;* SUBORDINATE ROUTINES CALLED: NONE.
6021 ;* *****
6022 025620 010046 TXIEO:; MOV RO,-(SP) ;SAVE CONTENTS OF RO ON THE STACK.
6023 025622 GETPRI -(SP) ;SAVE CURRENT PROCESSOR PRIORITY ON THE STACK.
        025622 104440 TRAP C$GPRI
        025624 010046 MOV RO,-(SP)
6024 025626 SETPRI @PRI07 ;IGNORE ANY INTERRUPTS THAT MAY BE GENERATED.
        025626 012700 000340 MOV @PRI07,RO
        025632 104441 TRAP C$SPRI
6025 025634 042767 177677 154372 BIC #177677,IESTAT ;CLEAR TX.INT.ENBL BIT IN IESTAT.
6026 025642 016777 154366 154330 MOV IESTAT,@CSRA ;DISABLE TX INTERRUPTS.
6027 025650 SETPRI (SP)+ ;ENABLE INTERRUPTS TO THE PROCESSOR AGAIN.
        025650 012600 MOV (SP)+,RO
        025652 104441 TRAP C$SPRI
6028 025654 012600 MOV (SP)+,RO
6029 025656 000207 RTS PC ;RESTORE RO.
    
```

6031
6032
6033
6034
6035
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053

025660 052767 040000 154346
025666 042767 137677 154340
025674 016777 154334 154276
025702 000207

```

.SBTTL GLOBAL SUBROUTINE - TXIE1 -
*****
;* TRANSMITTER INTERRUPT ENABLE
;* THIS ROUTINE IS USED TO ENABLE TRANSMITTER INTERRUPTS IN THE DHU11.
;*
;* INPUTS: NONE.
;*
;* OUTPUTS: THE TX.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,TXIE1
;*
;* COMMENTS: THE CONTENTS OF THE INDIRECT ADDRESS REGISTER FIELD IN
;* THE DUT CSR ARE DESTROYED.
;*
;* SUBORDINATE ROUTINES CALLED: NONE.
*****
TXIE1:: BIS @BIT14,IESTAT ;SET TX,INT,ENBL BIT IN IESTAT.
        BIC @137677,IESTAT ;CLEAR ALL BITS EXCEPT TX RX I,E BITS.
        MOV IESTAT,@CSRA ;ENABLE TX INTERRUPTS.
        RTS PC

```

6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111

```

.SBTTL GLOBAL SUBROUTINE - TXRINI
*****
      . TRANSMIT AND RECEIVE INITIALIZATION ROUTINE
      THIS SUBROUTINE PERFORMS THE INITIALIZATION OF THE VARIOUS POINTERS,
      COUNTERS, AND FLAGS WHICH ARE USED DURING THE TRANSMISSION AND
      RECEPTION PORTION OF A TEST. THIS INITIALIZATION IS PERFORMED ON
      THE SPECIFIED LINES ONLY. OTHER LINE VARIABLES REMAIN UNCHANGED.

INPUTS:  CHCNTB - LABEL AT BASE OF LINE CHARACTER COUNT TABLE.
          CHRTOT - MAX # OF CHARS TO RX ON LINES ALREADY INITIALIZED.
          DPENDB - LABEL AT BASE OF LINE DATA PATTERN END TABLE.
          DPLENB - LABEL AT BASE OF LINE DATA PATTERN LENGTH TABLE.
          EXCNTB - LABEL AT BASE ADDRESS OF EXTRA CHAR COUNTERS TABLE.
          IESTAT - PRESENT STATE OF THE RX, IE AND TX, IE BITS.
          NUMLNS - EQUATED TO NUMBER OF LINES ON THE DUT.
          RXCNTB - LABEL AT BASE ADDRESS OF RX CHARACTER COUNTERS TABLE.
          RXPTRB - LABEL AT BASE ADR OF "NEXT RX CHAR" POINTERS TABLE.
          TXCNTB - LABEL AT BASE ADDRESS OF TX CHARACTER COUNTERS TABLE.
          TXPTRB - LABEL AT BASE ADR OF "NEXT TX CHAR" POINTERS TABLE.
          CBB - LABEL AT BASE OF TX/RX CONTROL BLOCK.
          CB CONTENTS - TX/RX CONTROL BLOCK CONTAINS THE FOLLOWING:
                  CBLPRA - DUT LPR CONTENTS.
                  CBLNCA - DUT LNCTRL CONTENTS.
                  CBDPAA - ADDRESS OF BEGINNING OF DATA PATTERN.
                  CBDPLA - LENGTH IN BYTES OF DATA PATTERN.
                  CBDPNA - NUMBER OF DATA PATTERNS TO TRANSMIT.
                  CBMAPA - BIT MAP OF LINES TO BE INITIALIZED.
                  CBLPBA - TYPE OF LOOPBACK TO BE USED FOR TEST.
                  CBOFSA - AMOUNT TO OFFSET EACH TX START IN THE DATA PAT.
          TXRXLB - LABEL AT BASE OF TX/RX LINE ASSOCIATION TABLE.

OUTPUTS: CHCNT - TABLE OF NUMBER OF LINE TX CHARACTERS (INITIALIZED).
          CHRTOT - MAXIMUM NUMBER OF CHARS TO RECEIVE (2 * PAT LENGTH).
          DPEND - TABLE OF DATA PATTERN ENDS (INITIALIZED).
          DPLEN - TABLE OF DATA PATTERN LENGTHS (INITIALIZED).
          DUT LNCTRL - LINE CONTROL REGISTERS (INITIALIZED).
          DUT LPR - LINE PARAMETER REGISTERS (INITIALIZED).
          EXCNT - TABLE OF EXTRA RX CHAR COUNTS (CLRED, SELECTED LINES).
          RXCNT - TABLE OF RX CHARACTER COUNTS (CLRED, SELECTED LINES).
          RXDNF - "RECEPTION DONE" FLAGS (CLEARED FOR SELECTED LINES).
          RXPTR - TABLE OF RECEIVE POINTERS (INITIALIZED).
          TXCNT - TABLE OF TX CHARACTER COUNTERS (CLRED, SELECTED LINES).
          TXDNF - "TRANSMISSION DONE" FLAGS (CLRED FOR SELECTED LINES).
          TXPTR - TABLE OF TRANSMIT POINTERS (INITIALIZED).
          TXRXL - TX/RX LINE ASSOCIATION TABLE (INITIALIZED).

CALLING SEQUENCE: JSR PC, TXRINI

COMMENTS: IF THE CALCULATION OF THE CHRTOT VALUE (2 TIMES THE DATA
          PATTERN LENGTH) RESULTS IN A NUMBER GREATER THAN 64K THEN
          CHRTOT IS INITIALIZED TO 64K - 1.
          THIS ROUTINE WILL NOT FORCE INTERNAL LOOPBACK BASED ON THE
          LOOPBACK TYPE IN CBLPBA. THE USER MUST SET UP CBLNCA CORRECTLY
          TO GET INTERNAL LOOPBACK.

SUBORDINATE ROUTINES CALLED: WTWLNC, WTWLPR.
*****

```

```

6112 025704          TXRINI:: SAVE          ;SAVE CONTENTS OF GPRS R0 THRU R5.
6113 025704 004567 157414          JSR          R5,PREG05          ;CALL REGISTER SAVE SUBRT.
6114          ; SET UP THE LPR AND LNCTRL REGISTERS AS SPECIFIED IN THE TX/RX CONTROL BLOCK.
6115          ;
6116 025710 016705 155220          MOV          CBMAPA,R5          ;GET THE BIT MAP OF SELECTED LINES.
6117 025714 016700 155204          MOV          CBLNCA,R0          ;GET THE NEW LNCTRL CONTENTS.
6118 025720 026727 155212 000001          CMP          CBLPBA,#01          ;CHECK IF INTERNAL LOOPBACK HAS BEEN SELECTED.
6119 025726 001002          BNE          2$          ;SKIP SETTING INT. LOPBCK IN MAINTENANCE FIELD.
6120 025730 052700 000200          BIS          #200,R0          ;SET INTERNAL LOOPBACK IN MAINTENANCE FIELD.
6121 025734 004767 001126 2$: JSR          PC,WTWLNLC          ;SET UP THE LNCTRL REGS FOR SELECTED LINES.
6122 025740 016700 155156          MOV          CBLPRA,R0          ;GET THE NEW LPR CONTENTS.
6123 025744 004767 001146          JSR          PC,WTWLPRL          ;SET UP THE LPR REGISTERS FOR SELECTED LINES.
6124 025750 004767 177454          JSR          PC,TXENBL          ;ENABLE TX FOR ALL SELECTED LINES.
6125          ;
6126          ; SET UP AND BEGIN LOOP WHICH HANDLES ONE LINE PER ITERATION.
6127          ;
6128 025754 005004          CLR          R4          ;CLEAR THE LINE OFFSET.
6129 025756 016705 155144          MOV          CBDPAA,R5          ;INITIALIZE THE TX START ADDRESS VALUE.
6130 025762 016703 155142          MOV          CBDPLA,R3          ;GET THE LENGTH OF THE DATA PATTERN.
6131 025766 060503          ADD          R5,R3          ;CALCULATE END ADDRESS OF THE DATA PATTERN.
6132 025770 036467 002364 155136 4$: BIT          BITTBL(R4),CBMAPA ;CHECK IF THIS LINE IS SELECTED FOR INIT.
6133 025776 001452          BEQ          12$          ;SKIP SET UP IF LINE IS NOT SELECTED.
6134          ;
6135          ; THIS LINE IS SELECTED FOR INITIALIZATION.
6136          ; SET UP PROPER ENTRY IN NUMBER OF CHARS TO TX AND RX TABLE.
6137          ; INCLUDE CHAR COUNT ON THIS LINE IN MAX ALLOWABLE CHAR TOTAL FOR ALL LINES.
6138          ;
6139 026000 016701 155124          MOV          CBDPLA,R1          ;GET THE LENGTH OF THIS LINE'S DATA PATTERN.
6140 026004 016702 155122          MOV          CBDPNA,R2          ;GET THE NUMBER OF PATTERNS TO TX AND RX.
6141 026010 004767 173424          JSR          PC,MUL16U          ;CALCULATE THE TOTAL NUMBER OF CHARS TO TX/RX.
6142 026014 010164 003442          MOV          R1,CHCNTB(R4)          ;SET UP THE NUMBER OF TX/RX CHARS FOR LINE.
6143 026020 060167 154452          ADD          R1,CHRTOT          ;ADD TWICE THE NUMBER OF CHARACTERS TO TX/RX
6144 026024 103403          BCS          6$          ; ON THIS LINE TO THE TOTAL NUMBER OF CHARS
6145 026026 060167 154444          ADD          R1,CHRTOT          ; WHICH WE WILL ALLOW TO BE RECEIVED ON
6146 026032 103003          BCC          8$          ; ALL LINES.
6147 026034 012767 177777 154434 6$: MOV          #-1,CHRTOT          ; SET MAX CHAR TOTAL TO -1 IF OVERFLOW.
6148 026042          8$:
6149          ;
6150          ; SET UP THE DATA PATTERN END AND LENGTH FOR THIS LINE.
6151          ;
6152 026042 016764 155062 003202          MOV          CBDPLA,DPLENB(R4) ;SET UP TX DATA PATTERN LENGTH FOR THIS LINE.
6153 026050 010364 003142          MOV          R3,DPENDB(R4) ;SET UP TX DATA PAT END ADDRESS FOR THIS LINE.
6154          ;
6155          ; SET UP THE TX COUNTER AND CHARACTER POINTER FOR THIS LINE.
6156          ;
6157 026054 005064 003502          CLR          TXCNTB(R4)          ;CLEAR THE TX COUNTER FOR THIS LINE.
6158 026060 010564 003342          MOV          R5,TXPTRB(R4)          ;SET UP THE TX CHAR POINTER FOR THIS LINE.
6159          ;
6160          ; SET UP THE TX/RX LINE ASSOCIATION OFFSET TABLE ENTRY FOR THIS LINE.
6161          ;
6162 026064 010402          MOV          R4,R2          ;SELECT LINE OFFSET FOR NON-STAGGERED LPRK.
6163 026066 026727 155044 000002          CMP          CBLPBA,#02          ;TEST FOR STAGGERED LOOPBACK.
6164 026074 001003          BNE          10$          ;SKIP SETTING STAGGERED LPRK IF NOT.
6165 026076 006202          ASR          R2          ;FORM BYTE OFFSET INTO TABLE FROM TX LINE #.
6166 026100 11#202 005274          MOVB        STGTRB(R2),R2          ;GET THE RX LINE CORRESPONDING WITH TX LINE.
6167 026104 010264 005234 10$: MOV          R2,TXRXLB(R4)          ;LOAD TX TABLE ENTRY WITH RX LINE OFFSET.

```

```

6168
6169
6170
6171
6172 026110 005062 003542
6173 026114 005062 003242
6174 026120 010562 003402
6175
6176
6177
6178 026124 066705 155010
6179 026130 020503
6180 026132 103403
6181 026134 166705 154770
6182 026140 000773
6183
6184
6185
6186 026142 005204
6187 026144 005204
6188
6189
6190
6191 026146 020427 000040
6192 026152 002706
6193
6194 026154
        026154 004736
6195 026156 000207

; *
; SET UP THE RX COUNTERS AND CHARACTER POINTER FOR THE RX LINE WHICH
; IS ASSOCIATED WITH THIS TX LINE.
; -
        CLR    RXCNTB(R2)    ;CLEAR THE RX COUNTER FOR THIS RX LINE.
        CLR    EXCNTB(R2)    ;CLEAR THE EXTRA CHAR COUNTER FOR THIS RX LINE.
        MOV    R5,RXPTRB(R2) ;SET UP THE RX CHAR POINTER FOR THIS RX LINE.
; *
; UPDATE THE TX START POINTER IN PREPARATION FOR THE NEXT LINE.
; -
12$:   ADD    CBOFSA,R5      ;ADD THE TX OFFSET TO THE TX START POINTER.
14$:   CMP    R5,R3         ;COMPARE TX START WITH END OF DATA PATTERN.
        BLO   16$          ;SKIP WRAPAROUND IF START IS BEFORE PAT END.
        SUB   CBDPLA,R5     ;SUBTRACT DATA PATTERN LENGTH FROM START.
        BR   14$          ;LOOP UNTIL START IS WITHIN DATA PATTERN.
; *
; UPDATE THE TX LINE NUMBER OFFSET TO THE NEXT LINE.
; -
16$:   INC    R4
        INC    R4
; *
; TEST FOR DONE HANDLING ALL POSSIBLE LINES ON THE DEVICE.
; -
        CMP    R4,#NUMLNS*2 ;COMPARE OFFSET WITH 2 TIMES MAX # OF LINES.
        BLT   4$          ;LOOP IF NOT ALL LINES DONE.
; *
60$:   PASS
        JSR   PC          ;RESTORE GPRS.
        JSR   PC          ;RETURN TO PREG05 SUBRT.
        RTS   PC

```


6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226

026160
026160 004567 157140
026164
026164 104440
026166 010067 154056
026172
026172 012700 000300
026176 104441
026200 012705 177777
026204 004767 177124
026210 010567 154036
026214
026214 004736
026216 000207

```
.SBTTL GLOBAL SUBROUTINE - TXROFF -
; ** *****
;* - TURN TX AND RX OFF ROUTINE -
;* THIS SUBROUTINE IS USED TO TURN OFF DUT TRANSMISSION AND RECEPTION,
;* THIS ROUTINE ACHIEVES THIS BY BOOSTING PROCESSOR PRIORITY TO 5 TO
;* AVOID RX INTERRUPTS AND BY CLEARING ALL THE DUT TX.ENABLE BITS TO
;* HALT TX (EITHER DMA OR SINGLE CHARACTER TX). THE STATES OF THE
;* TX.ENABLE BITS AND THE PROCESSOR PRIORITY ARE SAVED FOR RESTORATION
;* WHEN TX AND RX ARE RE-ENABLED.
;*
;* INPUTS: MAPLNS - BIT MAP OF ALL POSSIBLE LINES ON THE DUT.
;*
;* OUTPUTS: SAVPRI - SAVED PROCESSOR PRIORITY.
;* SAVTEN - BIT MAP OF TX.ENBL BITS (BIT SET IF TX.ENBL WAS SET).
;*
;* CALLING SEQUENCE: JSR PC, TXROFF
;*
;* COMMENTS:
;*
;* SUBORDINATE ROUTINES CALLED: TXDSBL.
;-- *****
TXROFF:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5, PREG05 ;CALL REGISTER SAVE SUBRT.
GETPRI SAVPRI ;GET THE PRESENT PROCESSOR PRIORITY.
TRAP C$GPRI
MOV R0, SAVPRI
SETPRI @PRI06 ;DISABLE DUT INTERRUPTS.
MOV @PRI06, R0
TRAP C$SPRI
MOV @MAPLNS, R5 ;PREPARE TO DISABLE TX ON ALL DUT LINES.
JSR PC, TXDSBL ;CLEAR ALL DUT TX.ENABLE BITS.
MOV R5, SAVTEN ;PRESERVE THE PREVIOUS TX.ENABLE BIT STATES.
60$: PASS ;RESTORE GPRS.
PC, @ (SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC
```

6228
 6229
 6230
 6231
 6232
 6233
 6234
 6235
 6236
 6237
 6238
 6239
 6240
 6241
 6242
 6243
 6244
 6245
 6246
 6247
 6248 026220
 6249 026224
 6250 026230
 6251 026234
 6252 026242
 6253 026244

004567 157100
 016705 154022
 004767 177174
 016700 154010
 104441
 004736
 000207

```

.SBTTL GLOBAL SUBROUTINE - TXRON -
; ** *****
; * - TURN TX AND RX ON ROUTINE -
; * THIS SUBROUTINE IS USED TO TURN ON DUT TRANSMISSION AND RECEPTION.
; * THIS ROUTINE RESTORES THE DUT TX,ENABLE BITS AND THE PROCESSOR PRIORITY
; * TO THE STATES SAVED BY THE TXROFF ROUTINE.
; *
; * INPUTS: SAVPRI - SAVED PROCESSOR PRIORITY.
; * SAVTEN - BIT MAP OF TX,ENBL BITS (BIT SET IF TX,ENBL WAS SET).
; *
; * OUTPUTS: DUT TX,ENABLE BITS - SET TO SPECIFIED STATES.
; * PROCESSOR PRIORITY - SET TO SPECIFIED PRIORITY.
; *
; * CALLING SEQUENCE: JSR PC, TXRON
; *
; * COMMENTS:
; *
; * SUBORDINATE ROUTINES CALLED: TXENBL.
; - - *****

TXRON:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; R5, PREG05 ;CALL REGISTER SAVE SUBRT.
MOV SAVTEN, R5 ;GET THE SAVED STATES OF THE TX,ENABLE BITS.
JSR PC, TXENBL ;SET THE SPECIFIED TX,ENABLE BITS.
SETPRI SAVPRI ;RESTORE THE PROCESSOR PRIORITY.
MOV SAVPRI, R0
TRAP C$SPRI

60$: PASS ;RESTORE GPRS.
JSR PC, @ (SP)+ ;RETURN TO PREG05 SUBRT.
RTS PC
    
```

6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310

```

.SBTTL GLOBAL SUBROUTINE - TXRREP -
; * *****
; * - REPORT FINAL TX/RX ERRORS ROUTINE -
; * THIS SUBROUTINE REPORTS ERRORS WHICH ARE FOUND AFTER THE COMPLETION
; * OF THE TX, RX, AND VERIFICATION OF DATA PATTERNS. IT REPORTS ERRORS
; * DEALING WITH INCOMPLETE TX OR RX AND WITH DMA_START BITS.
; *
; * INPUTS: ACTLNS - BIT MAP OF ACTIVE DUT LINES.
; * DPLENB - LABEL AT BASE OF THE DATA PATTERN LENGTHS TABLE.
; * ERRMSG - ADDRESS OF PRIMARY ERROR MESSAGE FOR THIS ROUTINE.
; * ERRNBR - ERROR NUMBER OF ERROR REPORTED IN THIS ROUTINE.
; * RXCNTB - LABEL AT BASE OF THE RX CHARACTER COUNTERS TABLE.
; * RXDNF - RECEPTION DONE FLAGS.
; * TXCNTB - LABEL AT BASE OF THE TX CHARACTER COUNTERS TABLE.
; * TXDNF - TRANSMISSION DONE FLAGS.
; * TXINTF - CONTAINS BIT MAP OF LINES WITH DMA_START BIT ERRORS.
; *
; * OUTPUTS: CARRY FLAG - RESTORED TO ITS ENTERING VALUE.
; * ERRBLK - ADDRESS OF THE ERROR REPORTING ROUTINE (DESTROYED).
; * MESSAGES MAY BE PRINTED AT THE OPERATOR CONSOLE.
; *
; * CALLING SEQUENCE: JSR PC, TXRREP
; *
; * COMMENTS: THIS ROUTINE REPORTS ERRORS AT INITIAL ERRNBR THRU
; * INITIAL ERRNBR+2.
; * IF NO LINES FAILED TO COMPLETE THEIR RECEPTION OR FAILED TO
; * COMPLETE THEIR TRANSMISSION OR HAD DMA_START BIT ERRORS
; * THEN NO MESSAGES ARE PRINTED.
; *
; * SUBORDINATE ROUTINES CALLED: CONMAP, ER9005, ER9102, RDMAST, RRXNDN, RTXNDN.
; * - *****

```

```

TXRREP:: SAVE
;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5, PREG05 ;CALL REGISTER SAVE SUBRT.
ROR R3 ;ROTATE CARRY INTO GPR TO SAVE CARRY STATE.
MOV ERRNBR, R4 ;SAVE THE INITIAL ERROR NUMBER VALUE.
MOV ACTLNS, R5 ;GET THE ACTIVE LINES BIT MAP.
JSR PC, RDMAST ;REPORT ANY DMA_START BIT ERRORS.

2$: BIT #BIT06, OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
BNE 2$ ;YES, THEN BRANCH.
TST FERROR ;HAS AN ERROR BEEN DETECTED ?
BNE 60$ ;BRANCH AND EXIT IF IT HAS.

4$: INC ERRNBR ;SELECT INITIAL ERROR NUMBER + 1.
JSR PC, RTXNDN ;REPORT TX NOT COMPLETE IF NECESSARY.

60$: BIT #BIT06, OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
BNE 4$ ;YES, THEN BRANCH.
TST FERROR ;HAS AN ERROR BEEN DETECTED ?
BNE 60$ ;BRANCH AND EXIT IF IT HAS.

8$: INC ERRNBR ;SELECT INITIAL ERROR NUMBER + 2.
JSR PC, CONMAP ;GENERATE AN ASSOCIATED LINE BIT MAP.
JSR PC, RRXNDN ;REPORT RX NOT COMPLETE IF NECESSARY.
MOV R4, ERRNBR ;RESTORE THE INITIAL ERROR NUMBER VALUE.

```

6311 026354 006103
6312 026356 004736
6313 026360 000207

604: ROL R3
PASS
RTS PC

JSR ;ROTATE SAVED CARRY STATE BACK INTO CARRY.
;RESTORE GPRS, THIS ROUTINE PRESERVES THE
PC, @ (SP)+ ;RETURN TO PREG05 SUBRT.
; INITIAL CARRY STATE.

6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338 026362 004567 156736
6339 026362 004567 156736
6340
6341
6342 026366 010204
6343 026370 160104
6344 026372 103403
6345 026374 012701 177777
6346 026400 000442
6347
6348
6349
6350 026402 005004
6351 026404 000241
6352 026406 006001
6353 026410 006004
6354 026412 012700 000020
6355
6356
6357
6358 026416 010246
6359 026420 010346
6360 026422 160403
6361 026424 005602
6362 026426 103402
6363 026430 160102
6364 026432 103003
6365
6366
6367
6368
6369 026434 012603
6370 026436 012602

```

.SBTTL GLOBAL SUBROUTINE - UNSDIV -
;+ *****
;+ - UNSIGNED DIVIDE ROUTINE -
;+ 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 BITS.
;+ R2 - MOST SIGNIFICANT WORD OF THE DIVIDEND, UNSIGNED, 16 BITS.
;+ R3 - LEAST SIGNIFICANT WORD OF THE DIVIDEND, UNSIGNED, 16 BITS.
;+
;+ OUTPUTS: R1 - QUOTIENT, UNSIGNED, 16 BITS (177777 IF OVERFLOW).
;+ CARRY - SUCCESS FLAG, SET IF COMPLETE QUOTIENT FITS IN 16 BITS.
;+
;+ CALLING SEQUENCE: JSR PC,UNSDIV
;+
;+ COMMENTS: IF THE DIVISOR IS 0 THE QUOTIENT IS RETURNED AS ALL ONES
;+ (177777) AND THE CARRY IS CLEAR REGARDLESS OF THE DIVIDEND.
;+
;+ SUBORDINATE ROUTINES CALLED: NONE.
;+
;+ *****
UNSDIV:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;+ JSR R5,PREG05 ;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 0-1,1 ;SET QUOTIENT TO ALL ONES (177777).
;+ BR 60$ ;EXIT WITH CARRY CLEAR.
;+
;+ SET UP COUNTERS AND VARIOUS WORKING GPRS.
;+
;+ CLR R4 ;CLEAR THE LSW OF THE DIVISOR.
;+ CLC ;CLEAR CARRY FOR THE SHIFT OF THE DIVISOR.
;+ ROR R1 ;DIVISOR BY
;+ ROR R4 ;2(UNSIGNED)
;+ MOV 016.,R0 ;SET UP INITIAL SHIFT COUNT TO 16.
;+
;+ THE SUBTRACT AND SHIFT LOOP.
;+
;+ MOV R2,-(SP) ;SAVE MSWORD OF DIVIDEND.
;+ MOV R3,-(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.
;+
;+ MOV (SP)+,R3 ;RESTORE LSWORD OF DIVIDEND.
;+ MOV (SP)+,R2 ;RESTORE MSWORD OF DIVIDEND.

```

```

6371 026440 000401          BR      10$          ;GOTO SHIFT 1 INTO THE QUOTIENT.
6372
6373          ;+ IT WENT, SO WE RESTORE THE STACK AND SHIFT A 0 INTO QUOTIENT (WILL BE
6374          ; COMPLEMENTED LATER).  CARRY IS CLEAR.
6375
6376 026442 012626      8$:  MOV      (SP)+,(SP)+      ;POP THE SAVED DIVIDEND OFF OF THE STACK.
6377
6378          ;+ SHIFT THE RESULT OF THE SUBTRACT ATTEMPT INTO THE QUOTIENT SHIFT REG.
6379          ;-
6380 026444 006105      10$:  ROL      R5          ;SHIFT NEXT BIT INTO THE INVERTED QUOTIENT.
6381 026446 000241          CLC          ;DIVIDE THE
6382 026450 006001          ROR      R1          ; DEVISOR BY
6383 026452 006004          ROR      R4          ; 2 (UNSIGNED).
6384 026454 005300          DEC      R0          ;COUNT THIS SHIFT AND SUBTRACT.
6385 026456 001357          BNE     4$          ;LOOP FOR ANOTHER SHIFT & SUB IF NOT DONE.
6386 026460 005105          COM     R5          ;GET QUOTIENT FROM INVERTED QUOTIENT.
6387
6388          ;+
6389          ; NOW WE EITHER ROUND UP OR LEAVE QUOTIENT ALONE.
6390 026462 000241          CLC          ;CLEAR THE CARRY FOR THE SHIFT OF THE DIVIDEND.
6391 026464 006103          ROL     R3          ;MULTIPLY LSWORD OF DIVIDEND BY 2, MSWORD IS 0.
6392 026466 103402          BCS     12$        ;IF CARRY FROM SHIFT, ROUND UP.
6393 026470 160403          SUB     R4,R3      ;SUBTRACT DIVISOR FROM DIVIDEND.
6394 026472 103403          BCS     14$        ;IF BORROW, DON'T ROUND UP.
6395
6396          ;+
6397          ; ROUND UP, EXTRA SUBTRACT WENT.
6398 026474 005205      12$:  INC     R5          ;INCREMENT THE QUOTIENT BY ONE.
6399 026476 001001          BNE     14$        ;IF NO OVERFLOW, WE LEAVE THE ROUND UP.
6400 026500 005305          DEC     R5          ;DON'T LET ROUNDING CAUSE OVERFLOW.
6401
6402          ;+
6403          ; ALL DONE, PASS QUOTIENT AND EXIT.
6404 026502 010501      14$:  MOV     R5,R1      ;PASS QUOTIENT BACK IN R1.
6405 026504 000261          SEC          ;INDICATE NO OVERFLOW.
6406
6407 026506 010166 000004      60$:  PASS    R1          ;RESTORE GPRS, LEAVE THE FOLLOWING INTACT:
        026506 010166          MOV     R1,R1SLOT(SP) ;PUT R1 IN STACK SLOT.
        026512 004736          JSR     PC,@(SP)+    ;RETURN TO PREG05 SUBRT.
6408          ;R1 - 16 BIT, UNSIGNED QUOTIENT.
6409 026514 000207          RTS     PC          ;CARRY - SET INDICATES NO OVERFLOW (SUCCESS).

```

6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441 026516
026516 004567 156602
6442 026522 016302 005234
6443
6444
6445
6446 026526 016301 003402
6447 026532 005201
6448 026534 020162 003142
6449 026540 103402
6450 026542 166201 003202
6451 026546 010163 003402
6452
6453
6454
6455 026552 016301 003542
6456 026556 005201
6457 026560 001002
6458 026562 012701 177777
6459 026566 010163 003542
6460
6461
6462
6463
6464 026572 016204 003442
6465 026576 020104
6466 026600 103403

```

.SBTTL GLOBAL SUBROUTINE - UPDCHR -
;*****
; - UPDATE CHARACTER POINTERS AND COUNTERS ROUTINE -
; THIS SUBROUTINE UPDATES THE POINTERS AND COUNTERS ASSOCIATED WITH
; THE RECEPTION OF A CHARACTER ON A SPECIFIED LINE. THE RECEIVE CHAR
; POINTER IS SET TO THE NEXT EXPECTED CHARACTER, THE RECEIVE CHAR COUNT
; IS INCREMENTED, AND THE COUNT IS CHECKED TO DETERMINE IF THE RECEPTION
; IS COMPLETE. IF THE RECEPTION IS COMPLETE THE RECEPTION DONE FLAG
; IS SET FOR THE SPECIFIED LINE.
;
; INPUTS: R3 - LINE NUMBER TIMES 2 OF LINE ON WHICH CHAR WAS RECEIVED.
; BIT1BL - LABEL OF TABLE OF WORDS USED TO FORM SINGLE BIT MAPS.
; CHCNTB - BASE OF NUMBER OF CHARS TO TX ON EACH LINE TABLE.
; DPENDB - BASE OF DATA PATTERN END ADDRESSES TABLE.
; DPLENB - BASE OF DATA PATTERN LENGTHS TABLE.
; RXCNTB - BASE OF THE RX CHARACTER COUNTERS TABLE.
; RXPTRB - BASE OF THE RX CHARACTER POINTERS TABLE.
; TXRXLB - BASE OF TX/RX LINE NUMBER ASSOCIATION TABLE.
;
; OUTPUTS: FOLLOWING VARIABLES UPDATED FOR LINE ON WHICH CHAR WAS RECEIVED:
; RXCNT - COUNT OF THE NUMBER OF CHARACTERS RECEIVED ON LINE.
; RXDNF - RX DONE FLAGS WITH BIT0 FOR LINE 0 ... (UPDATED).
; RXPTR - UPDATED TO POINT TO THE NEXT EXPECTED CHAR ON LINE.
;
; CALLING SEQUENCE: JSR PC,UPDCHR
;
; COMMENTS:
;
; SUBORDINATE ROUTINES CALLED: NONE.
;*****
UPDCHR:: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
; JSR R5,PREG05 ;CALL REGISTER SAVE SUBRT.
MOV TXRXLB(R3),R2 ;GET TX LINE NUMBER OFFSET FOR THIS RX LINE.
;
; UPDATE THE RX DATA POINTER WITH WRAPAROUND AT THE END OF THE DATA PATTERN.
;
; MOV RXPTRB(R3),R1 ;GET THE RX DATA POINTER FROM THE RX PTR TABLE.
; INC R1 ;INCREMENT THE RX POINTER VALUE BY 1.
; CMP R1,DPENDB(R2) ;CMP RX PTR VALUE WITH ADR OF END OF DATA PAT.
; BLO 2$ ;SKIP WRAPPING RX PTR AROUND IF NOT AT END.
; SUB DPLENB(R2),R1 ;WRAP RX PTR AROUND TO START OF DATA PATTERN.
2$: MOV R1,RXPTRB(R3) ;UPDATE THE RX POINTER WITH THE NEW VALUE.
;
; UPDATE THE RX CHARACTER COUNT WITH OVERFLOW DETECTION.
;
; MOV RXCNTB(R3),R1 ;GET THE RX CHARACTER COUNT.
; INC R1 ;INCREMENT THE RX CHAR COUNT VALUE BY 1.
; BNE 4$ ;SKIP SETTING COUNT TO MAX IF NO OVERFLOW.
; MOV #1,R1 ;SET RX CHAR COUNT VALUE TO MAX VALUE.
4$: MOV R1,RXCNTB(R3) ;UPDATE THE RX CHAR COUNT WITH NEW VALUE.
;
; CHECK FOR RX COMPLETION ON THIS LINE.
; IF RX IS COMPLETE ON THIS LINE, SET THE CORRECT RX DONE FLAG.
;
; MOV CHCNTB(R2),R4 ;GET THE NUMBER OF TX CHARS IN COMPLETE TX.
; CMP R1,R4 ;COMPARE RX CHAR COUNT WITH NUMBER OF TX CHARS.
; BLO 60$ ;EXIT ROUTINE IF NOT ALL CHARS RECEIVED.

```

```
6467 026602 056367 002364 153674      BIS   BITTBL(R3),RXDNF      ;SET THE RX DONE FLAG FOR THIS LINE.  
6468  
6469 026610          60$: PASS          ;RESTORE GPRS.  
      026610 004736          JSR   PC,@(SP)+          ;RETURN TO PREG05 SUBRT.  
6470 026612 000207      RTS   PC
```


6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507 026614
6508 026614 004567 156504
6509 026620 005067 153652
6510 026624 005067 153434
6511 026630 005067 153646
6511 026634 005067 153644
6512
6513
6514
6515 026640 010167 154256
6516 026644 012701 003122
6517 026650 005201
6518 026652 005201
6519 026654 012721 000004
6520 026660 010221
6521 026662 010321
6522 026664 010421
6523 026666 016721 153500
6524 026672 032767 000004 153274
6525 026700 001404
6526 026702 012702 000001
6527 026706 110221

```

.SBTTL GLOBAL SUBROUTINE - VANSUP -
; * *****
; * - TRANSMISSION / RECEPTION SET-UP ROUTINE -
; *
; * THIS ROUTINE IS USED TO INITIALISE BOTH THE DUT AND THE
; * TRANSMISSION/RECEPTION CONTROL PARAMETERS TO THE CORRECT
; * STATE, PRIOR TO A SINGLE CHARACTER OR DMA TRANSMISSION,
; * RECEPTION TEST.
; *
; * INPUTS: R1 - TX, RX LPR CONTENTS.
; * R2 - START ADDRESS OF DATA PATTERN TO TRANSMIT.
; * R3 - LENGTH OF DATA PATTERN.
; * R4 - NUMBER OF PATTERNS TO TRANSMIT.
; * ACTLNS - CONTAINS A BIT MAP OF ALL CURRENTLY ACTIVE LINES.
; * LOPBCK - CONTAINS THE TYPE OF LOOPBACK MODE SELECTED.
; * CBB - LABEL AT BASE OF TX/RX CONTROL BLOCK.
; *
; * OUTPUTS: THE CONTENTS OF THE TX/RX CONTROL BLOCK (CCB) ARE DESTROYED.
; * THE INDIRECT ADDRESS FIELD OF THE DUT CSR MAY BE DESTROYED.
; * THE DUT'S LPR'S AND LNC'S MAY BE MODIFIED.
; * THE FOLLOWING POINTERS AND COUNTERS ARE INITIALISED;
; * CHCNT,CHRTOT,OPEND,DPLEN,EXCNT,RXCNT,RXPTR,TXCNT,
; * TXPTR,TXRXL.
; * CHRTOT, RXDONF, TXDONF AND TXINTF ARE CLEARED.
; *
; * CALLING SEQUENCE: JSR PC,VANSUP
; *
; * COMMENTS: MODEM LOOPBACK MODE IS INHIBITED IF IT HAS BEEN SELECTED
; * VIA HARDWARE P-TABLE QUESTIONS, AND INTERNAL LOOPBACK MODE
; * IS FORCED TO TAKE PLACE.
; *
; * SUBORDINATE ROUTINES CALLED: CONMAP,RXENBL,IXRINI.
; * - - *****
VANSUP:: SAVE
; SAVE CONTENTS OF THE GPR'S R0 THRU R5.
; R5,PREG05 ;CALL REGISTER SAVE SUBRT.
JSR
;CLEAR TOTAL RECEIVED CHAR COUNTER.
CLR CHRTOT
;CLEAR FLAGS USED TO LOG DMA H.OVER ERRORS.
CLR TXINTF
;CLEAR THE TX DONE FLAGS.
CLR TXDONF
;CLEAR THE RX DONE FLAGS.
CLR RXDONF
; *
; * SET UP THE TRANSMISSION/RECEPTION CONTROL BLOCK TO THE DESIRED STATE.
; *
; *
; * MOV R1,CBB ;SET CONTENTS OF LPR PARAMS IN TX/RX C.BLK.
; * MOV @CBB,R1 ;GET BASE ADDRESS OF CONTROL BLOCK.
; * INC R1 ;INCREMENT ADDRESS FOR NEXT WORD
; * INC R1 ;INITIALISE THE FOLLOWING IN THE CNTRL.BLK:
; * MOV #4,(R1)+ ; LNCTRL PARAMETER, ENABLE RECEIVERS.
; * MOV R2,(R1)+ ; START ADDRESS OF DATA PATTERN.
; * MOV R3,(R1)+ ; DATA PATTERN LENGTH.
; * MOV R4,(R1)+ ; NUMBER OF DATA PATTERNS TO TRANSMIT.
; * MOV ACTLNS,(R1)+ ; BIT MAP OF LINES TO INITIALISE.
; * BIT @BIT2,LOPBCK ;TEST IF MODEM LOOPBACK MODE HAS BEEN SELECTED.
; * BEQ 2$ ;DONT SELECT INTERNL LOPBCK IF STAGRD OR LOCAL.
; * MOV #1,R2 ;FORCE INTERNAL LOOPBACK MODE TO BE SELECTED.
; * MOV# R2,(R1)+ ;INITIALISE LOOPBACK MODE IN CONTROL BLOCK.

```


6563
6564
6565
6566
6567
6568
6569
6570
6571
6572
6573
6574
6575
6576
6577
6578
6579
6580
6581
6582
6583
6584
6585
6586
6587
6588
6589
6590

```

.SBTTL 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.
; * MSLCNT.
; *
; * 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:
; *
; * SUBORDINATE ROUTINES CALLED: MSLGET.
*****

```

6591 027012
027012 004567 156306
6592 027016 010204
6593 027020 010102
6594 027022 042701 170000
6595 027026 042702 007777
6596 027032 000302
6597 027034 006202
6598 027036 006202
6599 027040 006202
6600 027042 016202 002364
6601 027046 010203
6602 027050 004767 172020
6603
6604 027054 010002
6605 027056
027056 010266 000006
027062 004736
6606
6607 027064 000207

```

WAIBIS: SAVE ;SAVE CONTENTS OF GPRS R0 THRU R5.
;R5,PREG05 ;CALL REGISTER SAVE SUBRT.
;SET UP THE ADDRESS PARAMETER FOR MSLGET.
MOV R2,R4
MOV R1,R2
BIC #170000,R1 ;SEPERATE DELAY COUNT OUT OF PASSED PARAMETER.
BIC #7777,R2 ;SEPERATE LINE NUMBER FIELD OF PASSED PARAM.
SWAB R2 ;PUT LINE NUMBER FIELD IN LSBYTE.
ASR R2 ;SHIFT THE LINE NUMBER FIELD INTO THE PROPER
; POSITION TO USE IT AS A WORD TABLE OFFSET
ASR R2 ; FOR THE TABLE LOOKUP OF THE LINE BIT MAP.
ASR R2 ;GET BIT MAP OF LINE TO TEST FROM TABLE.
MOV BITTBL(R2),R2 ;INDICATE THAT THE BIT SHOULD BE SET.
MOV R2,R3 ;WAIT FOR THE BIT TO BE SET WITHIN TIME-OUT.
JSR PC,MSLGET ; CARRY IS CORRECT UPON MSLGET RETURN.
;PASS LAST VALUE READ AS OUTPUT PARAMETER.
MOV R0,R2
PASS R2 ;RESTORE GPRS, EXCEPT THE FOLLOWING:
MOV R2,R25LOT(SP) ;PUT R2 IN STACK SLOT.
JSR PC,@(SP) ;RETURN TO PREG05 SUBRT.
; R2 LAST VALUE READ LOOKING FOR CONDITION.
; CARRY - SUCCESS FLAG (SET IF BIT FOUND SET).
RTS PC

```

```

6609 .SBTTL GLOBAL SUBROUTINE - WTWLNC
6610 ;* *****
6611 ;* LINE CONTROL REGISTER SETUP ROUTINE
6612 ;* THIS SUBROUTINE IS USED TO SET THE DEVICE UNDER TEST (DUT) LINE
6613 ;* CONTROL REGISTERS (LNCTRL) TO THE SPECIFIED STATE. ONLY THE LNCTRLS
6614 ;* FOR THE SPECIFIED LINES ARE ALTERED.
6615 ;*
6616 ;* INPUTS: R0 - NEW LINE PARAMETERS.
6617 ;* R5 - BIT MAP OF LINES TO BE ALTERED.
6618 ;* CSRA - CONTAINS ADDRESS OF THE DUT CSR.
6619 ;* IESTAT - CONTAINS THE CURRENT STATE OF THE TX AND RX INTERRUPT
6620 ;* ENABLE BITS IN THE CSR.
6621 ;* LNCTRA - CONTAINS ADDRESS OF THE DUT LNCTRL REGISTERS.
6622 ;*
6623 ;* OUTPUTS: LNCTRL - SPECIFIED DUT LINE CONTROL REGISTERS ARE ALTERED.
6624 ;*
6625 ;* CALLING SEQUENCE: JSR PC,WTWLNC
6626 ;*
6627 ;* COMMENTS:
6628 ;*
6629 ;* SUBORDINATE ROUTINES CALLED: ALTFLD.
6630 ;* *****
6631
6632 027066 WTWLNC:: SAVE JSR ;SAVE CONTENTS OF GPRS R0 THRU R5.
        027066 004567 156232 R5,PREG05 ;CALL REGISTER SAVE SUBRT.
6633 ;*
6634 ;* SET UP THE PARAMETERS FOR THE CALL TO ALTFLD.
6635 ;*
6636 027072 016701 153112 MOV LNCTRA,R1 ;SET UP THE REGISTER ADDRESS PARAMETER.
6637 027076 010002 MOV R0,R2 ;SET UP THE DESIRED REGISTER CONTENTS.
6638 027100 010503 MOV R5,R3 ;SET UP THE BIT MAP OF LINES TO ALTER.
6639 027102 012704 177777 MOV 0-1,R4 ;SELECT ALL REGISTER BITS TO BE ALTERED.
6640 ;*
6641 ;* CALL THE SUBROUTINE WHICH ALTERS THE REGISTER CONTENTS.
6642 ;*
6643 027106 004767 166706 JSR PC,ALTFLD ;ALTER THE REGISTER CONTENTS.
6644 ;*
6645 027112 004736 60$: PASS ;RESTORE GPRS.
        027112 004736 JSR PC,(SP)+ ;RETURN TO PREG05 SUBRT.
6646 027114 000207 RTS PC
    
```

```

6648          ,SBTTL GLOBAL SUBROUTINE - WTWLPR -
6649          ;* *****
6650          ;* - LINE PARAMETER REGISTER SETUP ROUTINE
6651          ;* THIS SUBROUTINE IS USED TO SET THE DEVICE UNDER TEST (DUT) LINE
6652          ;* PARAMETER REGISTERS (LPR) TO THE SPECIFIED STATE, ONLY THE LPRS FOR
6653          ;* THE SPECIFIED LINES ARE ALTERED.
6654          ;*
6655          ;* INPUTS:      R0 - NEW LINE PARAMETERS.
6656          ;*              R5 - BIT MAP OF LINES TO BE ALTERED.
6657          ;*              CSRA - CONTAINS ADDRESS OF THE DUT CSR.
6658          ;*              IESTAT - CONTAINS THE CURRENT STATE OF THE TX AND RX INTERRUPT
6659          ;*              ENABLE BITS IN THE CSR.
6660          ;*              LPRA - CONTAINS ADDRESS OF THE DUT LPR.
6661          ;*
6662          ;* OUTPUTS:     LPR - SPECIFIED DUT LINE PARAMTER REGISTERS ARE ALTERED.
6663          ;*
6664          ;* CALLING SEQUENCE:  JSR      PC,WTWLPR
6665          ;*
6666          ;* COMMENTS:
6667          ;*
6668          ;* SUBORDINATE ROUTINES CALLED:  ALTFLD.
6669          ;* - *****
6670
6671 027116      WTWLPR:: SAVE                ;SAVE CONTENTS OF GPRS R0 THRU R5.
        027116 004567 156202                JSR      R5,PREG05 ;CALL REGISTER SAVE SUBRT.
6672
6673          ;*
6674          ;* SET UP THE PARAMETERS FOR THE CALL TO ALTFLD.
6675          ;* -
6675 027122      016701 153056                MOV     LPRA,R1    ;SET UP THE REGISTER ADDRESS PARAMETER.
6676 027126      010002                MOV     R0,R2     ;SET UP THE DESIRED REGISTER CONTENTS.
6677 027130      010503                MOV     R5,R3     ;SET UP THE BIT MAP OF LINES TO ALTER.
6678 027132      012704 177777                MOV     0-1,R4   ;SELECT ALL REGISTER BITS TO BE ALTERED.
6679
6680          ;*
6681          ;* CALL THE SUBROUTINE WHICH ALTERS THE REGISTER CONTENTS.
6682          ;* -
6682 027136      004767 166656                JSR     PC,ALTFLD ;ALTER THE REGISTER CONTENTS.
6683
6684 027142      60$: PASS
        027142 004736                ;RESTORE GPRS.
6685 027144      000207                JSR     PC,0(SP)  ;RETURN TO PREG05 SUBRT.
        RTS     PC
    
```

```

6687 .SBTTL INTERRUPT SERVICE ROUTINE - CLKINT -
6688 ;* *****
6689 ;* THIS ROUTINE IS EXECUTED CLKHRZ TIMES PER SECOND. IT DECREASES THE
6690 ;* TWO TIMER COUNTERS DOWN TO ZERO.
6691 ;*
6692 ;* INPUTS: TIMER1 - TIMER COUNTER #1.
6693 ;* TIMER2 - TIMER COUNTER #2.
6694 ;* TIMER3 - TIMER COUNTER FOR CALL OF BREAK MACRO.
6695 ;*
6696 ;* OUTPUTS: THE 2 TIMER COUNTERS ARE DECREMENTED IF THEY ARE NOT ZERO.
6697 ;*
6698 ;* CALLING SEQUENCE: PUT #CLKINT IN THE CLOCK INTERRUPT VECTOR SLOT.
6699 ;* PUT THE DESIRED TIME PERIOD (SECONDS TIMES CLKHRZ) IN
6700 ;* EITHER TIMER1 OR TIMER2 AND POLL THE RESPECTIVE TIMER
6701 ;* COUNTER TO DETECT ITS GOING TO 0 ON TIME-OUT.
6702 ;*
6703 ;* COMMENTS: THE 2 COUNTERS WILL NOT WRAPAROUND BUT WILL STOP AT 0. THIS
6704 ;* ALLOWS THE DETECTION OF A TIME-OUT ANY TIME AFTER THE TIME-OUT
6705 ;* HAS OCCURRED UNTIL THE TIMER COUNTER IS SET TO ANOTHER VALUE.
6706 ;*
6707 ;* SUBORDINATE ROUTINES CALLED: NONE.
6708 ;* *****
6709
6710 027146 005767 153126 CLKINT:: TST TIMER1 ;CHECK FOR TIMER1 AT ZERO.
6711 027152 001402 BEQ 2$ ;BRANCH TO LEAVE IT AT ZERO IF IT IS ZERO.
6712 027154 005367 153120 DEC TIMER1 ;DECREMENT TIME COUNT.
6713 027160 005767 153116 2$: TST TIMER2 ;CHECK FOR TIMER2 AT ZERO.
6714 027164 001402 BEQ 4$ ;BRANCH TO LEAVE IT ALONE IF IT'S ALREADY ZERO.
6715 027166 005367 153110 DEC TIMER2 ;DECREMENT TIME COUNT.
6716 027172 005367 153106 4$: DEC TIMER3 ;DECREMENT THE BREAK COUNT.
6717 027176 001006 BNE 60$ ;EXIT IF NOT TIME TO CALL BREAK.
6718 027200 016767 153102 153076 MOV BCOUNT,TIMER3 ;SET UP TIME TILL NEXT BREAK.
6719 027206 010046 MOV RO, -(SP) ;SAVE CONTENTS OF RO FROM BREAK MACRO.
6720 027210 BREAK ;CHECK FOR OPERATOR CONTROL/C.
        027210 104422 TRAP C$BRK
6721 027212 012600 MOV (SP)+,RO ;RESTORE CONTENTS OF RO.
6722 027214 000002 RTI
    
```

6724
6725
6726
6727
6728
6729
6730
6731
6732
6733
6734
6735
6736
6737
6738
6739
6740
6741
6742
6743
6744
6745
6746
6747
6748
6749
6750
6751
6752
6753
6754
6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774
6775
6776
6777
6778
6779
6780

027216 010246
027220 017702 152756
027224 100054
027226 026727 153464 000100
027234 103402
027236 004767 172562
027242 010277 153446
027246 062767 000002 153440
027254 026727 153434 003120
027262 103403
027264 012767 002720 153422
027272 005267 153420
027276 026727 153414 000030
027304 002745
027306 005767 153174
027312 100413
027314 010546
027316 012705 177777
027322 004767 176006
027326 010567 152730

```
.SBTTL INTERRUPT SERVICE ROUTINE - RXCHRS -
;*****
; - DMA RECEIVE INTERRUPT SERVICE ROUTINE -
; THIS ROUTINE EXECUTES IN RESPONSE TO AN INTERRUPT CAUSED BY THE DUT
; RX.DATA,AVAIL BIT BECOMING ACTIVE. THIS ROUTINE READS CHARACTERS FROM
; THE DUT RECEIVE CHARACTER FIFO AND DEPOSITS THEM INTO THE RECEIVE
; BUFFER IN MEMORY. IF THE NUMBER OF CHARACTERS IN THE RECEIVE BUFFER
; EXCEEDS A SPECIFIED THRESHOLD, TRANSMISSION IS HALTED (BY CLEARING ALL
; DUT TX,ENABLE BITS) AND IF THE RECEIVE BUFFER IS FULL RECEPTION IS
; HALTED (BY DISABLING RX INTERRUPTS). THE ROUTINE EXITS IF THE RECEIVE
; BUFFER BECOMES FULL OR IF A CHARACTER IS READ FROM THE FIFO WITH THE
; DATA,VALID BIT CLEAR.
;
; INPUTS: RBUFA - CONTAINS ADDRESS OF THE DUT RX CHARACTER FIFO.
;          RXBCNT - RX BUFFER CHARACTER COUNT.
;          RXBDTX - EQUATED TO RX BUFFER LEVEL AT WHICH TO DISABLE TX.
;          RXBEND - LABEL AFTER END OF THE RX BUFFER AREA IN MEMORY.
;          RXBFUL - EQUATED TO THE CAPACITY OF THE RX BUFFER.
;          RXBIPT - POINTER TO NEXT AVAILABLE INPUT SLOT OF RX BUFFER.
;          RXBSTA - LABEL AT START OF RX BUFFER AREA IN MEMORY.
;
; OUTPUTS: RXBIPT - UPDATED TO POINT TO NEXT INPUT SLOT OF RX BUFFER.
;          RXBCNT - RX BUFFER CHARACTER COUNT (INCREMENTED).
;          TXENBM - MAP OF PREVIOUS DUT TX,ENABLE STATES.
;          CARRY - "SUCCESS" FLAG (SET IF BUFFER IS NOT FULL).
;
; CALLING SEQUENCE: PUT THE ADDRESS OF THE LABEL RXCHRS IN THE VECTOR
; LOCATION.
;
; COMMENTS: IF THE RX BUFFER IS FULL UPON ENTRY, THIS ROUTINE ABORTS THE
; PROGRAM.
;
; SUBORDINATE ROUTINES CALLED: RXIE0,TXDSBL.
;*****
RXCHRS: MOV R2, -(SP) ;SAVE CONTENTS OF GPR R2.
2$: BPL 60$ ;READ A CHARACTER FROM THE DUT RX FIFO.
;EXIT THE ROUTINE IF THE DATA,VALID BIT IS CLR.
;
; CMP RXBCNT, RXBFUL ;COMPARE BUFFER COUNT WITH BUFFER CAPACITY.
; BLO 4$ ;SKIP ABORT IF BUFFER IS NOT FULL.
; JSR PC, OOPS ;ABORT, MUST BE A PROGRAM BUG.
4$: MOV R2, RXBIPT ;PUT THE CHAR IN THE BUFFER.
; ADD #2, RXBIPT ;UPDATE POINTER TO THE NEXT BUFFER SLOT.
; CMP RXBIPT, RXBEND ;CHECK IF POINTER SHOULD WRAP AROUND.
; BLO 6$ ;SKIP WRAPAROUND IF POINTER IS NOT AT END.
; MOV #RXBSTA, RXBIPT ;WRAP INPUT POINTER AROUND.
;
; 6$: INC RXBCNT ;COUNT THIS CHARACTER AS BEING IN THE BUFFER.
; CMP RXBCNT, RXBDTX ;CHECK FOR BUFFER AT DISABLE TX LEVEL.
; BIT 2$ ;SKIP DISABLING TX IF BUFFER LEVEL NOT CORRECT.
; TST TXDBLF ;CHECK STATE OF TX DISABLE FLAG.
; BMI 8$ ;BRANCH IF TRANSMISSION ALREADY DISABLED.
; MOV R5, -(SP) ;SAVE THE VALUE OF GPR R5.
; MOV #MAPLNS, R5 ;SPECIFY THAT ALL LINES SHOULD BE AFFECTED.
; JSR PC, TXDSBL ;CLEAR THE TX ENABLES FOR ALL LINES.
; MOV R5, TXENBM ;SAVE PREVIOUS TX ENABLE STATES IN STORAGE.
```

```
6781 027332 012605          MOV    (SP)+,R5      ;RESTORE GPR R5.
6782 027334 012767 100000 153144  MOV    @BIT15,TXDBLF ;PREVENT TX FROM BEING DISABLED AGAIN.
6783
6784 027342 026727 153350 000100 8$:  CMP    RXBCNT,@RXBFUL ;CHECK FOR BUFFER FULL CONDITION.
6785 027350 103723          BLO    2$           ;LOOP TO READ ANOTHER CHAR IF BUFFER NOT FULL.
6786
6787 027352 004767 175116          JSR    PC,RXIE0     ;BUFFER IS FULL, DISABLE RX INTERRUPTS.
6788
6789 027356 012602          60$:  MOV    (SP)+,R2     ;RESTORE R2 TO ITS SAVED VALUE.
6790 027360 000002          RTI
```



```

6792 .SBTTL TRAP SERVICE ROUTINE - TP4BRT -
6793 ;*****
6794 ;* BUS TIME-OUT TRAP (004 TRAP) SERVICE ROUTINE -
6795 ;* THIS ROUTINE IS USED DURING THE DMA ADDRESS TEST.
6796 ;* IT DETERMINES IF THE 004 TRAP WAS CAUSED BY AN "EXPECTED" ERROR OR
6797 ;* NOT BY EXAMINING THE RETURN PC VALUE ON THE STACK. IF THE TRAP IS
6798 ;* UNEXPECTED, THIS ROUTINE JUMPS TO THE NORMAL DIAGNOSTIC SUPERVISOR
6799 ;* 004 TRAP HANDLING ROUTINE.
6800 ;*
6801 ;* INPUTS: SP - POINTS TO THE PC WHERE THE TRAP OCCURED.
6802 ;* TRPAD2 - LABEL AT THE ADDRESS WHERE "EXPECTED" TRAPS OCCUR.
6803 ;* TP4FLG - 004 TRAP FLAGS.
6804 ;*
6805 ;* OUTPUTS: TP4FLG - BIT 15 IS SET IF "EXPECTED" TRAP OCCURED.
6806 ;*
6807 ;* CALLING SEQUENCE: PUT ADDRESS POINTED TO BY TP4BRT IN 004 VECTOR.
6808 ;* OCCURENCE OF 004 TRAP VECTORS TO THIS ROUTINE.
6809 ;*
6810 ;* COMMENTS: ANY 004 TRAP WHICH OCCURS AT AN ADDRESS OTHER THAN THAT LABELED
6811 ;* TRPAD2 WILL BE HANDLED BY THE NORMAL 004 TRAP SERVICE ROUTINE.
6812 ;* THIS ROUTINE IS USED IN CONJUNCTION WITH CKTRPB SUBROUTINE.
6813 ;*
6814 ;* SUBORDINATE ROUTINES CALLED: NONE.
6815 ;*****
6816
6817 027362 021627 017320 TP4BRT:: CMP (SP),@TRPAD2 ;COMPARE EXPECTED ADDR WITH TRAP RET PC.
6818 027366 001402 BEQ 2$ ;IF THEY MATCH, CONTINUE THIS ROUTINE.
6819 027370 000177 152662 JMP @TP4VEC ;IF NOT, JUMP TO NORMAL 004 TRAP SERVICE RTN.
6820 027374 052767 100000 152652 2$: BIS @BIT15,TP4FLG ;SET THE 004 TRAP OCCURED FLAG.
6821 027402 000002 RTI ;ALL DONE, GO BACK TO THE TEST.

```

```

6823 .SBTTL GLOBAL TRAP SERVICE ROUTINE - TP4RTN -
6824 ;*****
6825 ;* BUS TIME-OUT TRAP (004 TRAP) SERVICE ROUTINE -
6826 ;* THIS ROUTINE DETERMINES IF THE 004 TRAP WAS CAUSED BY
6827 ;* AN "EXPECTED" ERROR OR NOT BY EXAMINING THE RETURN PC VALUE ON THE
6828 ;* STACK. IF THE TRAP IS UNEXPECTED, THIS ROUTINE JUMPS TO THE NORMAL
6829 ;* DIAGNOSTIC SUPERVISOR 004 TRAP HANDLING ROUTINE.
6830 ;*
6831 ;*
6832 ;* INPUTS: SP - POINTS TO THE PC WHERE THE TRAP OCCURED.
6833 ;* ADRPTR - LABEL AT THE ADDRESS WHERE "EXPECTED" TRAPS OCCUR.
6834 ;* TP4FLG - 004 TRAP FLAGS.
6835 ;*
6836 ;* OUTPUTS: TP4FLG - BIT 15 IS SET IF "EXPECTED" TRAP OCCURED.
6837 ;*
6838 ;* CALLING SEQUENCE: PUT ADDRESS POINTED TO BY TP4RTN IN 004 VECTOR.
6839 ;* OCCURENCE OF 004 TRAP VECTORS TO THIS ROUTINE.
6840 ;*
6841 ;* COMMENTS: ANY 004 TRAP WHICH OCCURS AT AN ADDRESS OTHER THAN THAT LABELED
6842 ;* ADRPTR WILL BE HANDLED BY THE NORMAL 004 TRAP SERVICE ROUTINE.
6843 ;*
6844 ;* SUBORDINATE ROUTINES CALLED: NONE.
6845 ;*****
6846
6847 027404 021627 017270 TP4RTN;; CMP (SP),ADRPTR ;COMPARE EXPECTED ADR AGAINST TRAP RET PC.
6848 027410 001402 BEQ 2$ ;IF THEY MATCH, CONTINUE THIS ROUTINE.
6849 027412 000177 152640 JMP @TP4VEC ;IF NOT, JUMP TO NORMAL 004 TRAP SERVICE RTN.
6850 027416 052767 100000 152630 2$; BIS @BIT15,TP4FLG ;SET THE 004 TRAP OCCURED FLAG.
6851 027424 000002 RTI ;ALL DONE, GO BACK TO THE TEST.
    
```



```

6909 027476 050567 153000          BIS    R5,TXDONF      ;SET THIS LINE'S TX DONE FLAG.
6910 027502 000757                BR     2$             ;LOOP TO CHECK TX.ACTION AGAIN.
6911                                ;+
6912                                ; START THE DMA OF THE NEXT BUFFER (DATA PATTERN) ON THIS LINE.
6913                                ; GET THE DATA PATTERN LENGTH FOR THIS LINE.
6914                                ; GET THE START ADDRESS OF THE DATA PATTERN.
6915                                ;-
6916 027504 016403 003202        6$:   MOV    DPLENB(R4),R3    ;PASS DATA PATTERN LENGTH FOR LINE TO DODMA.
6917 027510 016402 003342        MOV    TXPTRB(R4),R2    ;PASS THE TX START ADR TO DODMA.
6918                                ;+
6919                                ; WRITE DMA PARAMETERS TO THE DUT.
6920                                ;-
6921 027514 004767 170172        JSR    PC,DODMA
6922 027520 103403                BCS    8$             ;SKIP ERROR IF DODMA WAS SUCCESSFUL.
6923                                ;+
6924                                ; SET THE PROPER BIT OF THE TX INTERRUPT FLAGS TO INDICATE THE LINE ERROR.
6925                                ;-
6926 027522 050567 152536        BIS    R5,TXINTF      ;INDICATE THE ERROR.
6927 027526 000402                BR     10$            ;SKIP UPDATING POINTERS AND COUNTERS.
6928                                ;+
6929                                ; UPDATE THE TX CHARACTER FOR THIS LINE.
6930                                ; UPDATE THE TX BUFFER POINTER FOR THIS LINE.
6931                                ;-
6932 027530 060364 003502        8$:   ADD    R3,IXCNTB(R4) ;ADD THE DATA PAT LENGTH TO THE TX COUNT.
6933                                ;+
6934                                ; LOOP TO CHECK THE TX.ACTION BIT FOR ANOTHER LINE.
6935                                ;-
6936 027534 000742                10$:  BR     2$             ;LOOP BACK TO CHECK TX.ACTION BIT AGAIN.
6937                                ;-
6938 027536 016701 152472        60$:  MOV    IESSTAT,R1    ;GET THE PRESENT STATES OF TX,IE & RX,IE BITS.
6939 027542 042700 177760        BIC    0177760,R0     ;GET SAVED IND.ADR.REG FIELD BITS.
6940 027546 050001                BIS    R0,R1          ;COMBINE IND.ADR.REG FIELD BITS WITH IE BITS.
6941 027550 010177 152424        MOV    R1,@CSRA      ;RESTORE THE DUT CSR IND.ADR.REG FIELD.
6942 027554                PASS                ;RESTORE GPRS.
6943 027556 004736                JSR    PC,@(SP)+      ;RETURN TO PREG05 SUBRT.
                                RTI
    
```

6952
6953
6954
6955
6956
6957
6958
6959
6960

.SBTTL REPORT CODING SECTION

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

6961 027560
027560
6962
6963 027560
027560 000167
027562 000000
6964
6965
6966
6967 027564
027564
027564 104425

BGNRPT

EXIT RPT

.EVEN

ENDRP1

L\$RPT::

.WORD J\$JMP
.WORD L10016-2-

L10016:

TRAP C\$RPT

6969
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991

.SBTTL PROTECTION TABLE

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

BGNPROT

L\$PROT::

027566
027566
027566 177777
027570 177777
027572 177777

-1
-1
-1

;OFFSET INTO P-TABLE FOR CSR ADDRESS
;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
;OFFSET INTO P-TABLE FOR DRIVE NUMBER

ENDPROT

7013
7014
7015
7016
7017
7018
7019
7020
7021
7022
7023
7024
7025
7026
7027 027574
027574
7028
7029 027574 012700 000040
027574 104447
027600
7030 027602 103416
027602
7031
7032 027604 012700 000037
027604 104447
027610
7033 027612 103556
027612
7034
7035 027614 012700 000035
027614 104447
027620
7036 027622 103555
027622
7037
7038 027624 012700 000036
027624 104447
027630
7039 027632 103161 000540
027632 000167
7040 027634 000167 000540
7041 027640
7042 027640 104433
027640
7043
7044
7045
7046 027642 012700 000114
027642 104462
027646 010001
027650
7047 027652 012167 152412
7048 027656 012167 152410
7049 027662 012167 152406
7050 027666 012167 152404
7051 027672 026727 152400 000062
7052 027700 001004

```

.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.
*****
***
      BGNINIT
      L$INIT::
;SEE IF PROGRAM JUST STARTED, BR IF YES
      READEF DEF.START
      MOV DEF.START,RO
      TRAP C$REFG
      BCOMPLETE NEWSTA
      BCS NEWSTA
;SEE IF PROGRAM JUST RESTARTED, BR IF YES
      READEF DEF.RESTART
      MOV DEF.RESTART,RO
      TRAP C$REFG
      BCOMPLETE NEWRES
      BCS NEWRES
;SEE IF THIS IS A NEW PASS, BR IF YES
      READEF DEF.NEW
      MOV DEF.NEW,RO
      TRAP C$REFG
      BCOMPLETE NEWPAS
      BCS NEWPAS
;SEE IF PROGRAM WAS JUST CONTINUED
      READEF DEF.CONTINUE
      MOV DEF.CONTINUE,RO
      TRAP C$REFG
      BNCOMPLETE GETPRM
      BCC GETPRM
      JMP ENDIT
NEWSTA:
      BRESET ;RESET THE BUS TO PREVENT ILLEGAL INTERRUPTS.
      TRAP C$RESET
;
; SET UP FOR LINE TIME CLOCK INTERRUPTS.
;
      CLOCK L,R1 ;GET THE CLOCK PARAMETERS.
      MOV L,RO
      TRAP C$CLOCK
      MOV RO,R1
      MOV (R1),CLKCSR ;STORE CLOCK CSR ADDRESS.
      MOV (R1),CLKBRI ;STORE CLOCK BUS REQ INT LEVEL.
      MOV (R1),CLKVEC ;STORE CLOCK INTERRUPT VECTOR.
      MOV (R1),CLKHRZ ;STORE CLOCK FREQUENCY.
      CMP CLKHRZ,#50. ;TEST FOR 50HZ LINE FREQUENCY.
      BNE 24 ;BRANCH IF CLOCK IS NOT 50HZ.

```

```

7053 027702 012767 000024 152400      MOV    020.,MSTICK      ;INDICATE 20MS PER CLOCK TICK.
7054 027710 000403                    BR      4#
7055 027712 012767 000021 152370 2# : MOV    017.,MSTICK      ;INDICATE 17 MS PER CLOCK TICK.
7056 027720 012767 000021 152370 4# : SETVEC CLKVEC,0CLKINT,0PRI06 ;INITIALIZE CLOCK INTERRUPT VECTOR.
                                MOV    0PRI06,-(SP)
                                MOV    0CLKINT,(SP)
                                MOV    CLKVEC,(SP)
                                MOV    03,(SP)
                                TRAP   C$SVEC
                                ADD    010,SP
7057 027746 016700 152324      MOV    CLKHRZ,RO      ;INITIALIZE THE BREAK COUNT
7058 027752 006300      ASI    RO              ; TO CAUSE A BREAK
7059 027754 010067 152326      MOV    RO,BCOUNT     ; EVERY 2 SECONDS.
7060 027760 012700 000240      SETPRI 0PRI05        ;ALLOW CLOCK INTERRUPTS DISABLE OTHERS.
                                MOV    0PRI05,RO
                                TRAP   C$SPRI
7061
7062 ; ENABLE THE LINE TIME CLOCK (LTC) CHECKING TO MAKE SURE THAT THE CSR
7063 ; IS ACCESSABLE.
7064 ; FIRST SET UP TO CATCH ANY 004 TRAPS WHICH OCCUR:
7065
7066 027766 016767 150012 152262      MOV    4,TP4VEC      ;SAVE THE EXISTING 004 TRAP VECTOR.
7067 027774 012767 027404 150002      MOV    0TP4RTN,4     ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
7068
7069 ; ENABLE LTC CHECKING FOR 004 TRAP IN CASE CSR IS NOT THERE.
7070
7071 030002 005067 152246      CLR    TP4FLG        ;CLEAR THE 004 TRAP FLAG.
7072 030006 012767 000100 152252      MOV    0BIT6,WORD1   ;SET UP TO SET BIT6 OF THE LTC CSR.
7073 030014 012700 002266      MOV    0WORD1,RO     ;SET UP WORD1 AS THE CKTRAP MOVE SOURCE.
7074 030020 016701 152244      MOV    CLKCSR,R1     ;SET UP LTC CSR AS DESTINATION FOR CKTRAP MOVE.
7075 030024 004767 167226      JSR    PC,CKTRAP     ;MOVE AND CHECK FOR TRAP.
7076 030030 016767 152222 147746      MOV    TP4VEC,4      ;RESTORE THE NORMAL 004 TRAP VECTOR.
7077 030036 103403      BCS    6#            ;IF NO TRAP, LTC IS THERE SO CONTINUE.
7078 030040 005067 152232      CLR    CLKHRZ       ;CLEAR LTC FREQUENCY WORD TO INDICATE NO LTC.
7079 030044 000402      BR     8#            ;BYPASS THE FOLLOWING CALIBRATION PROCEDURES.
7080
7081 ; CALIBRATE THE DELAY ROUTINE MILLI-SECOND DELAY COUNT VALUE.
7082
7083 030046 004767 166020 6# : JSR    PC,CALMSL
7084
7085 ; CHECK FOR MEMORY MANAGEMENT PRESENT ON THIS MACHINE.
7086 ; IF MEM MGT IS PRESENT, DISABLE IT.
7087
7088 030052 016767 147726 152176 8# : MOV    4,TP4VEC      ;SAVE THE EXISTING 004 TRAP VECTOR.
7089 030060 012767 027404 147716      MOV    0TP4RTN,4     ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
7090 030066 005067 152162      CLR    TP4FLG        ;CLEAR THE 004 TRAP FLAG.
7091 030072 005067 152170      CLR    WORD1         ;PREPARE TO CLEAR THE MEM MGT SRO REGISTER.
7092 030076 012700 002266      MOV    0WORD1,RO     ;SELECT CLEARED WORD AS CKTRAP RTN SOURCE.
7093 030102 016701 152206      MOV    MMSRO,R1     ;SELECT MEM MGT SRO REGISTER AS DESTINATION.
7094 030106 005067 152206      CLR    MMPRES       ;INDICATE NO MEM MGT PRESENT IN CASE IT ISN'T.
7095 030112 005067 152204      CLR    MMENAR       ;INDICATE MEM MGT IS NOT ENABLED.
7096 030116 004767 167134      JSR    PC,CKTRAP     ;CLEAR THE MEM MGT SRO REG AND CHECK FOR TRAP.
7097 030122 016767 152130 147654      MOV    TP4VEC,4      ;RESTORE THE NORMAL 004 TRAP VECTOR.
7098 030130 103003      BCC    10#          ;SKIP INDICATING MEM MGT PRESENT IF IT ISN'T.
7099 030132 012767 000001 152160      MOV    01,MMPRES     ;INDICATE THAT MEM MGT IS PRESENT.
7100 030140 005067 152076 10# : CLR    PASCNT       ;CLR COUNTER USED IN REPORTING ROM VERSION #.
7101 030144 000167 000006      JMP    NEWPAS        ;SKIP AROUND THE BUS RESET, IT'S BEEN DONE.

```



```

7103 030150          NEWRES; BRESET          ;RESET THE BUS TO PREVENT ILLEGAL INTERRUPTS.
          030150 104433          TRAP C$RESET
7104 030152 005067 152064          CLR PASCNT          ;CLR COUNTER USED IN REPORTING ROM VERSION #.
7105
7106 030156          NEWPAS;
7107 030156 012767 177777 152012  MOV # -1,UNITN          ;RESET LOGICAL DEVICE TO -1
7108
7109          ;
7110          ; INCREMENT THE PASS COUNTER, CORRECT FOR ANY OVERFLOW.
7111          ; THIS COUNTER IS USED IN THE ROM VERSION TEST.
7112 030164 005267 152052          INC PASCNT          ;INCREMENT THE PASS COUNTER.
7113 030170 001002          BNE GETPRM          ;BRANCH IF WE HAVE NOT YET! OVERFLOWED.
7114 030172 005367 152044          DEC PASCNT          ;SET PASS COUNT TO 177777 OCTAL.
7115
7116          ; GET THE HARDWARE PARAMETERS FOR THIS UNIT.
7117 030176          GETPRM;
7118 030176 005267 151774          INC UNITN          ;INCREMENT LOGICAL DEVICE NUMBER
7119 030202 026767 151770 151602  CMP UNITN,L$UNIT          ;SEE IF MAXIMUM UNIT NO. EXCEEDED
7120 030210 002362          BGE NEWPAS          ;BR IF YES
7121
7122 030212          GPHRD UNITN,R1          ;GET P-TABLE POINTER INTO R1
          030212 016700 151760          MOV UNITN,R0
          030216 104442          TRAP C$GPHRD
          030220 010001          MOV R0,R1
7123 030222          BCOMPLETE 30$          ;BR IF DEVICE AVAILABLE
7124 030224 000764          BR GETPRM          ;SKIP THIS DEVICE
7125
7126
7127          ;***** HARDWARE PARAMETER MOVING CODE *****
7128 030226 012167 151746 30$: MOV (R1)+,CSRA          ;STORE DHU-11 CSR ADDRESS IN DEV.REG.ADDRESS TABLE
7129 030232 012102          MOV (R1)+,R2          ;GET THE RX INTERRUPT VECTOR ADDRESS.
7130 030234 010267 151726          MOV R2,RXVECA          ;STORE RX INT VECTOR ADDRESS.
7131 030240 062702 000004          ADD #4,R2          ;CALCULATE TX INTERRUPT VECTOR ADDRESS.
7132 030244 010267 151720          MOV R2,TXVECA          ;STORE TX INT VECTOR ADDRESS.
7133 030250 012167 151716          MOV (R1)+,ACTLNS          ;STORE DHU-11 ACTIVE LINE BIT MAP
7134 030254 111167 151714          MOV# (R1),LOPCK          ;STORE DHU-11 LOOPBACK MODE
7135
7136          ;
7137          ; CALCULATE DEVICE REGISTER ADDRESSES, AND PUT THEM IN THE
7138          ; DEVICE REGISTER ADDRESS TABLE.
7139 030260 016701 151714          MOV CSRA,R1          ;COPY CSR ADDRESS
7140 030264 005201          INC R1          ;INCREMENT CSR ADDRESS
7141 030266 005201          INC R1          ; COPY BY 2.
7142 030270 012703 000007          MOV #7,R3          ;SET UP REGISTER COUNT
7143 030274 012702 002202          MOV @RBL#A,R2          ;GET LOCATION WHERE RBL# ADDRESS GOES IN TABLE
7144 030300 010122          1$: MOV R1,(R2)+          ;STORE REGISTER ADDRESS IN TABLE
7145 030302 005201          INC R1          ;INCREMENT REGISTER ADDRESS
7146 030304 005201          INC R1          ; BY 2, FOR THE NEXT DEVICE REGISTER.
7147 030306 005303          DEC R3          ;DECREMENT REGISTER COUNT
7148 030310 001373          BNE 12$          ;LOOP IF NOT DONE
7149
7150          ;
7151          ; INITIALISE THE BMP CODE QUEUE.
7152
7153 030312 012700 002512          MOV @BMPCQB,R0          ;GET THE START ADDRESS OF THE QUEUE.

```

```

7154 030316 012701 002712          MOV    #BMPCQE,R1      ;GET THE END ADDRESS OF THE QUEUE.
7155 030327 010067 152162          MOV    R0,BMPCQP      ;SET THE POINTER TO THE START OF THE QUEUE.
7156 030328 005020          14$:  CLR    (R0)        ;CLEAR OUT THE CONTENTS OF THE QUEUE.
7157 030330 020001          CMP    R0,R1          ;CHECK IF END OF QUEUE HAS BEEN REACHED.
7158 030332 103775          BLO   14$            ;LOOP IF NOT ALL DONE.
7159
7160          ;
7161          ; REPORT THE UNIT NUMBER IF THE SOFTWARE P-TABLE QUESTION WAS ANSWERED YES,
7162          ; AND THE MAXIMUM UNIT NUMBER IS GREATER THAN 1.
7163 030334 032767 000020 151620          BIT    #BIT4,OPTION   ;CHECK IF THE QUESTION WAS ANSWERED YES.
7164 030342 001416          BEQ   16$            ;SKIP REPORTING UNIT NUMBER IF IT IS DISABLED.
7165 030344 026727 151442 000001          CMP    L$UNIT,#1     ;CHECK MAXIMUM NUMBER OF UNITS SELECTED.
7166 030352 003412          BLE   16$            ;DO NOT REPORT UNIT NUMBER IF MAX NUMBER < 1.
7167 030354          PRINTF #MFUNIT,UNITN ;REPORT UNIT NUMBER.
          MOV    UNITN,-(SP)
          MOV    #MFUNIT,-(SP)
          MOV    #2,-(SP)
          MOV    SP,R0
          TRAP  C$PNTF
          ADD   #6,SP
7168 030400          16$:
7169
7170 030400          ENDIT;
7171          ;
7172          ; SET THE PROCESSOR PRIORITY TO DISABLE ALL INTERRUPTS.
7173          ;
7174 030400          SETPRI #PRI07      ;SET PROCESSOR PRIORITY TO 7.
          MOV    #PRI07,R0
          TRAP  C$SPRI
7175
7176 030406          ENDINIT
          L10020:
          TRAP  C$INIT
7177
7178          TNUM == 0          ;INITIALIZE THE ASSEMBLER TEST NUMBER VARIABLE.

```


7217
 7218
 7219
 7220
 7221
 7222
 7223
 7224
 7225
 7226
 7227
 7236
 7237
 7238
 7239
 7240
 7241
 7242
 7254
 7255
 7256
 7257

030412
 030412
 030412 005767 151602
 030416 001401
 030420 104433
 030422
 030422 104432
 030424 000002
 030426
 030426
 030426 104412

.SBTTL CLEANUP CODING SECTION

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

BGNCLN

L\$CLEAN: :

TST CTRLCF
 BEQ 2\$
 BRESET

; DID WE GET HERE BY CTRL-C FROM TEST?
 ; CTRL-C FROM TEST? NO, SKIP BUS RESET.
 ; YES, CLR ANY DMAS OR OUTSTANDING INTERRUPTS.

TRAP C\$RESET

2\$:

EXIT CLN

TRAP C\$EXIT
 .WORD L10022-

.EVEN

ENDCLN

L10022:

TRAP C\$CLEAN

7266
7267
7268
7269
7270
7271
7272
7273
7274

7275 030430
030430

7276
7277
7278
7279
7280
7281
7282

7283 030430 010046
030430 012746 030454
030436 012746 000002
030442 010600
030444 104417
030446 062706 000006

7284 030452 000427

7285

7286 030454 045 101 040
030457 125 116 111
030462 124 045 104
030465 086 045 101
030470 040 104 122
030473 117 120 120
030476 105 104 040
030501 106 122 117
030504 115 040 106
030507 125 122 124
030512 110 105 122
030515 040 124 105
030520 123 124 111
030523 116 107 056
030526 045 116 000

7287

7288 030532

7289

7290 030532
030532 000167
030534 000000

7291

7292

7293 030536
030536
030536 104453

.SBTTL DROP UNIT SECTION

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

RGNDU

L\$DU: :

; INSERT DROP CODE HERE. THIS CODE WILL BE EXECUTED AFTER
; A "DROP" COMMAND OR A "DODU" MACRO EXECUTION. THE PURPOSE
; OF THIS CODE IS TO DO ANY NECESSARY HOUSEKEEPING AFTER A
; UNIT HAS BEEN DROPPED. THIS SECTION IS OPTIONAL.
;*****
PRINTF @DROP,R0 ;REPORT UNIT THAT HAS BEEN DROPPED.

MOV R0, -(SP)
MOV @DROP, -(SP)
MOV @2, (SP)
MOV SP, R0
TRAP C\$PNTF
ADD @6, SP

BR EDROP ;BRANCH AROUND THE MESSAGE.

DROP: .ASCIZ/*A UNIT#D6*A DROPPED FROM FURTHER TESTING.*N/

EDROP: .EVEN

EXIT DU

.WORD L\$JMP
.WORD L10023-2-

ENDDU

L10023: TRAP C\$DC

7295
 7296
 7297
 7298
 7299
 7300
 7301
 7302
 7303
 7304
 7305
 7306
 7307
 7308
 7309
 7310
 7311
 7312 030540
 030540
 7313
 7314
 7315
 7316
 7317
 7318
 7319
 7320
 7321 030540
 030540 0001r7
 030542 000000
 7322
 7323
 7324
 7325
 7326 030544
 030544
 030544 104452
 7327

```

*****
:
:           FVTD.ADD
:
*****

```

```

.SBTTL  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

```

```

      L:AU:;

```

```

*****
:
:   INSERT ADD CODE HERE.  THIS CODE WILL BE EXECUTED AFTER
:   AN "ADD" COMMAND.  THE PURPOSE OF THIS CODE IS TO DO ANY
:   HOUSEKEEPING THAT MAY BE NECESSARY AFTER A UNIT HAS BEEN ADDED.
:   THIS SECTION IS OPTIONAL.
*****

```

```

      EXIT  AU

```

```

      .WORD  J$JMP
      .WORD  L10024-2-

```

```

      .EVEN

```

```

      ENDAU

```

```

      L10024: TRAP  C$AU

```

```

7329
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342
7343 030546
      030546
7344      000001
7345 030546 012767 000001 151504
7346 030554 012767 177777 151436
7347 030562 012767 000145 154526
7348 030570 012767 010247 154522
7349 030576 012767 013500 154516
7350
7351
7352
7353 030604 016767 147174 151444
7354 030612 012767 027404 147164
7355 030620 005005
7356
7357
7358
7359
7360
7361 030622 016700 151352
7362 030626 012701 031020
7363 030632 004767 166420
7364 030636 103402
7365 030640 052705 100001
7366 030644 042767 000017 000146 4$:
7367 030652 010100
7368 030654 016701 151320
7369 030660 004767 166372
7370 030664 103403
7371 030666 052705 100002
7372 030672 000434
7373
7374
7375
7376 030674 012702 000010
7377 030700 016767 151274 000110
7378 030706 016700 000104
7379 030712 012701 031020
7380 030716 004767 166334
7381 030722 103402
7382 030724 052705 100001
7383 030730 010100
7384 030732 016701 000060

```

```

.SBTTL HARDWARE TEST - ADRA -
;+
;*****
; REGISTER ADDRESS TEST -
;+
; THIS TEST VERIFIES THAT THE DEVICE REGISTERS WILL RESPOND TO THE PROPER
; UNIBUS HANDSHAKING SIGNALS WHEN ACCESSED. IF THE DHU11 DOES NOT RESPOND
; TO THE ACCESS ATTEMPTS (IF THE DHU11 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.
;+
;*****
;--
      BGNTST
;+
; T1::
; INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      TNUM ** TNUM + 1
; SET UP THE TEST NUMBER. (1)
      MOV @TNUM,TSTNUM
; INDICATE THAT WE ARE IN A TEST.
      MOV @-1,CTRLCF
; SET THE TEST ERROR NUMBER IN THE TABLE.
      MOV @101,,ERRNBR
; SET UP THE TEST FAILURE MESSAGE IN THE TABLE.
      MOV @EMO103,ERRMSG
; SET-UP THE ERROR ROUTINE IN THE ERROR TABLE.
      MOV @ERO101,ERRBLK
;+
; SET UP TO CATCH ANY 004 TRAPS WHICH OCCUR:
;--
      MOV 4,TP4VEC ;SAVE THE EXISTING 004 TRAP VECTOR.
      MOV @TPARTN,4 ;SET 004 TRAP VECTOR TO OUR SERVICE RTN ADR.
      CLR R5 ;CLEAR THE ERROR FLAGS.
;+
; HERE BEGINS THE LOOP TO TEST THE REGISTERS FOR A LINE.
; FIRST TEST THE CSR AND SET THE IND.ADR.REG (I.A.R) FIELD.
;--
      MOV CSRA,R0 ;SET UP CSR AS THE CKTRAP MOVE SOURCE.
      MOV @52$,R1 ;SET UP DESTINATION LOCATION FOR CKTRAP MOVE.
      JSR PC,CKTRAP ;MOVE AND CHECK FOR TRAP.
      BCS 4$ ;IF NO TRAP, BYPASS ERROR.
      BIS @100001,R5 ;SET FATAL READ ERROR FLAGS.
      BIC @17,52$ ;CLEAR THE I.A.R FIELD OF THE CSR DATA.
      MOV R1,R0 ;USE OLD DESTINATION FOR SOURCE OF CKTRAP MOVE.
      MOV CSRA,R1 ;SET UP CSR AS THE CKTRAP MOVE DESTINATION.
      JSR PC,CKTRAP ;MOVE AND CHECK FOR TRAP.
      BCS 6$ ;IF NO TRAP, BYPASS ERROR.
      BIS @100002,R5 ;SET FATAL WRITE ERROR FLAGS.
      BR 40$ ;EXIT AND REPORT FATAL ERROR.
;+
; NOW, WE TEST EACH REGISTER FOR THIS LINE.
;+
; 6$:
; INIT REGISTER COUNTER TO 8.
      MOV @8,,R2
; INITIALIZE THE REGISTER POINTER.
      MOV CSRA,50$
; SET UP REGISTER AS THE SOURCE FOR CKTRAP MOVE.
; 8$:
      MOV 50$,R0
; SET UP LOCAL STORAGE AS THE DEST FOR CKTRAP.
      MOV @52$,R1
; PERFORM THE MOVE, CHECK FOR TRAP.
      JSR PC,CKTRAP
; IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS.
      BCS 10$
; SET FATAL READ ERROR FLAGS.
      BIS @100001,R5
; USE OLD DEST AS SRC FOR CKTRAP MOVE.
; 10$:
      MOV R1,R0
; SET UP REGISTER AS THE DEST FOR CKTRAP MOVE.
      MOV 50$,R1

```

```

7385 030736 004767 166314      JSR      PC,CKTRAP      ;PERFORM THE MOVE, CHECK FOR TRAP,
7386 030742 103402      BCS      12$           ;IF NO TRAP, BYPASS THE SETTING OF ERROR FLAGS,
7387 030744 052705 100002      BIS      @100002,R5    ;SET FATAL WRITE ERROR FLAGS,
7388 030750 005267 000042      12$:    INC      50$   ;INCREMENT THE REGISTER
7389 030754 005267 000036      INC      50$         ; POINTER BY 2.
7390 030760 005302      DEC      R2          ;COUNT THE REGISTER.
7391 030762 001351      BNE      8$          ;LOOP TO TEST THE NEXT REGISTER ADDRESS.
7392
7393
7394      ;+
7394      ; DONE CHECKING DEVICE REGISTER ADDRESSES.
7395      ; REPORT ANY ERRORS AND EXIT.
7396      ;-
7397
7398 030764 016767 151266 147012 40$:  MOV      TP4VEC,4     ;RESTORE THE NORMAL 004 TRAP VECTOR.
7399 030772 005705      TST      R5          ;CHECK THE ERROR FLAGS.
7400 030774 100012      BPL      60$         ;EXIT ROUTINE IF NO ERRORS.
7401
7402      ;+
7402      ; REPORT "DEVICE REGISTER ACCESS TEST FAILED"
7403      ;-
7404 030776      ERROR
7404 030776 104460      TRAP      C$ERROR
7405
7406
7407 031000      DODU      UNITN     ;DROP THIS UNIT FROM FUTHER TESTING.
7407 031000 016700 151172      MOV      UNITN,R0    ;
7407 031004 104451      TRAP      C$DODU
7408 031006 005067 151206      CLR      CTRLCF     ;INDICATE NO CTRL-C ABORT FROM TEST.
7409 031012      DOCLN     ;ABORT THIS SUB PASS.
7409 031012 104444      TRAP      C$DCLN
7410 031014 000402      BR       60$
7411
7412      ;***** LOCAL STORAGE, *****
7413 031016 000000      50$:    .WORD 0      ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
7414 031020 000000      52$:    .WORD 0      ;STORAGE FOR THE SOURCE OR DEST OF THE CKTRAP MOVE.
7415      ;***** END *****
7416
7417 031022 005067 151172      60$:    CLR      CTRLCF     ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7418 031026      ENDTST
7418 031026      L10025:
7418 031026 104401      TRAP      C$ETST

```



```

7420 .SBTTL  HARDWARE TEST          - KBECHO -
7421 ;* *****
7422 ;* - KEYBOARD ECHO TEST -
7423 ;* THIS IS A TEST WHICH PUTS UARTS FOR THE ACTIVE LINES INTO REMOTE
7424 ;* LOOPBACK MODE.  THE ACTIVE LINE UARTS ARE SET UP WITH A BAUDRATE
7425 ;* WHICH IS SPECIFIED BY THE OPERATOR.  THIS TEST SETS UP THE LINES
7426 ;* FOR: 1 STOP BIT, NO PARITY AND 8 BITS/CHARACTER.
7427 ;* THE TEST EXECUTES INDEFINITELY UNTIL TERMINATED BY THE OPERATOR.
7428 ;*
7429 ;* THIS TEST CAN BE USED FOR LOOPING BACK TERMINAL KEYBOARD INPUT ONTO
7430 ;* A TERMINAL CRT OR IT CAN BE USED AS A GENERAL LOOPBACK METHOD FOR
7431 ;* TESTING COMMUNICATIONS LINKS TO THE OUT FROM THE OTHER END OF THE
7432 ;* CHANNEL.  DTR AND RTS ARE SET ON THE SELECTED LINES DURING THIS
7433 ;* TEST TO ALLOW THE TESTING OF MODEM LINKS.
7434 ;*
7435 ;* *****
7436 031030 BGNTST
7437 031030          SETPRI  #PRI05          ;ALLOW LTC INTERRUPTS.          T2::
7438 031030 012700 000240          MOV      #PRI05,RO
7439 031034 104441          TRAP     C$SPRI
7440 031034 000002          TNUM  ** TNUM + 1      ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
7441 031036 012767 000002 151214  MOV     #TNUM,TSTNUM      ;SET UP THE TEST NUMBER.          (94)
7442 ;*
7443 ;* VERIFY THAT THE TEST SHOULD BE PERFORMED.  MUST HAVE THE FOLLOWING:
7444 ;* KEYBOARD ECHO LOOPBACK SELECTED.
7445 ;* MANUAL INTERVENTION ALLOWED.
7446 ;*
7447 031044 126727 151124 000005  CMPB   LOPBCK,#5      ;TEST THE LOOPBACK TYPE INDICATOR.
7448 031052 001402          BEQ     2$              ;KBD ECHO LPBCK SELECTED? YES, CONTINUE TEST.
7449 031054 104432          EXIT    TST              ;NO, ABORT THE TEST.
7450 031056 000212          TRAP   C$EXIT
7451 031060 104450          .WORD  L10026-
7452 031062 103402          2$:   MANUAL          ;CHECK FOR MANUAL INTERVENTION ALLOWED.
7453 031064 104432          TRAP   C$MANI
7454 031066 000202          BCOMPETE 4$          ;MANUAL INTERVENTION ALLOWED? YES, DO TEST.
7455 031066 104432          BCS    4$
7456 031066 000202          EXIT    TST              ;NO, ABORT THE TEST.
7457 031066 000202          TRAP   C$EXIT
7458 031066 000202          .WORD  L10026-
7459 031070 012767 177777 151122 4$:   MOV     #1,CTRLCF      ;INDICATE THAT WE ARE IN A TEST.
7460 031076 012767 000001 154210  MOV     #1,ERRTYP      ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
7461 031104 012767 022271 154204  MOV     #9401,,ERRNBR   ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
7462 031112 012767 013022 154200  MOV     #EM9401,ERRMSG  ;SET ERROR MESSAGE ADDRESS IN ERRIBL.
7463 031120 005067 151354  CLR     ERSMRF         ;INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
7464 ;*
7465 ;* RESET THE DU. TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO.
7466 ;* CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.
7467 ;* THIS SUBROUTINE REPORTS ERROR >>>> 9401 <<<<<.
7468 ;*
7469 031124 004767 166206  JSR     PC,CLNRST      ;RESET THE DHU-11, REPORT ANY ERRORS FOUND.
7470 031130 103402          BCS    ,+6            ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
7471 031132 000167 000120  JMP     60$           ;RESET FAILURE, ABORT THIS TEST.
7472 ;*
7473 ;* PRINT THE TEST NAME.
7474 ;*

```

```

7468 031136          PRINTF #EF0503,#EM9401
      031136 012746 013022
      031142 012746 005530
      031146 012746 000002
      031152 010600
      031154 104417
      031156 062706 000006
                                MOV #EM9401,-(SP)
                                MOV #EF0503,-(SP)
                                MOV #2,-(SP)
                                MOV SP,R0
                                TRAP C$PNTF
                                ADD #6,SP

```

7469
7470
7471
7472
7473
7474
7475
7476
7477

```

;+
; SET UP THE DUT UARTS WITH THE PROPER LINE PARAMETERS.
; GET THE DESIRED BAUDRATE FROM THE OPERATOR.
; CALCULATE PROPER DUT LPR CONTENTS.
; SET UP THE DUT LPR REGISTERS.
; GET THE PROPER DUT LNCTRL REGISTER CONTENTS.
; SET UP THE DUT LNCTRL REGISTERS.
;-

```

```

7478 031162 004767 167140      JSR PC,GETBDR
7479 031166 010100              MOV R1,R0
7480 031170 006301              ASL R1
7481 031172 006301              ASL R1
7482 031174 006301              ASL R1 ;GET DUPLICATE COPIES OF BAUDRATE CODE
7483 031176 006301              ASL R1 ; IN THE UPPER BYTE OF THE NEW
7484 031200 050100              BIS R1,R0 ; LPR CONTENTS.
7485 031202 000300              SWAB R0
7486 031204 042700 000377      BIC #377,R0 ;SET UP 1 STOP BIT, NO PARITY, 8 BITS/CHAR
7487 031210 052700 000030      BIS #30,R0 ; IN THE LPR CONTENTS.
7488
7489 031214 016705 150752      MOV ACTLNS,P5 ;GET THE ACTIVE LINES BIT MAP.
7490 031220 004767 175672      JSR PC,WTW'PR ;SET UP THE DUT LPR REGISTERS FOR ACTIVE LINES.
7491
7492 031224 012700 011304      MOV #11304,R0 ;SET UP DTR, R1S, REMOTE LPBK, AND RX ENABLE.
7493 031230 004767 175632      JSR PC,WTWLNC ;SET UP THE DUT LNCTRL REGS FOR ACTIVE LINES.
7494
7495
7496
7497
7498

```

7499
7500
7501
7502
7503
7504

```

;+
; WAIT FOR THE OPERATOR TO TERMINATE THE TEST.
; PROMPT "TYPE <CR> TO TERMINATE THE TEST:"
;-

```

```

7499 031234 010767 000001 150766  MOV #1,GMANWD ;SET UP DEFAULT ANSWER TO YES.
7500 031242          GMANIL TERMMSG,GMANWD,1,YES
      031242 104443
      031244 000404
      031246 002230
      031250 000130
      031252 013437
      031254 000001
      031256
                                TRAP C$GMAN
                                BR 10000$
                                .WORD GMANWD
                                .WORD T$CODE
                                .WORD TERMMSG
                                .WORD 1
                                10000$:

```

7505
7506
7507

```

;+
; WE GOT A RESPONSE FROM THE OPERATOR, SO TERMINATE THE TEST.
;

```

```

7505 031256          60$: SETPRI #PRI07 ;DISABLE ALL INTERRUPTS.
      031256 012700 000340
      031262 104441
      031264 005067 150730      CLR CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7506 031270          ENDTST
      031270
      031270 104401
                                L10026:
                                TRAP C$ETST

```

7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539

7540 031272
031272
7541 031272 012700 000240
031272 104441
031276 000003
7542 000003
7543 031300 012767 000003 150752
7544
7545
7546
7547
7548
7549 031306 126727 150662 000004
7550 031314 001402
7551 031316
031316 104432
031320 000702
7552 031322 2\$: MANUAL
031322 104450
7553 031324
031324 103402
7554 031326
031326 104432
031330 000672
7555
7556 031332 012767 177777 150660 4\$:

```
.SBTTL  HARDWARE TEST          - MODLPB -
*****
- MODEM LOOPBACK TEST -
THIS TEST IS USED TO MOVE DATA THROUGH A MODEM WHICH IS CONNECTED TO
ONE OF THE DEVICE SERIAL PORTS. THIS TEST IS RUN ONLY IF MODEM
LOOPBACK IS SPECIFIED. THIS TEST UTILIZES THE FOLLOWING OPERATOR
DIALOGUE:
MODEM BAUDRATE IN BPS: (D) 1200 ?
TYPE <CR> WHEN MODEM LINK ESTABLISHED: (L) Y ?
MODEM STATUS SIGNAL REPORT:
LINE #N: DSR=N, RI=N, DCD=N, CTS=N
... REPEATED FOR EACH ACTIVE LINE
NUMBER OF 256 BYTE DATA PATTERNS TO SEND ON EACH SELECTED LINE
(1-255, 0=SEND UNTIL +C): (D) 1 ?
PRINT MODEM STATUS SIGNAL REPORT AFTER EACH PATTERN: (L) Y ?

AT THE COMPLETION OF SENDING THE SPECIFIED NUMBER OF DATA PATTERNS THE
TEST ISSUES THE FOLLOWING PROMPT:
EXIT THE TEST (N = LOOP BACK TO SEND MORE DATA): (L) Y ?

IF EXTENDED ERROR REPORTING IS ALLOWED, A REPORT IS PRINTED AT THE END
OF EACH DATA PATTERN WITH THE FOLLOWING FORMAT:
MODEM LOOPBACK TEST STATUS REPORT: PATTERN #NNN (D) COMPLETED.

THIS TEST IS PERFORMED USING 3 BITS PER CHARACTER, 1 STOP BIT, AND NO
PARITY. THIS TEST DOES NOT SUPPORT SPLIT SPEED. ALL SELECTED LINES
ARE TESTED AT THE SELECTED BAUDRATE. AN ERROR SUMMARY IS REPORTED AT
THE END OF THE TEST IF ANY LINES HAVE EXCEEDED THE NUMBER OF INDIVIDUAL
DATA ERRORS TO REPORT AS SELECTED IN THE SOFTWARE P-TABLE DIALOGUE.
*****
BGNIST
T3:
SETPRI #PRI05 ;ALLOW LTC INTERRUPTS.
MOV #PRI05,RO
TRAP C$SPRI
TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (89)

; VERIFY THAT THE TEST SHOULD BE PERFORMED. MUST HAVE THE FOLLOWING:
; MODEM LOOPBACK SELECTED.
; MANUAL INTERVENTION ALLOWED.
CMPB LOPBCK,#4 ;TEST THE LOOPBACK TYPE INDICATOR.
BEQ 2$ ;MODEM LOOPBACK SELECTED? YES, CONTINUE TEST.
EXIT TST ;NO, ABORT THE TEST.
TRAP C$EXIT
WORD L10027-

2$: MANUAL ;CHECK FOR MANUAL INTERVENTION ALLOWED.
TRAP C$MANI
BCOMPLETE 4$ ;MANUAL INTERVENTION ALLOWED? YES, DO TEST.
BCS 4$
EXIT TST ;NO, ABORT THE TEST.
TRAP C$EXIT
WORD L10027-

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

```

7557 031340 012767 000001 153746      MOV    #1,ERRIYP      ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
7558 031346 012767 021305 153742      MOV    #8901,,ERRNBR  ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
7559 031354 012767 011146 153736      MOV    #EM8901,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRTABL.
7560 031362 005067 151112              CLR    ERSMRF        ;INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
7561
7562
7563
7564
7565
7566
7567 031366 004767 165744      JSR    PC,CLNRST     ;RESET THE DMU-11, REPORT ANY ERRORS FOUND.
7568 031372 103402              BCS    .+6           ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
7569 031374 000167 000610      JMP    60$           ;RESET FAILURE, ABORT THIS TEST.
7570
7571
7572
7573
7574 031400              SETPRI #PRI07        ;DISABLE ALL INTERRUPTS.
7575 031400 012700 000340              MOV    #PRI07,RO
7576 031404 104341              TRAP  C$SPRI
7577 031406              SETVEC TXVECA,#TXDMA,#PRI06 ;SELECT DMA TX INT SERVICE WIN.
7578 031406 012746 000300              MOV    #PRI06,-(SP)
7579 031412 012746 027426              MOV    #TXDMA,-(SP)
7580 031416 016746 150546              MOV    TXVECA,-(SP)
7581 031422 012746 000003              MOV    #3,-(SP)
7582 031426 104437              TRAP  C$SVEC
7583 031430 062706 000010              ADD    #10,SP
7584 031434              SETVEC RXVECA,#RXCHRS,#PRI06 ;SELECT RX INT SERVICE RTN.
7585 031434 012746 000300              MOV    #PRI06,-(SP)
7586 031440 012746 027216              MOV    #RXCHRS,-(SP)
7587 031444 016746 150516              MOV    RXVECA,-(SP)
7588 031450 012746 000003              MOV    #3,-(SP)
7589 031454 104437              TRAP  C$SVEC
7590 031456 062706 000010              ADD    #10,SP
7591 031462              SETPRI #PRI04        ;ALLOW INTERRUPTS.
7592 031462 012700 000200              MOV    #PRI04,RO
7593 031466 104441              TRAP  C$SPRI
7594
7595
7596
7597
7598
7599
7600
7601
7602
7603
7604
7605
7606
7607
7608
7609
7610
7611
7612
7613
7614
7615
7616
7617
7618
7619
7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642
7643
7644
7645
7646
7647
7648
7649
7650
7651
7652
7653
7654
7655
7656
7657
7658
7659
7660
7661
7662
7663
7664
7665
7666
7667
7668
7669
7670
7671
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681 031470 012700 003302      MOV    #RCNTB,RO
7682 031474 004767 165660      JSR    PC,CLR16W     ;CLEAR THE RX ERROR COUNTERS TABLE.
7683
7684
7685
7686
7687
7688
7689
7690
7691
7692
7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821
7822
7823
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840
7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899
7900
7901
7902
7903
7904
7905
7906
7907
7908
7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000
    
```

```

7592          | SET UP THE BIT MAP OF UNUSED TX/RX BITS.
7593          |
7594 031524 004767 166576          | JSR    PC,GETBDR
7595 031530 010100          | MOV    R1,R0
7596 031532 006301          | ASL   R1
7597 031534 006301          | ASL   R1
7598 031536 006301          | ASL   R1          |GET DUPLICATE COPIES OF BAUDRATE CODE
7599 031540 006301          | ASL   R1          | IN THE UPPER BYTE OF THE NEW
7600 031542 050001          | BIS   R0,R1          | LPR CONTENTS.
7601 031544 000301          | SWAB  R1
7602 031546 042701 000377          | BIC   0377,R1          |SET UP 1 STOP BIT, NO PARITY, 8 BITS/CHAR
7603 031552 052701 000030          | BIS   030,R1          | IN THE LPR CONTENTS.
7604
7605 031556 004767 166754          | JSR    PC,GETTIM          |GET TIME-OUT BASED ON MINIMUM BAUDRATE IN USE.
7606 031562 012767 177400 150442          | MOV    0177400,IBM          |FORM BIT MAP OF UNUSED TX/RX BITS.
7607
7608          | SET UP A 256 BYTE DATA PATTERN.
7609          |
7610 031570 005003          | CLR   R3          |PREPARE TO START DATA PATTERN AT 255.
7611 031572 012702 003602          | MOV   0BUFBA5,R2          |GET THE BASE OF THE DATA PATTERN BUFFER.
7612 031576 010204          | MOV   R2,R4
7613 031600 105303          | 6$:   DECB  R3          |GET THE NEXT BYTE OF THE DATA PATTERN.
7614 031602 110324          | MOVB  R3,(R4)          |WRITE A BYTE OF THE DATA PATTERN.
7615 031604 105703          | YSTR  R3          |CHECK FOR DONE WRITING DATA PATTERN.
7616 031606 001374          | BNE   6$          |DATA PATTERN DONE? NO, LOOP TO DO NEXT BYTE.
7617          | YES, WRITE 32 BYTE OVERFLOW REGION.
7618 031610 010205          | MOV   R2,R5          |PREPARE SOURCE POINTER.
7619 031612 012700 000020          | MOV   016.,R0          |PREPARE LOOP COUNTER.
7620 031616 012524          | 8$:   MOV   (R5)+,(R4)          |WRITE 2 BYTES OF THE OVERFLOW PATTERN.
7621 031620 005300          | DEC   R0          |COUNT THESE 2 BYTES.
7622 031622 001375          | RNE   8$          |16 WORDS WRITTEN? NO, LOOP TO WRITE ANOTHER.
7623          | YES, COMPLETE DATA PATTERN IS DONE.
7624 031624 012703 000400          | MOV   0256.,R3          |SET DATA PATTERN LENGTH TO 256.
7625
7626          | SET THE DUT RTS AND DTR BITS FOR THE ACTIVE LINES.
7627          |
7628 031630 012700 011000          | MOV   011000,R0          |SPECIFY TO SET RTS AND DTR.
7629 031634 016705 150332          | MOV   ACTLNS,R5          |SPECIFY ACTIVE LINES.
7630 031640 004767 175222          | JSR   PC,WTWLN0          |SET DUT RTS AND DTR ON ALL ACTIVE LINES.
7631
7632          | WAIT FOR THE OPERATOR TO ESTABLISH THE MODEM CONNECTION.
7633          | PROMPT "TYPE <CR> WHEN MODEM LINK ESTABLISHED:"
7634          |
7635 031644 012767 000001 150356          | MOV   01,GMANWD          |SET UP DEFAULT ANSWER TO YES.
7636 031652          | GMANIL EMLMSG,GMANWD,1,YES
7637          | TRAP  C$GMAN
7638          | BR    10000$
7639          | .WORD GMANWD
7640          | .WORD T$CODE
7641          | .WORD EMLMSG
7642          | .WORD 1
7643          | 10000$:
7644          |
7645          | REPORT THE STATE OF THE MODEM STATUS SIGNALS.
7646          | SET DEFAULT OF PRINTING MODEM STATUS AFTER EVERY DATA PATTERN.
7647          |
7648          | JSR   PC,MSSRPT

```

```

7642 031672 012767 000001 150344      MOV    #1,PMSFLG
7643
7644      ; ASK OPERATOR FOR THE NUMBER OF DATA PATTERNS TO SEND.
7645      ; PROMPT: "NUMBER OF 256 BYTE DATA PATTERNS TO SEND ON EACH SELECTED LINE
7646      ;           (1-255, 0=SEND UNTIL ^C); (D) 1?"
7647
7648 031700 012767 000001 150322 10$:  MOV    #1,GMANWD      ;SET DEFAULT NUMBER OF PATTERNS TO 1.
7649 031706      GMANID  NDPMSG,GMANWD,D,377,0,255,YES
      TRAP  C$GMAN
      BR   10001$
      .WORD GMANWD
      .WORD T$CODE
      .WORD NDPMSG
      .WORD 377
      .WORD T$OLIM
      .WORD T$HILIM
      10001$:
7650 031726 016704 150276      MOV    GMANWD,R4
7651 031732 005005      CLR    R5           ;CLEAR THE DATA PATTERN COUNTER.
7652 031734 005067 150264      CLR    FERROR      ;CLEAR THE "AT LEAST ONE ERROR" FLAG
7653
7654      ; ASK IF MODEM STATUS SIGNALS SHOULD BE REPORTED AFTER EACH DATA PATTERN.
7655      ; PROMPT: "PRINT MODEM STATUS SIGNAL REPORT AFTER EACH PATTERN; (L) Y?"
7656      ; USE LAST RESPONSE AS DEFAULT (DEFAULT OF YES THE FIRST TIME).
7657
7658 031740      GMANIL  PMSMSG,PMSFLG,1,YES
      TRAP  C$GMAN
      BR   10002$
      .WORD PMSFLG
      .WORD T$CODE
      .WORD PMSMSG
      .WORD 1
      10002$:
7659
7660
7661      ; SET UP THE OUT AND TX/RX VARIABLES.
7662      ; R1 - TX, RX LPR CONTENTS.
7663      ; R2 - START ADDRESS OF DATA PATTERN TO TX/RX.
7664      ; R3 - LENGTH OF DATA PATTERN.
7665      ; SEND THE DATA.
7666
7667 031754 005205 12$:  INC    R5           ;COUNT THIS DATA PATTERN.
7668 031756 004767 166752      JSR    PC,MODSUP    ;SET UP THE OUT AND TX/RX VARIABLES.
7669
7670 031762 004767 170662      JSR    PC,PURIFO    ;PURGE THE OUT RECEIVE CHARACTER FIFO.
7671 031766 103110      BCC    60$         ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
7672
7673 031770 004767 171140      JSR    PC,PURRXB    ;PURGE THE RX CHAR BUFFER IN MEMORY.
7674 031774 004767 166614      JSR    PC,INIDMA    ;SEND THE FIRST BATCH OF DATA PATTERNS.
7675 032000 012767 021306 153310      MOV    #8902,,ERRNBR ;SET ERROR NUMBER TO 8902.
7676
7677      ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS 8902 THRU 8907.
7678
7679 032006 004767 171156      JSR    PC,RDCHRS    ;READ AND VERIFY THE RX CHARACTERS.
7680
7681 032012 005767 150206      JLT    FERROR      ;HAS AN ERROR BEEN DETECTED?
7682 032016 001404      BEQ    13$         ;BRANCH IF IT HASN'T.

```

```

7683 032020 032767 000100 150134      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
7684 032026 001430                BEQ    16$           ;BRANCH TO THE "EXIT TEST ?" QUESTION IF NOT.
7685
7686 032030 012767 021314 153260 13$:  MOV    #8908,,ERRNBR ;SET ERROR NUMBER TO 8908.
7687
7688                ;* THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 8908 THRU 8911 <<<<.
7689                ;*
7690 032036 004767 174204      JSR    PC,IXRREP    ;REPORT FINAL ERRORS FROM RX/RX.
7691
7692                ;* REPORT END OF DATA PATTERN IF ALLOWED.
7693                ;* "MODEM LOOPBACK TEST STATUS REPORT; PATTERN #NNN (D) COMPLETED."
7694                ;* REPORT THE MODEM STATUS SIGNAL STATES IF REQUESTED.
7695                ;*
7696 032042                PRINTX #EDPFMT,R5
032042 010546                MOV    R5,-(SP)
032044 012746 005427                MOV    #EDPFMT,-(SP)
032050 012746 000002                MOV    #2,-(SP)
032054 010600                MOV    SP,RO
032056 104415                TRAP  C$PNTX
032060 062706 000006                ADD   #6,SP
7697 032064 005767 150154      TST   PMSFLG       ;CHECK THE "PRINT MODEM STATUS" FLAG.
7698 032070 001402                BEQ   14$           ;PRINT MODEM STATUS? NO, SKIP PRINTING.
7699 032072 004767 167126      JSR   PC,MSSRPT    ;REPORT THE MODEM STATUS.
7700
7701                ;* IF THERE ARE MORE DATA PATTERNS TO SEND, LOOP BACK TO SEND AGAIN.
7702                ;*
7703 032076 005304      14$:  DEC   R4           ;COUNT THIS DATA PATTERN.
7704 032100 001403                BEQ   16$           ;LAST DATA PAT SENT? YES, PROMPT FOR EXIT.
7705 032102 100324                BPL   12$           ;NO, CONTINUOUS SENDING? NO, SEND NEXT PAT.
7706 032104 005204                INC   R4           ;YES, RESTORE PATTERN COUNTER.
7707 032106 000722                BR    12$           ;GO TO SEND NEXT DATA PATTERN.
7708
7709                ;* PROMPT FOR EXIT OF THE TEST OR SENDING OF MORE DATA PATTERNS.
7710                ;* PROMPT: "EXIT THE TEST (N = LOOP BACK TO SEND MORE DATA); (L) Y ?"
7711                ;*
7712 032110 012767 000001 150112 16$:  MOV    #1,GMANWD    ;SET DEFAULT ANSWER TO YES.
7713 032116                GMANIL EXTMSG,GMANWD,1,YES
032116 104443                TRAP  C$GMAN
032120 000404                BR    10003$
032122 002230                .WORD GMANWD
032124 000130                .WORD T$CODE
032126 013143                .WORD EXTMSG
032130 000001                .WORD 1
032132                10003$:
7714 032132 026727 150072 000001      CMP   GMANWD,#1    ;CHECK OPERATOR RESPONSE.
7715 032140 001257                BNE   10$           ;EXIT RESPONSE? NO, LOOP TO SEND MORE DATA.
7716
7717                ;* ALL DONE. HAVE BEEN TOLD TO EXIT.
7718                ;* CLEAR DEVICE DTR AND RTS SIGNALS.
7719                ;* DISABLE INTERRUPTS.
7720                ;* CLEAR THE INTERRUPT VECTORS.
7721                ;* REPORT ANY NECESSARY ERROR SUMMARIES.
7722                ;*
7723                ;*
7724 032142 005000                CLR   RO           ;INDICATE TO CLEAR ALL INCR. BITS.
7725 032144 012705 177777                MOV   #MAPLN,R5    ;INDICATE TO CLEAR FOR ALL LINES.
7726 032150 004767 174712      JSR   PC,WTWLC     ;CLEAR ALL THE RTS AND DTR SIGNALS.

```

```

7727
7728 032154          SETPRI  #PRI07          ;DISABLE ALL INTERRUPTS.
      032154 012700 000340
      032160 104441
7729 032162          CLRVEC  TXVECA          ;RETURN TX INT VECTOR TO UNUSED POOL.
      032162 016700 150002
      032166 104436
7730 032170          CLRVEC  RXVECA          ;RETURN RX INT VECTOR TO UNUSED POOL.
      032170 016700 147772
      032174 104436
7731
7732 032176 012767 021320 153112      MOV  #8912,ERRNBR ;SELECT NUMBER 8912 FOR THE NEXT ERROR REPORT.
7733 032204 004767 171620              JSR  PC,REPSMR   ;REPORT ERROR SUMMARIES IF CALLED FOR.
7734 032210          SETPRI  #PRI07          ;DISABLE ALL INTERRUPTS.
      032210 012700 000340
      032214 104441
7735 032216 005067 147776              CLR  CTRLCF     ;INDICATE THAT WE ARE NOT WITHIN A TEST.
7736 032222          ENDTST
      032222 104401
                                L10027:
                                TRAP  C$ETST

```



```

7738 .SBTTL HARDWARE TEST - DMAADR
7739 ;* *****
7740 ;* - DMA ADDRESSING TEST
7741 ;* THIS TEST VERIFIES , AS FAR AS POSSIBLE , THAT THE DUT CAN PERFORM A
7742 ;* DMA FROM A FULL 18 BIT OR 16 BIT ADDRESS. THE TEST RELIES ON FINDING A
7743 ;* COMPLEMENTARY PAIR OF ADDRESSES BETWEEN THE TOP OF PHYSICAL MEMORY AND
7744 ;* THE START OF THE TOP OF THE DIAGNOSTIC PROGRAM .
7745 ;* THIS MAY INVOLVE REMOVING PART OF THE DIAGNOSTIC RUNTIME SERVICES AND
7746 ;* THEN RESTORING. THE NUMBER OF BITS THAT HAVE BEEN SUCCESSFULLY TESTED
7747 ;* WILL BE PRINTED AT THE CONSOLE AT THE END OF THE TEST, IF REQUESTED.
7748 ;*
7749 ;* *****
7750 032224 BGNST
7751 032224 T4::
7752 032224 012700 000240 SETPRI #PRI05 ;ALLOW LTC INTERRUPTS
7753 032230 104441 MOV #PRI05,R0
7754 000004 TRAP C$SPRI
7755 032232 012767 000004 150020 TNUM == TNUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER
7756 032240 012767 177777 147752 MOV #TNUM,TSTNUM ;SET UP THE TEST NUMBER
7757 032246 012767 000001 153040 MOV #-1,CTRLCF ;INDICATE THAT WE ARE IN A TEST
7758 032254 012767 010461 153034 MOV #1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE
7759 032262 012767 010174 153030 MOV #4401,ERRNBR ;SET ERROR NUMBER TO 4401
7760 032270 012767 014032 153024 MOV #EM4401,ERRMSG ;SET ERROR MESSAGE ADDRESS IN TABLE
7761 MOV #ER0503,ERRBLK ;SELECT THE CORRECT ERROR REPORTING ROUTINE
7762 ; CLEAR THE SUCCESS FLAG TO INDICATE TEST FAILURE IN CASE IT DOES
7763 ;
7764 032276 005067 001402 CLR SUCCS ;INDICATE FAILURE , IN CASE THE DUT FAILS
7765 ;
7766 ; RESET THE DUT TO A KNOWN STATE,REMOVE THE STATUS CODES FROM THE FIFO.
7767 ; CLEAR TX AND RX INTERUPT ENABLE BITS IN THE CSR
7768 ;
7769 032302 004767 165030 JSR PC,CLNRST ;RESET THE DHU , REPORT ANY ERRORS
7770 032306 103402 BCS .+6 ;SKIP EXIT OF TEST IF NO FATAL ERROR FOUND.
7771 032310 000167 001332 JMP 60$ ;EXIT THE TEST, FATAL ERROR WAS FOUND.
7772 ;
7773 ; SET UP THE 004 TRAP VECTOR TO POINT TO OUR TRAP SERVICE ROUTINE.
7774 ;
7775 032314 016767 145464 147734 MOV 4,TP4VEC ;SAVE THE EXISTING 004 TRAP VECTOR
7776 032322 012767 027404 145454 MOV #TPARTN,4 ;POINT THE VECTOR AT OUR SERVICE ROUTINE.
7777 ;
7778 ; DETERMINE WHETHER MEMORY MANAGEMENT IS PRESENT
7779 ;
7780 032330 005767 147754 TST MMPRES ;IF MEM MGT IS PRESENT THEN
7781 032334 001007 BNE 1$ ;AVOID SETTING THE DMA TEST ADDR FOR
7782 ; A 16 BIT MACHINE.
7783 032336 012767 001252 147656 MOV #1252,DMTSTA ;SET UP THE FIRST DMA TEST ADDR FOR
7784 ; A 16 BIT MACHINE
7785 032344 012767 000021 001324 MOV #17,BITSTD ;SET THE BITS TESTED TO 16 + 1
7786 032352 000442 BR 10$ ;SINCE MEM MGT ISN'T PRESENT
7787 ; THERE'S NO NEED TO SET UP THE MEM MGT REGS.
7788 ;
7789 ; SET UP THE HIGHEST POSSIBLE TEST ADDRESS IN DMTSTA
7790 ;
7791 032354 012767 005252 147640 1$: MOV #5252,DMTSTA ;SET UP THE FIRST DMA TEST ADDRESS FOR THE

```

```

7792
7793 032362 012767 000023 001306      MOV    #19, BITSTD      ; 18 BIT MACHINE.
7794                                     ; SET THE BITS TESTED TO 18 BITS + 1.
7795                                     ; SET UP THE PARS 0 THROUGH 6 TO RELOCATE TO THE SAME ADDRESS
7796
7797 032370 005000                                     CLR    R0              ; SET THE PAGE BASE ADDRESS TO ZERO
7798 032372 012701 002324      MOV    @PARATB, R1     ; POINT AT THE START OF THE PAR ADDRESS TABLE
7799
7800 032376 010031      2$:    MOV    @R1, @R1+      ; LOAD THE PAR
7801 032400 062700 000200      ADD    #200, R0        ; CALCULATE THE NEXT PAGE ADDRESS
7802 032404 022701 002342      CMP    @PAR7A, R1     ; LOOP UNTIL PARS 0 THROUGH 6
7803 032410 001372      BNE    2$             ; ARE LOADED.
7804
7805                                     ; SET UP THE INPUT/OUTPUT PAGE PAR TO THE TOP 4KW OF PHYSICAL MEMORY
7806
7807 032412 012777 177600 147722      MOV    #177600, @PAR7A ; SET PAR #7 TO POINT AT THE TOP 4KW
7808
7809                                     ; SET UP THE PDRS FOR , NO ABORT/TRAP, UPWARD EXPANSION, 128 BLOCKS PER PAGE
7810
7811 032420 012700 077406      MOV    #77406, R0     ; BIT PATTERN FOR THE PDRS
7812 032424 012701 002344      MOV    @PDRATB, R1    ; POINT AT START OF PDR ADDR TABLE
7813 032430 010031      4$:    MOV    @R1, @R1+      ;
7814 032432 022701 002364      CMP    @PDRATE, R1    ; LOOP UNTIL ALL PDRS HAVE
7815 032436 001374      BNE    4$             ; BEEN SET UP.
7816
7817                                     ; SET THE MEM MGT STATUS REG #3 FOR, 18 BIT ADDRESSING,
7818                                     ; NO UNIBUS MAPPING, NO D SPACE, IN CASE THIS REG EXISTS.
7819                                     ; NOT ALL UNIBUS MACHINES HAVE MEM MGT STATUS REG #3, SO THE CKTRAP ROUTINE
7820                                     ; MUST BE USED TO CATCH AN 004 TRAPS THAT OCCUR IF THE REG DOES NOT EXIST.
7821
7822 032440 005067 001226      CLR    #7             ; SET UP THE SOURCE OPERAND
7823 032444 012700 033672      MOV    #70, R0        ; SET UP THE SOURCE ADDR FOR THE MOVE.
7824 032450 016701 147642      MOV    #MSR3, R1      ; SET UP THE DESTINATION ADDR FOR THE MOVE.
7825 032454 004767 164576      JSR    PC, CKTRAP     ; PERFORM THE MOVE AND CATCH ANY 004 TRAPS.
7826
7827                                     ; TRY AND FIND A COMPLEMENTARY PAIR OF ADDRESSES WITHIN THE MEMORY AND SAVE
7828                                     ; THE CONTENTS OF THE TWO AREAS. THE TEST IS ABANDONED IF A COMPLEMENTARY
7829                                     ; PAIR HAS NOT BEEN FOUND BEFORE THE AREA OF MEMORY CONTAINING THE
7830                                     ; DIAGNOSTIC IS ENCOUNTERED.
7831
7832 032460 012767 027362 145316      10$:   MOV    @TP4BRT, A   ; CHANGE THE 004 TRAP VECTOR TO POINT TO
7833                                     ; TP4BRT SINCE THIS IS THE ROUTINE ASSOCIATED
7834                                     ; WITH THE BYTE SUBROUTINE CKTRPB.
7835
7836 032466                                     MEMORY FFREM          ; GET THE ADDRESS OF THE 1ST FREE WORD
7837 032466 104431                                     TRAP    C$MEM
7838 032470 010067 147532      MOV    R0, FFREM     ; OF MEMORY ABOVE THE DIAGNOSTIC.
7839
7839 032474 012701 003602      MOV    @BU#BAS, R1    ; POINT AT THE BUFFER WHERE THE CONTENTS OF
7840                                     ; THE MEMORY BEING READ ARE TO BE SAVED.
7841 032500 005004                                     CLR    R4              ; CLEAR THE COMPLEMENTARY PAIR INDICATOR (CPL)
7842 032502                                     SETPRI @PRI07         ; DISABLE CLOCK INTERRUPTS
7843 032502 012700 000340                                     MOV    @PRI07, R0
7844 032510 005004                                     TRAP   C$SPRI
7844 032512 005005      12$:   INC    R4          ; INCREMENT THE CPI
7844                                     CLR    R5              ; INDICATE THAT A SAVE OF THE DATA AT
    
```



```

7902 032710 012703 000020      MOV    #16,R3      ;SET THE NUMBER OF DATA BYTES TO BE WRITTEN
7903 032714 012705 000001      MOV    #1,R5      ;INDICATE TO WRITE TO DMTSTA
7904
7905 032720 012701 000340      MOV    #340,R1    ;SET PRIORITY 7 TO DISABLE THE CLOCK
7906 032724 004767 172254      JSR    PC,STPSW   ;
7907
7908 032730 005267 152362      INC    ERRNBR     ;SET THE ERROR NUMBER TO 4403
7909 032734 012701 010272      MOV    #EM4403,R1 ;SELECT THE MESSAGE,
7910                               ; "HOST FAILURE, WRITE FAILED TO AN ADDR WHICH
7911                               ; HAD BEEN SUCCESSFULLY READ, TEST ABANDONED "
7912
7913 032740 004767 164624      JSR    PC,DMRW    ;PERFORM THE TRANSFER
7914 032744 005767 147304      TST   TP4FLG     ;EXIT IF HOST FAILURE
7915 032750 001345                BNE   16$        ;AND REPORT ERROR,
7916 032752 016767 000722 147242  MOV    DUMY,DMTSTA ;SELECT THE LOWER DMA TEST ADDR.
7917 032760 012700 005154      MOV    #SDP28,R0  ;SELECT THE NEXT DATA PATTERN
7918 032764 004767 164600      JSR    PC,DMRW    ;PERFORM THE TRANSFER
7919 032770 005767 147260      TST   TP4FLG     ;EXIT IF HOST FAILURE
7920 032774 001333                BNE   16$        ;
7921
7922                               ; SET UP THE DHU TO PERFORM THE DMA.
7923                               ; -
7924
7925                               ; *
7926                               ; SET INTERNAL LOOPBACK, ENABLE THE RECIEVER FUNCTION ON THE LINE.
7927                               ; SET THE LPR ON THE LINE TO 38.4K BAUD, 8 BITS PER CHARACTER, ODD PARITY,
7928                               ; 2 STOP BITS, ENABLE THE TRANSMITTER ON THE LINE.
7929                               ; -
7930 032776 005267 152314      INC    ERRNBR     ;SET THE ERRNBR TO 4404
7931 033002 004767 164774      JSR    PC,FINACT  ;FIRST FIND AN ACTIVE LINE ON WHICH TO PERFORM
7932                               ; THE DMA,
7933 033006 010102                MOV    R1,R2      ;SAVE THE LINE NUMBER ON WHICH THE DMA WILL OCCUR
7934 033010 012701 010367      MOV    #EM4404,R1 ;SELECT THE MESSAGE,
7935                               ; "NO ACTIVE LINES , TEST ABANDONED"
7936 033014 103402                BCS   .,6        ;EXIT IF A LINE COULD NOT BE FOUND ,AFTER FIRST
7937 033016 000167 000424      JMP   30$        ;RESTORING THE CONTENTS OF MEMORY.
7938 033022 010201                MOV    R2,R1     ;RESTORE THE ACTIVE LINE NUMBER.
7939
7940                               ; *
7941                               ; AN ACTIVE LINE HAS BEEN FOUND
7942                               ; -
7942 033024 012700 000204      MOV    #204,R0   ;PASS THE LNCTRL CONTENTS
7943 033030 004767 174032      JSR    PC,WTWLNCR ;INITIALISE THE LNCTRL REGISTER
7944 033034 012700 177670      MOV    #177670,R0 ;PASS THE LPR CONTENTS
7945 033040 004767 174052      JSR    PC,WTWLPRL ;INITIALISE THE LPR REGISTER
7946 033044 004767 172360      JSR    PC,TXENBL  ;ENABLE TRANSMITTER ON THE LINE
7947
7948                               ; *
7949                               ; INITIATE THE DMA
7950                               ; -
7950 033050 016705 000626      MOV    ODISTA,R5  ;START FROM THE HIGHER OF THE PAIR OF ADDR.
7951 033054 012704 005130      MOV    #SDPBA5,R4 ;SET UP THE ADDR OF THE DATA PATTERN
7952 033060 010167 000610      MOV    R1,R0$    ;SAVE THE LINE NUMBER FOR THE DMA
7953 033064 012767 000002 000600  MOV    #2,70$    ;INITIALISE THE LOOP COUNT
7954
7955 033072 012700 000052 18$:  MOV    #5,R0     ;SET UP THE LSB'S
7956 033076 005003                CLR   R3         ;CLEAR THE REG THAT WILL HOLD THE 5 MSB'S
7957 033100 012702 000006      MOV    #6,R2     ;CONVERT THE DMTSTA INTO
7958 033104 006305 20$:  ASL   R5         ;A PHYSICAL ADDRESS WITH
    
```

```

7959 033106 006103          ROL    R3          ;THE MSB'S IN REG #3.
7960 033110 005302          DEC    R2          ;
7961 033112 001374          BNE    20$         ;
7962 033114 032705 000100    BIT    #100,R5     ;TEST BIT #6 OF THE DMTSTA
7963 033120 001402          BEQ    22$         ;
7964 033122 012700 000025    MOV    #25,R0      ;ALTER THE LSB'S IF BIT #6 WAS SET.
7965 033126 060005          ADD    R0,R5       ;ADD IN THE LSB'S
7966 033130 052703 000200    BIS    #200,R3     ;SET BIT #7.
7967
7968 033134 016777 000534 147036  MOV    80$,@CSRA   ;SELECT THE LINE ON WHICH TO PERFORM THE DMA.
7969
7970 033142 012767 010465 152146  MOV    #4405,ERRNBR ;SET ERROR NUMBER 4405
7971 033150 012701 010426    MOV    #EM4405,R1  ;SELECT THE MESSAGE.
7972
7973
7974 033154 105777 147034          TSTB   @TXAD2A     ;TEST THE DUT DMA-START BIT
7975 033160 100532          BMI    30$         ;EXIT WITH ERROR IF SET ,AFTER FIRST RESTORING
7976
7977 033162 012777 000020 147026  MOV    #16,@TXBFCA ;SET UP CHARACTER COUNT
7978 033170 010577 147016    MOV    R5,@TXAD1A ;SET UP BITS 0 TO 15 OF THE PHYSICAL ADDR.
7979 033174 110377 147014    MOVB   R3,@TXAD2A ;SET UP BITS 16 TO 21 , AND INITIATE THE DMA.
7980
7981
7982
7983 033200 012701 170144          ;*
7984 033204 016702 146770          ; WAIT FOR THE DMA TO COMPLETE AND THE LAST CHARACTER TO BE RECIEVED
7985
7986 033210 005267 152102          ;*
7987
7988 033214 004767 173572          MOV    #170144,R1  ;TEST BIT 15, TIME-OUT OF 100 MS.
7989 033220 012701 010522    MOV    CSRA,R2     ;PASS THE ADDR OF THE REG TO TEST.
7990
7991
7992 033224 103110          INC    ERRNBR      ;SET ERROR NUMBER TO 4406
7993
7994 033226 010402          JSR    PC,WAIBIS   ;WAIT FOR BIT TO SET
7995 033230 012704 000005    MOV    #EM4406,R1 ;SELECT THE MESSAGE.
7996 033234 004767 164216          ; " TIME-OUT OCCURED WAITING FOR DMA TO
7997 033240 010204          MOV    R4,R2       ;COMPLETE. TEST ABANDONED"
7998
7999
8000
8001 033242 005003          BCC    30$         ;EXIT IF TIME-OUT OCCURED, AFTER FIRST ,
8002 033244 012705 000200          ;RESTORING THE CONTENTS OF MEMORY.
8003
8004 033250 012767 010467 152040 24$:  MOV    #4,R4       ;SAVE R4
8005 033256 012701 010576    MOV    #5,R4       ;SET 5 MS DELAY
8006
8007
8008
8009 033262 017702 146714          JSR    PC,DELAY    ;DELAY TO ALLOW LAST CHARACTER TO BE RECIEVED
8010 033266 100067          MOV    R2,R4       ;RESTORE R4
8011
8012 033270 012700 170301          ;*
8013 033274 040200          ; READ THE CONTENTS OF THE RXFIFO AND COMPARE THEM WITH THE CORRECT DATA
8014 033276 001011          ;*
8015 033300 004767 171254          CLR    R3          ;CLEAR THE READ DATA COUNTER
                                MOV    #128,R5     ;SET THE MAX BMP CODE READ COUNT
                                MOV    #107,ERRNBR ;SET THE ERRNBR TO 4407
                                MOV    #EM4407,R1  ;SELECT THE MESSAGE.
                                ; " RXFIFO EMPTY TOO SOON, DMA FAILED
                                ;TEST ABANDONED"
                                MOV    @RBUFA,R2   ;READ THE CHARACTER FROM THE FIFO
                                BPL    30$         ;BRANCH TO REPORT ERROR IF FIFO EMPTY TOO SOON.
                                ;AFTER FIRST RESTORING THE CONTENTS OF MEMORY.
                                MOV    #170301,R0  ;SET UP BIT MASK OF A BMP CODE
                                BIC    R2,R0       ;TRY TO CLEAR THE BMP CODE MASK
                                BNE    28$         ;BRANCH IF NOT A BMP CODE.
                                JSR    PC,SAVBMP   ;SAVE THE BMP CODE ON THE QUEUE

```

```

8016
8017 033304 005267 152006      INC  ERRNBR      ;SET THE ERRNBR TO 4408
8018 033310 012701 010660      MOV  #EM4408,R1 ;SELECT THE MESSAGE,
8019                                     ; " TOO MANY BMP CODES FOUND IN THE RXFIFO,
8020                                     ;TEST ABANDONED"
8021
8022 033314 005305      DEC  R5          ;DEC THE MAX BMP CODE READ COUNT
8023 033316 001453      BEQ  30$        ;GO REPORT ERROR IF TOO MANY BMP CODES FOUND ,
8024                                     ;AFTER FIRST RESTORING THE CONTENTS OF MEMORY.
8025 033320 000753      BR   24$        ;DON'T COUNT THE BMP CODE AS A VALID CHARACTER
8026
8027 033322 012767 010471 151766 28$:  MOV  #4409,ERRNBR ;SET THE ERRNBR TO 4409
8028 033330 010201      MOV  R2,R1      ;SAVE THE CHARACTER FROM THE FIFO
8029 033332 012702 010723      MOV  #EM4409,R2 ;SELECT THE MESSAGE,
8030                                     ; " BAD BIT BETWEEN BITS 0 AND "
8031 033336 012767 015444 151756      MOV  #ER9101,ERRBLK ;SELECT THE ERROR ROUTINE.
8032 033344 012767 177777 000332      MOV  #-1,SUCSS  ;INDICATE 'BAD BITS' FAILURE
8033 033352 122401      CMPB (R4)+,R1   ;COMPARE CHAR FROM FIFO WITH THE CORRECT DATA.
8034 033354 001074      BNE  30$        ;BRANCH IF INCORRECT AND RESTORE MEM CONT'S.
8035 033356 005067 000322      CLR  SUCSS      ;INDICATE NON TEST SPECIFIC FAILURE E.G. TIME-OUTS
8036 033362 005203      INC  R3          ;COUNT THIS CHARACTER.
8037 033364 022703 000020      CMP  #16, R3    ;HAVE WE RECIEVED ALL THE CHARACTERS ?
8038 033370 001327      BNE  24$        ;LOOP UNTIL ALL CHARACTERS (NON-BMP) ARE READ.
8039 033372 005367 000274      DEC  70$        ;DECREMENT THE LOOP COUNT
8040 033376 001420      BEQ  29$        ;BRANCH IF BOTH DMA'S ARE COMPLETED
8041 033400 012704 005154      MOV  #SDP2B,R4  ;SET UP THE SECOND DATA PATTERN
8042 033404 016705 146612      MOV  DMTSTA,R5  ;SET UP THE OTHER DMA TEST ADDRESS
8043
8044 033410 012767 010472 151700      MOV  #4410,ERRNBR ;SET ERRNBR TO 4410
8045 033416 012701 010760      MOV  #EM4410,R1 ;SELECT THE MESSAGE,
8046                                     ; " RXFIFO FAILED TO PURGE, TEST ABANDONED "
8047 033422 012767 014032 151672      MOV  #ER0503,ERRBLK ;SELECT THE ERROR ROUTINE
8048
8049 033430 004767 167214      JSR  PC,PUFIFO  ;PURGE THE RXFIFO
8050 033434 103004      BCC  30$        ;EXIT WITH ERROR IF FIFO WOULD NOT PURGE,
8051                                     ;AFTER FIRST RESTORING THE CONTENTS OF MEMORY.
8052 033436 000615      BR   18$        ;OTHERWISE REPEAT.
8053
8054 033440 012767 000001 000236 29$:  MOV  #1,SUCSS   ;INDICATE THAT WE HAVE BEEN ABLE TO TEST,
8055                                     ;SOME OF THE BITS.
8056
8057                                     ; RESTORE THE ORIGINAL DATA IN THE MEMORY
8058                                     ;
8059
8060 033446 016767 146550 000224 30$:  MOV  DMTSTA,DUMY ;START WITH THE HIGHER OF THE PAIR OF DMTSTA
8061 033454 016767 000222 146540      MOV  DMTSTA,DMTSTA ;
8062 033462 012700 003602      MOV  #BLMBAS,R0 ;POINT AT THE START OF THE SAVED DATA AREA
8063 033466 012705 000001      MOV  #1,R5      ;SELECT WRITE TO (DMTSTA)
8064 033472 012703 000020      MOV  #16,R3     ;PASS NUMBER OF BYTES TO BE WRITTEN
8065 033476 004767 164066      JSR  PC,DMRW    ;RESTORE THE DATA
8066 033502 005767 146546      TST  TP4FLG     ;GO REPORT ERROR IF A TRAP OCCURED
8067 033506 001012      BNE  31$        ;
8068 033510 016767 000154 146504      MOV  DUMY,DMTSTA ;NOW RESTORE THE DATA FROM THE LOWER
8069                                     ;OF THE PAIR OF TEST ADDRESSES.
8070 033516 012700 004202      MOV  #B#MID,R0 ;POINT AT THE START OF THE SAVED DATA AREA
8071 033522 004767 164042      JSR  PC,DMRW    ;RESTORE THE DATA
8072

```

```

8073 033526 005767 146522          TST      TP4FLG      ;GO REPORT ANY ERRORS IF A NO TRAP
8074 033532 001411          BEQ      32$        ; OCCURED DURING THE RESTORE.
8075
8076 033534 012767 010473 151554 31$;      MOV      #4411,,ERRNBR ;SET THE ERROR NUMBER TO 4411
8077 033542 012701 011007          MOV      #EM4411,R1  ;SELECT THE MESSAGE
8078
8079
8080
8081 033546 012767 014032 151546          MOV      #ER0503,ERRBLK ; " HOST FAILURE. WRITE FAILURE TO AN ADDR
8082 033554 000433          BR       34$        ;WHICH HAD PREVIOUSLY BEEN SUCCESSFULLY
8083
8084
8085
8086
8087
8088 033556 005767 000122          32$:    TST      SUCSS      ;IF THE ERROR IS NON TEST SPECIFIC THEN
8089 033562 001430          BEQ      34$        ;BRANCH TO REPORT ERRORS
8090 033564 016701 000106          MOV      BITSTD,R1   ;LOAD THE NUMBER OF BITS TESTED
8091 033570 005301          DEC      R1         ;DEC TO GIVE THE BIT POSITION OF THE MSB TESTED.
8092 033572 022767 000001 000104          CMP      #1,SUCSS   ;IF THE BITS TESTED ARE BAD THEN
8093 033600 001021          BNE     34$        ;BRANCH AND REPORT ERRORS.
8094
8095
8096
8097 033602 032767 000040 146352          BIT      #BIT05,OPTION ; PRINT THE BITS TESTED IF THE SOFTWARE
8098 033610 001416          BEQ     60$        ; OPTION HAS REQUESTED IT
8099 033612 010102          MOV     R1,R2     ; CALCULATE THE NUMBER OF BITS WHICH HAVE
8100 033614 005202          INC     R2        ; BEEN TESTED SUCCESSFULLY.
8101 033616          PRINTB #EF4401,R1,R2 ;PRINT THE NUMBER OF BITS TESTED MESSAGE.
8101 033616 010246          MOV     R2,-(SP)
8101 033620 010146          MOV     R1,-(SP)
8101 033622 012746 005631          MOV     #EF4401,-(SP)
8101 033626 012746 000003          MOV     #3,-(SP)
8101 033632 010600          MOV     SP,RO
8101 033634 104414          TRAP   C$PNTB
8101 033636 062706 000010          ADD     #10,SP
8102 033642 000401          BR      60$        ;EXIT THE TEST
8103
8104 033644          34$:    ERROR      ; REPORT ERRORS
8104 033644 104460          TRAP   C$ERROR
8105 033646          60$:    SETPRI   #PRI05 ;ENABLE THE CLOCK
8105 033646 012700 000240          MOV     #PRI05,RO
8105 033652 104441          TRAP   C$SPRI
8106 033654 016767 146376 144122          MOV     TP4VEC,4   ;RESTORE THE NORMAL 004 TRAP VECTOR
8107 033662 005067 146332          CLR     CTRL.CF   ;INDICATE THAT WE ARE NOT WITHIN A TEST
8108
8109 033666          EXIT    TST
8109 033666 104432          TRAP   C$EXIT
8109 033670 000016          .WORD  L10030-.
8110
8111
8112
8113 033672 000000          ; ***** LOCAL VARIABLE AREA *****
8113 033672 000000          70$:    .WORD  0   ;COUNTER FOR THE NUMBER OF DMA'S COMPLETED
8114 033674 000000          80$:    .WORD  0   ;SAVE AREA FOR THE ACTIVE LINE NUMBER
8115 033676 000000          BITSTD: .WORD  0   ;NUMBER OF BITS TESTED
8116 033700 000000          DUM1:   .WORD  0   ;DUMMY VARIABLE
8117 033702 000000          UDTSTA: .WORD  0   ;HIGHER OF THE PAIR OF COMPLEMENTARY ADDR.

```

PROGRAM DOCUMENTB1	GLOBAL ERROR REPORTI....B5	GLOBAL SUBROUTINEB9	GLOBAL SUBROUTINEB13	
PROGRAM DOCUMENTC1	GLOBAL ERROR REPORTI....C5	GLOBAL SUBROUTINEC9	GLOBAL SUBROUTINEC13	
PROGRAM DOCUMENTD1	GLOBAL ERROR REPORTI....D5	GLOBAL SUBROUTINED9	GLOBAL SUBROUTINED13	
PROGRAM DOCUMENTE1	GLOBAL ERROR REPORTI....E5	GLOBAL SUBROUTINEE9	GLOBAL SUBROUTINEE13	
PROGRAM DOCUMENTF1	GLOBAL ERROR REPORTI....F5	GLOBAL SUBROUTINEF9	GLOBAL SUBROUTINEF13	
PROGRAM DOCUMENTG1	GLOBAL ERROR REPORTI....G5	GLOBAL SUBROUTINEG9	GLOBAL SUBROUTINEG13	
PROGRAM DOCUMENTH1	GLOBAL ERROR REPORTI....H5	GLOBAL SUBROUTINEH9	GLOBAL SUBROUTINEH13	
PROGRAM DOCUMENTI1	GLOBAL ERROR REPORTI....I5	GLOBAL SUBROUTINEI9	GLOBAL SUBROUTINEI13	
PROGRAM DOCUMENTJ1	GLOBAL ERROR REPORTI....J5	GLOBAL SUBROUTINEJ9	GLOBAL SUBROUTINEJ13	
PROGRAM DOCUMENTK1	GLOBAL ERROR REPORTI....K5	GLOBAL SUBROUTINEK9	GLOBAL SUBROUTINEK13	
PROGRAM DOCUMENTL1	GLOBAL ERROR REPORTI....L5	GLOBAL SUBROUTINEL9	GLOBAL SUBROUTINEL13	
PROGRAM DOCUMENTM1	GLOBAL ERROR REPORTI....M5	GLOBAL SUBROUTINEM9	GLOBAL SUBROUTINEM13	
PROGRAM DOCUMENTN1	GLOBAL ERROR REPORTI....N5	GLOBAL SUBROUTINEN9	GLOBAL SUBROUTINEN13	
PROGRAM DOCUMENTB2	GLOBAL ERROR REPORTI....B6	GLOBAL SUBROUTINEB10	GLOBAL SUBROUTINEB14	
PROGRAM DOCUMENTC2	GLOBAL ERROR REPORTI....C6	GLOBAL SUBROUTINEC10	GLOBAL SUBROUTINEC14	
PROGRAM DOCUMENTD2	GLOBAL SUBROUTINES S....D6	GLOBAL SUBROUTINED10	GLOBAL SUBROUTINED14	
PROGRAM DOCUMENTE2	GLOBAL SUBROUTINEE6	GLOBAL SUBROUTINEE10	GLOBAL SUBROUTINEE14
PROGRAM DOCUMENTF2	GLOBAL SUBROUTINEF6	GLOBAL SUBROUTINEF10	GLOBAL SUBROUTINEF14
PROGRAM DOCUMENTG2	GLOBAL SUBROUTINEG6	GLOBAL SUBROUTINEG10	GLOBAL SUBROUTINEG14
PROGRAM DOCUMENTH2	GLOBAL SUBROUTINEH6	GLOBAL SUBROUTINEH10	GLOBAL SUBROUTINEH14
PROGRAM DOCUMENTI2	GLOBAL SUBROUTINEI6	GLOBAL SUBROUTINEI10	GLOBAL SUBROUTINEI14
PROGRAM DOCUMENTJ2	GLOBAL SUBROUTINEJ6	GLOBAL SUBROUTINEJ10	GLOBAL SUBROUTINEJ14
PROGRAM DOCUMENTK2	GLOBAL SUBROUTINEK6	GLOBAL SUBROUTINEK10	GLOBAL SUBROUTINEK14
PROGRAM HEADERL2	GLOBAL SUBROUTINEL6	GLOBAL SUBROUTINEL10	GLOBAL SUBROUTINEL14
PROGRAM HEADERM2	GLOBAL SUBROUTINEM6	GLOBAL SUBROUTINEM10	GLOBAL SUBROUTINEM14
DISPATCH TABLEN2	GLOBAL SUBROUTINEN6	GLOBAL SUBROUTINEN10	GLOBAL SUBROUTINEN14
DISPATCH TABLEB3	GLOBAL SUBROUTINEB7	GLOBAL SUBROUTINEB11	GLOBAL SUBROUTINEB15
DEFAULT HARDWARE P-T....C3C3	GLOBAL SUBROUTINEC7	GLOBAL SUBROUTINEC11	GLOBAL SUBROUTINEC15
SOFTWARE P-TABLED3	GLOBAL SUBROUTINED7	GLOBAL SUBROUTINED11	GLOBAL SUBROUTINED15
GLOBAL EQUATES SECTI....E3E3	GLOBAL SUBROUTINEE7	GLOBAL SUBROUTINEE11	GLOBAL SUBROUTINEE15
GLOBAL EQUATES SECTI....F3F3	GLOBAL SUBROUTINEF7	GLOBAL SUBROUTINEF11	GLOBAL SUBROUTINEF15
GLOBAL DATA SECTIONG3	GLOBAL SUBROUTINEG7	GLOBAL SUBROUTINEG11	GLOBAL SUBROUTINEG15
GLOBAL DATA SECTIONH3	GLOBAL SUBROUTINEH7	GLOBAL SUBROUTINEH11	GLOBAL SUBROUTINEH15
GLOBAL DATA SECTIONI3	GLOBAL SUBROUTINEI7	GLOBAL SUBROUTINEI11	GLOBAL SUBROUTINEI15
GLOBAL DATA SECTIONJ3	GLOBAL SUBROUTINEJ7	GLOBAL SUBROUTINEJ11	GLOBAL SUBROUTINEJ15
GLOBAL DATA SECTIONK3	GLOBAL SUBROUTINEK7	GLOBAL SUBROUTINEK11	GLOBAL SUBROUTINEK15
GLOBAL DATA SECTIONL3	GLOBAL SUBROUTINEL7	GLOBAL SUBROUTINEL11	GLOBAL SUBROUTINEL15
GLOBAL DATA SECTIONM3	GLOBAL SUBROUTINEM7	GLOBAL SUBROUTINEM11	GLOBAL SUBROUTINEM15
GPR HANDLING ROUTINE....N3N3	GLOBAL SUBROUTINEN7	GLOBAL SUBROUTINEN11	GLOBAL SUBROUTINEN15
GPR FRAME ACCESS EQU....B4B4	GLOBAL SUBROUTINEB8	GLOBAL SUBROUTINEB12	GLOBAL SUBROUTINEB16
GLOBAL MACRO DEFINIT....C4C4	GLOBAL SUBROUTINEC8	GLOBAL SUBROUTINEC12	GLOBAL SUBROUTINEC16
GLOBAL MACRO DEFINIT....D4D4	GLOBAL SUBROUTINED8	GLOBAL SUBROUTINED12	GLOBAL SUBROUTINED16
GLOBAL SUBROUTINEE4	GLOBAL SUBROUTINEE8	GLOBAL SUBROUTINEE12	GLOBAL SUBROUTINEE16
GLOBAL TEXT SECTIONF4	GLOBAL SUBROUTINEF8	GLOBAL SUBROUTINEF12	GLOBAL SUBROUTINEF16
GLOBAL TEXT SECTIONG4	GLOBAL SUBROUTINEG8	GLOBAL SUBROUTINEG12	GLOBAL SUBROUTINEG16
GLOBAL TEXT SECTIONH4	GLOBAL SUBROUTINEH8	GLOBAL SUBROUTINEH12	GLOBAL SUBROUTINEH16
GLOBAL TEXT SECTIONI4	GLOBAL SUBROUTINEI8	GLOBAL SUBROUTINEI12	GLOBAL SUBROUTINEI16
GLOBAL TEXT SECTIONJ4	GLOBAL SUBROUTINEJ8	GLOBAL SUBROUTINEJ12	GLOBAL SUBROUTINEJ16
GLOBAL ERROR REPORTI....K4K4	GLOBAL SUBROUTINEK8	GLOBAL SUBROUTINEK12	GLOBAL SUBROUTINEK16
GLOBAL ERROR REPORTI....L4L4	GLOBAL SUBROUTINEL8	GLOBAL SUBROUTINEL12	GLOBAL SUBROUTINEL16
GLOBAL ERROR REPORTI....M4M4	GLOBAL SUBROUTINEM8	GLOBAL SUBROUTINEM12	GLOBAL SUBROUTINEM16
GLOBAL ERROR REPORTI....N4N4	GLOBAL SUBROUTINEN8	GLOBAL SUBROUTINEN12	GLOBAL SUBROUTINEN16

B1

8118 033704 000000
8119
8120
8121
8122
8123
8124 033706
033706
033706 104401

SUCSS: .WORD 0

,SUCCESS INDICATOR, -1 - ERROR DUE TO BAD BITS
1 - SUCCESSFUL TEST
0 - OTHER ERRORS

***** END *****

ENDTST

L10030: TRAP C#ETST

```

8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141 033710
      033710
8142
8143
8144
8145 033710 126727 146260 000002
8146 033716 001402
8147
8148 033720 000167 000372
8149 033724
      033724 012700 000240
      033730 104441
8150 033732 012767 177777 146260
8151      000005
8152 033740 012767 000005 146312
8153 033746 012767 000001 151340
8154 033754 012767 014071 151334
8155 033762 012767 011054 151330
8156 033770 005067 146304
8157 033774 005067 146224
8158
8159
8160
8161
8162
8163 034000 004767 163332
8164 034004 103144
8165
8166
8167
8168
8169 034006
      034006 012700 000340
      034012 104441
8170 034014
      034014 012746 000300
      034020 012746 027426
      034024 016746 146140
      034030 012746 000003
      034034 104437
      034036 062706 000010
8171 034042

```

```

.SBTTL  HARDWARE TEST - FRMERR -
*****
- FRAMING ERROR GENERATION TEST -
*****
THIS TEST IS USED TO VERIFY THE FRAMING ERROR DETECTION CAPABILITIES
OF THE DHU11.
WHEN IN STAGGARED LOOPBACK MODE, CHARACTERS ARE TRANSMITTED FROM
ONE GROUP OF LINES AT 8 BITS/CHAR, AND RECEIVED BY THE OTHER GROUP
AT 5 BITS/CHAR. THIS WILL GENERATE A FRAMING ERROR FOR EACH CHARACTER.
THIS TEST WILL ONLY EXECUTE IF THE STAGGARED LOOPBACK MODE IS SELECTED.
THE SPECIAL STAGGARED LOOPBACK BERG CONNECTOR MUST BE FITTED.
THE ACTIVE LINES BIT MASK IS USED TO INDICATE WHICH LINES HAVE BEEN
REMOVED FROM FURTHER TESTING.
*****
      BGNTST
      T5::
EXECUTE THIS TEST IN STAGGARED LOOPBACK MODE ONLY.
      CMPB  LOPBCK, #2      ;CHECK MODE SELECTED.
      BEQ   .+6            ;AVOID EXITING THE TEST IF STAGGERED LOOPBACK
                          ;MODE IS SELECTED.
      JMP   601            ;EXIT THE TEST.
      SETPRI #PRI05        ;ALLOW LTC INTERRUPTS.
                          MOV   #PRI05, RO
                          TRAP  C1SPRI
      MOV   #-1, CTRLCF    ;INDICATE THAT WE ARE IN A TEST.
      TNUM = TNUM + 1      ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV   #TNUM, TSTNUM  ;SET UP THE TEST NUMBER. (62)
      MOV   #1, ERR1TYP    ;SET ERROR TYPE IN ERROR TABLE.
      MOV   #6201, ERR1NBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV   #EM6201, ERR1MSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
      CLR   ERS1MRF        ;INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
      CLR   FERROR        ;CLEAR THE "AT LEAST ONE ERROR" INDICATOR.
;
; RESET THE DUT TO A KNOWN STATE, REMOVE STATUS CODES FROM THE FIFO.
; CLEAR TX AND RX INTERRUPT ENABLE BITS.
; THIS SUBROUTINE REPORTS ERROR >>>> 6201 <<<<<.
      JSR   PC, CLNRST     ;RESET THE DUT.
      BCC   601            ;ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;
; DISABLE ALL INTERRUPTS.
; SET UP DMA TX AND RX INTERRUPT SERVICE ROUTINES.
      SETPRI #PRI07        ;DISABLE ALL INTERRUPTS.
                          MOV   #PRI07, RO
                          TRAP  C1SPRI
      SETVEC TXVECA, #TXDMA, #PRI06 ;SELECT DMA TX INT SERVICE RTN.
                          MOV   #PRI06, -(SP)
                          MOV   #TXDMA, -(SP)
                          MOV   TXVECA, -(SP)
                          MOV   #3, -(SP)
                          TRAP  C1SVEC
                          ADD   #10, SP
      SETPRI #PRI04        ;ALLOW INTERRUPTS.

```

	034042	012700	000200				MOV	#PRT04,R0	
	034046	104441					TRAP	C+SPRI	
8172									
8173									
8174									
8175	034050	005067	146426				CLR	TXDNF	;CLEAR TX DONE FLAGS FOR ALL LINES.
8176	034054	005067	146424				CLR	RXDNF	;CLEAR RX DONE FLAGS FOR ALL LINES.
8177	034060	005067	146200				CLR	TXINTF	;CLEAR TX ERROR FLAGS FOR ALL LINES.
8178									
8179									
8180									
8181									
8182									
8183	034064	012700	003302				MOV	#ERCNTB,R0	;PASS THE ADDRESS OF THE TABLE TO BE CLEARED.
8184	034070	004767	163264				JSR	PC,CLR16W	;CLEAR THE RX ERROR COUNTERS TABLE.
8185	034074	005067	147502				CLR	BUFBAS	;SET SINGLE CHAR DATA TO BE A NULL.
8186									
8187									
8188									
8189									
8190									
8191	034100	012700	156470				MOV	#156470,R0	;PASS LPR PARAMETER FOR 8 BITS/CHAR.
8192	034104	012701	156440				MOV	#156440,R1	;PASS LPR PARAMETER FOR 5 BITS/CHAR.
8193	034110	004767	164422				JSR	PC,GETTIM	;GET TIME-OUT BASED ON MINIMUM BAUDRATE IN USE.
8194	034114	012702	003602				MOV	#BUFBAS,R2	;PASS START ADDRESS OF DATA PATTERN.
8195	034120	012703	000001				MOV	#1,R3	;PASS LENGTH OF DATA PATTERN.
8196	034124	016704	146106				MOV	LGRP1M,R4	;PASS LINE GROUP OF LINES THAT ARE TO TX.
8197	034130	004767	163726				JSR	PC,FRPSUP	;SET UP DUT FOR TRANSMISSION AND RECEPTION.
8198									
8199									
8200									
8201									
8202									
8203									
8204									
8205	034134	005267	151156				INC	ERRNBR	;SET THE ERROR REPORT NUMBER TO 6202.
8206	034140	004767	166566				JSR	PC,PUFIFR	;CLEAN OUT THE FIFO.
8207	034144	103064					BCC	601	;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8208	034146	012767	014075	151142			MOV	#6205,ERRNBR	;SET THE ERROR NUMBER TO 6205.
8209	034154	004767	171344				JSR	PC,TXFRPR	;TX DATA PATTERN ON SELECTED ACTIVE LINES.
8210	034160	012705	100000				MOV	#100000,R5	;PASS FRAMING ERROR TEST FLAG.
8211									
8212									
8213									
8214	034164	004767	162546				JSR	PC,CKFRPR	;READ CHARACTERS, REPORT ANY ERRORS FOUND.
8215									
8216	034170	005767	146030				TST	FERROR	;HAS AN ERROR BEEN DETECTED?
8217	034174	001404					BEQ	21	;BRANCH IF NO ERROR
8218	034176	032767	000100	145756			BIT	#BIT06,OPTION	;HAS EXTENDED ERROR REPORTING BEEN ENBL'D?
8219	034204	001436					BEQ	541	;BRANCH IF IT HASN'T AND EXIT THE TEST. THE
8220									;TEST FAILURE MESSAGE HAS BEEN PRINTED.
8221									
8222									
8223									
8224	034206	005104					COM	R4	;REVERSE ROLES FOR TRANSMISSION AND RECEPTION.
8225	034210	004767	163646				JSR	PC,FRPSUP	;SET UP DUT FOR TRANSMISSION AND RECEPTION.
8226	034214	005267	151076				INC	ERRNBR	;SET ERROR NUMBER TO 6206.

```

8227
8228
8229
8230 034220 004767 166506
8231 034224 103034
8232 034226 012767 014101 151062
8233 034234 004767 171264
8234 034240 012705 100000
8235
8236
8237
8238 034244 004767 162466
8239
8240 034250 005767 145750
8241 034254 001404
8242 034256 032767 000100 145676
8243 034264 001406
8244
8245
8246 034266 005267 151024
8247
8248
8249
8250
8251
8252
8253 034272 004767 171322
8254 034276 004767 171744
8255
8256 034302
      034302 012700 000340
      034306 104441
3257 034310
      034310 016700 145654
      034314 104436
8258
8259 034316 005067 145676
8260
8261 034322
      034322
      034322 104401

; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6206 THRU 6208 <<<<<.
;
; JSR PC,PUFIFR ;CLEAN OUT THE FIFO.
; BCC 601 ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
; MOV #6209,ERRNBR ;SET ERROR NUMBER TO 6209.
; JSR PC,TXRPR ;TX DATA PATTERN ON SELECTED ACTIVE LINES.
; MOV #10000,R5 ;PASS FRAMING ERROR TEST FLAG.

; THIS SUBROUTINE REPORTS ERRORS >>>> 6209 <<<<<.
;
; JSR PC,CKFRPR ;READ CHARACTERS, REPORT ANY ERRORS FOUND.
;
; TST FERROR ;HAS AN ERROR BEEN DETECTED?
; BEQ 41 ;BRANCH IF NO ERROR
; BIT #BIT05,OPTION ;HAS EXTENDED ERROR REPORTING BEEN ENBL'D?
; BEQ 541 ;BRANCH IF IT HASN'T AND EXIT THE TEST. THE
; ;TEST FAILURE MESSAGE HAS BEEN PRINTED.

41: INC ERRNBR ;SET ERROR NUMBER TO 6210.

; DISABLE INTERRUPTS.
; CLEAR THE INTERRUPT VECTORS.
; UPDATE THE ACTIVE LINES BIT MAP TO REFLECT LINES REMOVED FROM TESTING.
; THIS SUBROUTINE REPORTS ERRORS >>>> 6210 THRU 6212 <<<<<.
;
; JSR PC,TXIE0 ;DISABLE ALL TX INTERRUPTS.
; JSR PC,TXRREP ;REPORT FINAL ERRORS FROM TX/RX.

541: SETPRI #PRI07 ;DISABLE ALL INTERRUPTS.
;
; MOV #PRI07,R0
; TRAP C$SPRI
;
; CLRVEC TXVECA ;RETURN TX INT VECTOR TO UNUSED POOL.
; MOV TXVECA,R0
; TRAP C$CVEC

601: CLR CTRLCF ;INDICATE THAT WE ARE NOT WITHIN A TEST.

ENDTST

L10031: TRAP C$ETST

```

```

8263
8264
8265
8266
8267
8268
8269
8270
8271
8272
8273
8274
8275
8276
8277
8278
8279 034324
      034324
8280
8281
8282
8283 034324 126727 145644 000002
8284 034332 001402
8285
8286 034334 000167 000434
8287
8288 034340
      034340 012700 000240
      034344 104441
8289 034346 012767 177777 145644
8290
      000006
8291 034354 012767 000006 145676
8292 034362 012767 000001 150724
8293 034370 012767 014235 150720
8294 034376 012767 011115 150714
8295 034404 005067 146070
8296 034410 005067 145610
8297
8298
8299
8300
8301
8302 034414 004767 162716
8303 034420 103165
8304
8305
8306
8307
8308 034422
      034422 012700 000340
      034426 104441
8309 034430
      034430 012746 000300
      034434 012746 027426
      034440 016746 145524
      034444 012746 000003
      034450 104437

```

```

.SBTL  HARDWARE TEST          - PARERR -
*****
- PARITY ERROR GENERATION TEST -
*
* THIS TEST IS USED TO VERIFY THE PARITY ERROR DETECTION AND REPORT
* CAPABILITIES OF THE DUT.
* WHEN STAGGARED LOOPBACK MODE IS SELECTED, DATA IS TRANSMITTED
* ON ALL ACTIVE LINES IN LINE GROUP 1 WITH ODD PARITY SELECTED,
* AND RECEIVED ON LINES IN GROUP 2 WITH EVEN PARITY SELECTED.
* THIS WILL GENERATE A PARITY ERROR FOR EACH CHARACTER RECEIVED.
* THE PARITY SELECTION IS THEN REVERSED ON THE LINES IN EACH GROUP
* AND THE TEST IS REPEATED.
* THIS TEST WILL ONLY EXECUTE IF THE STAGGARED LOOPBACK MODE IS SELECTED.
* THE SPECIAL STAGGARED LOOPBACK BERG CONNECTOR MUST BE FITTED.
*
*****
      BGNTST
*
* EXECUTE THIS TEST IN STAGGARED LOOPBACK MODE ONLY.
*
      CMPB  LOPBCK,#2      ;CHECK MODE SELECTED.
      BEQ   .+6           ;AVOID EXITING THE TEST IF STAGGERED LOOPBACK
                          ;MODE IS SELECTED.
      JMP   60$          ; EXIT THE TEST.
*
      SETPRI #PRI05      ;ALLOW LTC INTERRUPTS.
                          MOV   #PRI05,R0
                          TRAP  C$SPRI
*
      MOV   #-1,CTRLCF   ;INDICATE THAT WE ARE IN A TEST.
      TNUM += TNUM + 1   ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV   #TNUM,TSTNUM ;SET UP THE TEST NUMBER. (63)
      MOV   #1,ERRTYP    ;SET ERROR TYPE IN ERROR TABLE.
      MOV   #6301,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
      MOV   #EM6301,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
      CLR   ERSRWF       ;INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
      CLR   FERROR      ;CLEAR THE "AT LEAST ONE ERROR" INDICATOR
*
; RESET THE DUT TO A KNOWN STATE, REMOVE STATUS CODES FROM THE FIFO.
; CLEAR TX AND RX INTERRUPT ENABLE BITS.
; THIS SUBROUTINE REPORTS ERROR >>>> 6301 <<<<<.
*
      JSR   PC,CLNRST    ;RESET THE DUT.
      BCC   60$         ;ABOUT THE TEST IF FATAL ERROR FOUND IN RESET.
*
; DISABLE ALL INTERRUPTS.
; SET UP DMA TX AND RX INTERRUPT SERVICE ROUTINES.
*
      SETPRI #PRI07      ;DISABLE ALL INTERRUPTS.
                          MOV   #PRI07,R0
                          TRAP  C$SPRI
      SETVEC TXVECA,#TXDMA,#PRI06 ;SELECT DMA TX INT SERVICE RTN.
                          MOV   #PRI06,-(SP)
                          MOV   #TXDMA,-(SP)
                          MOV   TXVECA,-(SP)
                          MOV   #3,-(SP)
                          TRAP  C$SVEC

```

61

```

8310 034452 062706 000010          ADD    #10,SP
      034456          SETPRI #PRT04          ;ALLOW INTERRUPTS.
      034456 012700 000200          MOV    #PRT04,R0
      034462 104441          TRAP   C$SPRI

8311          ;*
8312          ;* CLEAR TX/RX FLAGS.
8313          ;*
8314 034464 005067 146012          CLR    TXDNF          ;CLEAR TX DONE FLAGS FOR ALL LINES.
8315 034470 005067 146010          CLR    RXDNF          ;CLEAR RX DONE FLAGS FOR ALL LINES.
8316 034474 005067 145564          CLR    TXINTF        ;CLEAR TX ERROR FLAGS FOR ALL LINES.
8317          ;*
8318          ;* SET UP ERROR COUNTER TABLE.
8319          ;*
8320 034500 012700 003302          MOV    #ERCNTB,R0     ;PASS THE ADDRESS OF THE TABLE TO BE CLEARED.
8321 034504 004767 162650          JSR    PC,CLR16W      ;CLEAR THE RX ERROR COUNTERS TABLE.
8322          ;*
8323          ;* INITIALISE DMA PARAMETERS IN THE CONTROL BLOCK.
8324          ;*
8325          ;*
8326 034510 012700 156470          MOV    #156470,R0     ;PASS LPR PARAMETER WITH ODD PARITY.
8327 034514 012701 156570          MOV    #156570,R1     ;PASS LPR PARAMETER WITH EVEN PARITY.
8328 034520 004767 164012          JSR    PC,GETTIM      ;GET TIME OUT BASED ON MINIMUM BAUDRATE IN USE.
8329 034524 012702 005154          MOV    #SOP2B,R2     ;PASS START ADDRESS OF DATA PATTERN.
8330 034530 012703 000020          MOV    #16.,R3       ;PASS LENGTH OF DATA PATTERN.
8331 034534 016704 145476          MOV    LGRP1M,R4     ;PASS BIT MAP OF LINES TO BE SET WITH ODD PAR.
8332 034540 004767 163316          JSR    PC,FRPSUP     ;SET UP DUT FOR TRANSMISSION AND RECEPTION.
8333          ;*
8334          ;*
8335          ;* PURGE THE FIFO OF ANY UNWANTED CHARACTERS.
8336          ;* PERFORM TRANSMISSION AND RECEPTION OF THE 16 BYTE DATA PATTERN AT 9600 BAUD.
8337          ;* TRANSMISSION ON LINE IN GROUP 1, 8 BITS/CHAR, 1 STOP BITS, ODD PARITY.
8338          ;* RECEPTION ON LINES IN GROUP 2 AT 8 BITS/CHAR, 1 STOP, EVEN PARITY.
8339          ;* REMOVE CHARACTERS FROM THE FIFO AND LOOK FOR THE PARITY ERROR BIT BEING SET.
8340          ;* REPORT ANY ERRORS FOUND, IF FRAMMING ERROR BIT SET OR PARITY ERROR CLEAR.
8341          ;*
8342          ;*
8343          ;* THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6302 THRU 6304 <<<<.
8344          ;*
8345 034544 004767 166162          JSR    PC,PURFER      ;CLEAN OUT THE FIFO.
8346 034550 103111          BCC    #0             ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8347 034552 012767 014741 150536    MOV    #6305.,ERRNBR  ;SET ERROR NUMBER TO 6305
8348 034560 004767 164030          JSR    PC,INIDMA     ;TX DATA PATTERN ON ALL ACTIVE LINES.
8349 034564 005005          CLR    #1             ;PASS PARITY ERROR TEST FLAG.
8350          ;*
8351          ;* THIS SUBROUTINE REPORTS ERROR NUMBER >>>> 6305 <<<<.
8352          ;*
8353 034566 004767 162144          JSR    PC,ERRRPR     ;READ CHARACTERS, REPORT ANY ERRORS FOUND.
8354          ;*
8355 034572 005767 145476          TST    #ERROR        ;HAS AN ERROR BEEN FOUND?
8356 034576 001404          BEQ    #0             ;BRANCH TO CONTINUE IF IT HASN'T.
8357 034600 032767 000100 145354    BIT    #BIT08,OPTION  ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
8358 034606 001457          BEQ    #54$          ;EXIT THE TEST IF IT HASN'T, THE TEST FAILURE.
8359          ;* MESSAGE HAS ALREADY BEEN REPORTED.
8360          ;*
8361 034610 005267 150502          INC    #ERRNBR       ;SET ERROR NUMBER TO 6306.
8362          ;*
8363          ;* THIS SUBROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6306 THRU 6309 <<<<
    
```

```

8364
8365 034614 004767 171426      JSR    PC, TXRREP      ;REPORT FINAL ERRORS FROM TX/RX.
8366 034620 005767 145400      TST    FERROR        ;HAS AN ERROR BEEN FOUND ?
8367 034624 001404                BEQ    4$             ;BRANCH TO CONTINUE IF IT HASN'T.
8368 034626 032767 000100 145326  BIT    @BIT06, OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8369 034634 001457                BEQ    60$           ;EXIT THE TEST IF IT HASN'T. THE TEST FAILURE
8370                                ;MESSAGE HAS ALREADY BEEN REPORTED.
8371
8372 034636 012767 014246 150452 4$:  MOV    @6310., ERRNBR ;SET ERROR NUMBER TO 6310.
8373 034644 005067 145632      CLR    TXDONF        ;CLEAR TX DONE FLAGS FOR ALL LINES.
8374 034650 005067 145630      CLR    RXDONF        ;CLEAR RX DONE FLAGS FOR ALL LINES.
8375 034654 005067 145404      CLR    TXINTF        ;CLEAR TX DMA HANDOVER ERROR FLAGS.
8376
8377                                ; REVERSE TRANSMISSION/RECEPTION ROLES ON ALL ACTIVE LINES, AND REPEAT TEST.
8378
8379 034660 005104                COM    R4             ;REVERSE ROLES FOR TRANSMISSION AND RECEPTION.
8380 034662 004767 163174      JSR    PC, FRPSUP     ;SET UP OUT FOR TRANSMISSION AND RECEPTION.
8381
8382                                ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6310 THRU 6311 <<<<.
8383
8384 034666 004767 166040      JSR    PC, PUFIFR     ;CLEAN OUT THE FIFO.
8385 034672 103040                BCC    60$           ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8386 034674 012767 014250 150414  MOV    @6312., ERRNBR ;SET ERROR NUMBER TO 6312.
8387 034702 004767 163706      JSR    PC, INIDMA     ;TX DATA PATTERN ON SELECTED ACTIVE LINES.
8388
8389                                ; THIS SUBROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6312 THRU 6316 <<<<.
8390
8391 034706 004767 162024      JSR    PC, CERRPR     ;READ CHARACTERS. REPORT ANY ERRORS FOUND.
8392 034712 005767 145306      TST    FERROR        ;HAS AN ERROR BEEN FOUND ?
8393 034716 001404                BEQ    6$             ;BRANCH TO CONTINUE IF IT HASN'T.
8394 034720 032767 000100 145234  BIT    @BIT06, OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8395 034726 001407                BEQ    54$           ;EXIT THE TEST IF IT HASN'T. THE TEST FAILURE
8396                                ;MESSAGE HAS ALREADY BEEN REPORTED.
8397
8398 034730 012767 014255 150360 6$:  MOV    @6317., ERRNBR ;SET ERROR NUMBER TO 6317.
8399
8400                                ; DISABLE INTERRUPTS.
8401                                ; CLEAR THE INTERRUPT VECTORS.
8402                                ; UPDATE THE ACTIVE LINES BIT MAP TO REFLECT LINES REMOVED FROM TESTING.
8403
8404 034736 004767 170656      JSR    PC, TXIF0      ;DISABLE ALL TX INTERRUPTS.
8405
8406                                ; THIS SUBROUTINE REPORTS ERRORS >>>> 6317 THRU 6320 <<<<.
8407
8408 034742 004767 171300      JSR    PC, TXRREP     ;REPORT FINAL ERRORS FROM TX/RX.
8409
8410 034746 54$:  SETPRI @PRI07        ;DISABLE ALL INTERRUPTS.
8411 034746 012700 000340                MOV    @PRI07, RO    ;
8412 034752 104441                TRAP   C$SPRT       ;
8413 034754                CURVEC TXVEC0      ;RETURN TX INT VECTOR TO UNUSED P000.
8414 034754 012700 145210                MOV    TXVEC0, RO    ;
8415 034760 104436                TRAP   C$CVFC       ;
8416
8417
8418                                ; THIS SUBROUTINE REPORTS ERRORS WITH NUMBERS >>>> 6321 <<<<.
8419
8420 034762 012767 014261 150326  MOV    @6321., ERRNBR ;SET ERROR NUMBER TO 6321.

```

11

8417	034770	004767	167034	JSR	PC,REPSMR
8418					
8419	034774	005067	145220	60\$:	CLR CTRLCF
8420	035000				ENDTST
	035000				
	035000	104401			

;REPORT ERROR SUMMARIES IF CALLED FOR.
;INDICATE THAT WE ARE NOT WITHIN A TEST.

L10032: TRAP C\$ETST


```

8422
8423
8424
8425
8426
8427
8428
8429
8430
8431
8432
8433
8434
8435 035002
      035002
8436 035002 012700 000240
      035006 104441
8437 000007
8438 035010 012767 000007 145242
8439 035016 012767 177777 145174
8440 035024 012767 000001 150262
8441 035032 012767 021615 150256
8442 035040 012767 012335 150252
8443 035046 005067 145426
8444 035052 005067 145146
8445
8446
8447
8448
8449
8450 035056 004767 162254
8451 035062 103402
8452 035064 000167 000664
8453
8454
8455
8456 035070
      035070 012700 000340
      035074 104441
8457 035076
      035076 012746 000300
      035102 012746 027426
      035106 016746 145026
      035112 012746 000003
      035116 104437
      035120 062706 000010
8458 035124
      035124 012746 000300
      035130 012746 027216
      035134 016746 145026
      035140 012746 000003
      035144 104437
      035146 062706 000010
8459 035152
      035152 012700 000200
      035156 104441

```

```

SUBTTL HARDWARE TEST DMA
*****
DMA MODE TEST -
THIS TEST VERIFIES THAT THE DEVICE UNDER TEST (DUT) WILL PERFORM
TRANSMISSION AND RECEPTION CORRECTLY USING THE DMA MODE TRANSMISSION.
THE TEST IS PERFORMED AT ALL BAUDRATES (EXCEPT 50 BAUD), 8 BITS PER
CHARACTER, 1 STOP BIT, AND WITH PARITY CHECKING (BOTH ODD AND EVEN).
A HIGH SPEED TEST IS ALSO PERFORMED AT THE HIGHEST 3 BAUDRATES AT
BOTH 5 AND 8 BITS PER CHARACTER, 1 STOP BIT, AND NO PARITY CHECKING.
THIS TEST IS PERFORMED WITH THE TYPE OF LOOPBACK WHICH WAS SPECIFIED
IN THE DUT HARDWARE P TABLE ON ALL ACTIVE LINES.
*****
BGNTEST
T7::
SETPRI @PRI05 ;ALLOW LTC INTERRUPTS.
MOV @PRI05,R0
TRAP C$SPRI
INUM == INUM + 1 ;INCREMENT THE ASSEMBLY TIME TEST COUNTER.
MOV @INUM,ISTNUM ;SET UP THE TEST NUMBER. (91)
MOV @1,CTRLCF ;INDICATE THAT WE ARE IN A TEST.
MOV @1,ERRTYP ;SET ERROR TYPE AS FATAL IN ERROR TABLE.
MOV @9101,ERRNBR ;SET THE FIRST ERROR NUMBER IN ERROR TABLE.
MOV @EM9101,ERRMSG ;SET ERROR MESSAGE ADDRESS IN ERRTBL.
CLR ERSMRF ;INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
CLR ERROR ;CLEAR THE "AT LEAST ONE ERROR" INDICATOR.
;
; RESET THE DUT TO A KNOWN STATE, REMOVE THE STATUS CODES FROM THE FIFO,
; CLEAR TX AND RX INTERRUPT ENABLE BITS IN THE CSR.
; THIS SUBROUTINE REPORTS ERROR >>>> 9101 <<<<.
;
JSR PC,CLRST ;RESET THE DHU 11, REPORT ANY ERRORS FOUND.
BCS CS ;SKIP AROUND TEST EXIT IF NO FATAL ERROR FOUND.
JMP 60$ ;RESET FAILURE, ABORT THIS TEST.
;
; SET UP FOR TRANSMIT INTERRUPTS.
2$: SETPRI @PRI07 ;DISABLE ALL INTERRUPTS.
MOV @PRI07,R0
TRAP C$SPRI
SETVEC TXVECA,@TXDMA,@PRI06 ;SELECT DMA TX INT SERVICE RTN.
MOV @PRI06,(SP)
MOV @TXDMA,(SP)
MOV TXVECA,(SP)
MOV @3,(SP)
TRAP C$VECC
ADD @10,SP
SETVEC RXVECA,@RXCHRS,@PRI06 ;SELECT RX INT SERVICE RTN.
MOV @PRI06,(SP)
MOV @RXCHRS,(SP)
MOV RXVECA,(SP)
MOV @3,(SP)
TRAP C$VECC
ADD @10,SP
SETPRI @PRI04 ;ALLOW INTERRUPTS.
MOV @PRI04,R0
TRAP C$SPRI

```

```

8460
8461
8462
8463
8464
8465 035160 012700 003302
8466 035164 004767 162170
8467 035170 012701 010470
8468 035174 004767 163336
8469 035200 012702 005154
8470 035204 012703 000020
8471 035210 012704 000001
8472 035214 004767 171374
8473 035220 012767 177400 145004
8474 035226 012767 021616 150062
8475
8476
8477
8478 035234 004767 165472
8479 035240 103402
8480 035242 000167 000452
8481
8482 035246 004767 165662
8483 035252 004767 163336
8484 035256 012767 021621 150032
8485
8486
8487
8488 035264 004767 165700
8489 035270 005767 144730
8490 035274 001406
8491 035276 032767 000100 144656
8492 035304 001002
8493 035306 000167 000406
8494
8495 035312 012767 021627 147776 5$:
8496
8497
8498
8499 035320 004767 170722
8500 035324 005767 144674
8501 035330 001406
8502 035332 032767 000100 144622
8503 035340 001567
8504
8505
8506
8507
8508 035342 010100
8509 035344 042701 000100
8510 035350 005100
8511 035352 042700 177677
8512 035356 050001
8513 035360 062701 010400
8514 035364 103303
8515
8516

; TRANSMIT AND RECEIVE SHORT DATA PATTERN AT ALL BAUDRATES,
; WITH 8 BITS PER CHARACTER, 1 STOP BIT, AND BOTH TYPES OF PARITY.
; BOTH LINE GROUPS (GPRS) TX AND RX WITH THE SAME PARAMETERS.
;
; MOV #ERCNTB,R0
; JSR PC,CIR16W ;CLEAR THE RX ERROR COUNTERS TABLE.
; MOV #10470,R1 ;SET UP LPR CONTENTS FOR TX/RX AT 75 BAUD.
4$: ; JSR PC,GETTIM ;GET TIME-OUT BASED ON MINIMUM BAUDRATE IN USE.
; MOV #SDP2B,R2 ;SET UP THE START ADR OF THE DATA PATTERN.
; MOV #SDP2E-SDP2B,R3 ;SET UP THE DATA PATTERN LENGTH.
; MOV #1,R4 ;SPECIFY TO SEND 1 DATA PATTERN TO EACH LINE.
; JSR PC,VANSUP ;SET UP "VANILLA FLAVORED" TX/RX.
; MOV #177400,IBM ;FORM BIT MAP OF UNUSED TX/RX BITS.
; MOV #9102,,ERRNBR ;SET THE ERROR REPORT NUMBER TO 9102.
;
; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9102 THRU 9104 <<<<.
;
; JSR PC,PUFIFR ;PURGE THE OUT RECEIVE CHARACTER FIFO.
; BCS .+6
; JMP 50$ ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
;
; JSR PC,PURRXB ;PURGE THE RX CHAR BUFFER IN MEMORY.
; JSR PC,INIDATA ;SEND THE FIRST BATCH OF DATA PATTERNS.
; MOV #9105,,ERRNBR ;SET ERROR NUMBER TO 9105.
;
; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9105 THRU 9110 <<<<.
;
; JSR PC,RDCHRS ;READ AND VERIFY THE RX CHARACTERS.
; TST FERROR ;HAS AN ERROR BEEN DETECTED?
; BEQ 5$ ;NO, THEN BRANCH.
; BIT #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
; BNE .+6
; JMP 50$ ;NO, THEN EXIT THE TEST.
;
5$: MOV #9111,,ERRNBR ;SET ERROR NUMBER TO 9111.
;
; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9111 THRU 9114 <<<<.
;
; JSR PC,IXRREP ;REPORT FINAL ERRORS FROM RX/RX.
; TST FERROR ;HAS AN ERROR BEEN DETECTED?
; BEQ 6$ ;NO, THEN BRANCH.
; BIT #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED?
; BEQ 50$ ;NO, THEN EXIT THE TEST.
;
; TOGGLE THE PARITY TYPE BIT SPECIFIER IN THE TX/RX SETUP PARAMETERS.
; SELECT THE NEXT BAUDRATE AND PERFORM THE TEST AGAIN IF NOT DONE.
;
6$: MOV R1,R0 ;COMPLEMENT THE PARITY TYPE
; BIC #100,R1 ; BIT IN THE TX/RX LPR SETUP
; COM R0 ; PARAMETER LEAVING THE
; BIC #177677,R0 ; OTHER LPR PARAMETER
; BIT R0,R1 ; BITS UNCHANGED.
; ADD #10400,R1 ;SELECT THE NEXT BAUDRATE.
; HCC 4$ ;LOOP TO TX/RX AGAIN IF NOT PAST LAST BAUDRATE.
;
; PERFORM WIDE OPEN DMA TEST.
    
```

```

8517 ; TRANSMIT AND RECEIVE 512 BYTE DATA PATTERNS AT ALL COMBINATIONS OF 9.6K,
8518 ; 19.2K, AND 38.4K BAUDRATES AND 5 AND 8 BITS PER CHARACTER. USE 1 STOP BIT
8519 ; AND NO PARITY GENERATION OR DETECTION.
8520 ;
8521 ;
8522 ; INITIALIZE THE 512 BYTE PATTERN AND THE VARIOUS DATA PATTERN POINTERS.
8523 ;
8524 035366 005001 ; CLEAR THE DATA BYTE COUNTER.
8525 035370 012702 003602 MOV #0BUFBA5,R2 ; GET THE BASE OF THE DATA PATTERN BUFFER.
8526 035374 110122 7$: MOV R1,(R2)+ ; WRITE A BYTE OF THE DATA PATTERN.
8527 035376 105201 INCB R1 ; GET THE NEXT BYTE FOR THE DATA PATTERN.
8528 035400 001375 BNE 7$ ; LOOP UNTIL FIRST 1/2 OF PATTERN IS DONE.
8529 035402 105301 8$: DECB R1 ; GET THE NEXT BYTE FOR THE DATA PATTERN.
8530 035404 110122 MOV R1,(R2)+ ; WRITE A BYTE OF THE DATA PATTERN.
8531 035406 105701 TSTB R1 ; CHECK FOR DONE WRITING DATA PATTERN.
8532 035410 001374 BNE 8$ ; LOOP IF DATA PATTERN IS NOT DONE.
8533 035412 110122 10$: MOV R1,(R2)+ ; WRITE A BYTE OF THE 32 BYTE OVERFLOW REGION.
8534 035414 005201 INC R1 ; COUNT THIS BYTE.
8535 035416 020127 000040 CMP R1,#32. ; TEST FOR 32 BYTES WRITTEN.
8536 035422 001373 BNE 10$ ; LOOP UNTIL 32 BYTES ARE WRITTEN.
8537 ;
8538 ; PREPARE TO LOOP ON THE 3 DIFFERENT BAUDRATES (9.6K, 19.2K, AND 38.4K).
8539 ;
8540 035424 012705 005072 MOV #DLPRTB,R5 ; GET THE BASE ADR OF THE DMA BAUDRATE TABLE.
8541 ;
8542 ; SPECIFY THE PROPER BAUDRATE.
8543 ; SPECIFY 8 BITS PER CHARACTER.
8544 ; PERFORM DMA TRANSMISSION AND RECEPTION OF 512 BYTE DATA PATTERN.
8545 ;
8546 ;
8547 ; THE FOLLOWING ROUTINE REPORTS THE ERROR WITH NUMBERS 914 THRU 921.
8548 ; LPR CHANGE BIT ERROR FLAGS MAY BE SET BY THIS SUBROUTINE.
8549 ;
8550 035430 012501 12$: MOV (R5)+,R1 ; SET UP LPR PARAM AT NEXT BAUD, 8 BITS/CHAR.
8551 035432 004767 163100 JSR PC,GETTIM ; GET TIME-OUT BASED ON MINIMUM BAUDRATE IN USE.
8552 035436 012702 003602 MOV #0BUFBA5,R2 ; SET UP THE START ADR OF THE DATA PATTERN.
8553 035442 112703 001000 MOV #512,R3 ; SET UP THE DATA PATTERN LENGTH.
8554 035445 12704 000001 MOV #1,R4 ; SPECIFY TO SEND 1 DATA PATTERN TO EACH LINE.
8555 035452 004767 171136 JSR PC,VANSUP ; SET UP "VANILLA FLAG/ORED" TX/RX.
8556 035456 012767 177400 14454r MOV #177400,IBM ; FORM BIT MAP OF UNUSED BITS FOR 8 BITS/CHAR.
8557 035464 012767 021633 147624 MOV #9115,ERRNBR ; SET ERROR NUMBER TO 9115.
8558 ;
8559 ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9115 THRU 9117 <<<<.
8560 ;
8561 035472 004767 165234 JSR PC,PURGER ; PURGE THE OUT RECEIVE CHARACTER FIFO.
8562 035476 103126 BCC 60$ ; ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8563 035500 012767 02163r 147610 MOV #9118,ERRNBR ; SET ERROR NUMBER TO 9118.
8564 ;
8565 035506 004767 16542r JSR PC,PURRXP ; PURGE THE RX CHAR BUFFER IN MEMORY.
8566 035512 004767 163076 JSR PC,INDF ; SEND THE FIRST BATCH OF DATA PATTERNS.
8567 ;
8568 ; THIS ROUTINE REPORTS THE ERROR WITH NUMBERS >>>> 9118 THRU 9123 <<<<.
8569 ;
8570 035516 004767 165446 JSR PC,RDCHRS ; READ AND VERIFY THE RX CHARACTERS.
8571 ;
8572 035522 001767 14447r TEST ERROR ; HAS AN ERROR BEEN DETECTED?
8573 035526 001407 BEQ 14$ ; NO, THEN BRANCH.

```

```

8574 035530 032767 000100 144424      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8575 035535 001470      BEQ    50$           ;NO, THEN EXIT THE TEST.
8576
8577 035540 012767 021644 147550      MOV    #9124.,ERRNBR ;SET ERROR NUMBER TO 9124.
8578
8579      ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9124 THRU 9127 <<<<.
8580
8581 035546 004767 170474 14$:    JSR    PC, TXRREP    ;REPORT FINAL ERRORS FROM RX/RX.
8582 035552 005767 144446      TST    FERROR      ;HAS AN ERROR BEEN DETECTED ?
8583 035556 001404      BEQ    16$         ;NO, THEN BRANCH.
8584 035560 032767 000100 144374      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8585 035566 001454      BEQ    50$         ;NO, THEN EXIT THE TEST.
8586
8587 035570 012767 021650 147520 16$:    MOV    #9128.,ERRNBR ;SET ERROR NUMBER TO 9128.
8588
8589      ; SPECIFY 5 BITS PER CHARACTER.
8590      ; PERFORM DMA TRANSMISSION AND RECEPTION OF 512 BYTE DATA PATTERN.
8591
8592 035576 042701 000030      BIC    #30,R1      ;SET UP CHAR LENGTH PARAM TO 5 BITS/CHAR.
8593 035602 004767 171006      JSR    PC,VANSUP   ;SET UP "VANILLA FLAVORED" TX/RX.
8594 035606 012767 177740 144416      MOV    #177740,IBM ;FORM BIT MAP OF UNUSED BITS FOR 5 BITS/CHAR.
8595
8596      ; THIS ROUTINE REPORTS THE ERROR WITH NUMBERS >>> 9128 THRU 9131 <<<.
8597
8598 035614 004767 165112      JSR    PC,PUFIFR   ;PURGE THE OUT RECEIVE CHARACTER FIFO.
8599 035620 103055      BCC    60$         ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8600 035622 012767 021654 147466      MOV    #9132.,ERRNBR ;SET THE ERROR REPORT NUMBER TO 9132.
8601
8602 035630 004767 165300      JSR    PC,PURRXB   ;PURGE THE RX CHAR BUFFER IN MEMORY.
8603 035634 004767 162754      JSR    PC,INIDMA   ;SEND THE FIRST BATCH OF DATA PATTERNS.
8604
8605      ; THIS ROUTINE REPORTS THE ERROR WITH NUMBERS >>>> 9132 THRU 9137 <<<<.
8606
8607 035640 004767 165324      JSR    PC,RDCHRS   ;READ AND VERIFY THE RX CHARACTERS.
8608 035644 005767 144354      TST    FERROR      ;HAS AN ERROR BEEN DETECTED ?
8609 035650 001404      BEQ    18$         ;NO, THEN BRANCH.
8610 035652 032767 000100 144302      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8611 035660 001417      BEQ    50$         ;NO, THEN EXIT THE TEST.
8612
8613 035662 012767 021662 147426 18$:    MOV    #9138.,ERRNBR ;SET ERROR NUMBER TO 9138.
8614
8615      ; THIS ROUTINE REPORTS THE ERROR WITH NUMBERS >>>> 9138 THRU 9141 <<<<.
8616
8617 035670 004767 170352      JSR    PC, TXRREP    ;REPORT FINAL ERRORS FROM RX/RX.
8618 035674 005767 144324      TST    FERROR      ;HAS AN ERROR BEEN DETECTED ?
8619 035700 001404      BEQ    20$         ;NO, THEN BRANCH.
8620 035702 032767 000100 144252      BIT    #BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8621 035710 001403      BEQ    50$         ;NO, THEN EXIT THE TEST.
8622
8623 035712 020527 005100 20$:    CMP    R5,#DI PRTE ;COMPARE DMA BAUDRATE TABLE PTR WITH TABLE END.
8624 035716 103644      BEQ    12$         ;LOOP IF NOT ALL BAUDRATES DONE YET.
8625
8626      ; ALL DONE, HAVE EITHER RPT OUT OF ACTIVE LINES, OR COMPLETED THE TEST.
8627      ; DISABLE INTERRUPTS.
8628      ; CLEAR THE INTERRUPT VECTORS.
8629
8630 035720 50$:    SETPRI #PRI07    ;DISABLE ALL INTERRUPTS.

```

NL

	035720	012700	000340					MOV	#PRIORITY,RO
	035724	104441						TRAP	C\$SPRI
8631	035726			CLRVEC	TXVECA			;RETURN TX INT VECTOR TO UNUSED POOL.	
	035726	016700	144236					MOV	TXVECA,RO
	035732	104436						TRAP	C\$CVEC
8632	035734			CLRVEC	RXVECA			;RETURN RX INT VECTOR TO UNUSED POOL.	
	035734	016700	144226					MOV	RXVECA,RO
	035740	104436						TRAP	C\$CVEC
8633									
8634	035742	012767	021666	147346	MOV	#9142,ERRNBR		;SELECT NUMBER 9142 FOR THE NEXT ERROR REPORT.	
8635	035750	004767	166054		JSR	PC,REPSMR		;REPORT ERROR SUMMARIES IF CALLED FOR.	
8636	035754			60\$:	SETPRI	#PRIORITY		;DISABLE ALL INTERRUPTS.	
	035754	012700	000340					MOV	#PRIORITY,RO
	035760	104441						TRAP	C\$SPRI
8637	035762	005067	144232	CLR	CTRLCF			;INDICATE THAT WE ARE NOT WITHIN A TEST.	
8638	035766			ENDTST					
	035766							110033:	
	035766	104401						TRAP	C\$SETST

```

8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650 035120
      035120
8651 035120 126127 144200 000002
8652 035126 001402
8653 036000 000167 000524
8654 036004
      036004 012700 000240
      036010 104441
8655
      000010
8656 036012 012767 000010 144240
8657 036020 012767 177777 144172
8658 036026 012767 000001 117260
8659 036034 012767 021781 147254
8660 036042 012767 012542 147250
8661 036050 005067 14414
8662 036054 005067 144144
8663
8664
8665
8666
8667
8668 036060 004767 161252
8669 036064 103402
8670 036066 000167 000436
8671
8672
8673
8674
8675 036072
      036072 012700 000340
      036076 104441
8676 036100
      036100 012746 000300
      036104 012746 027426
      036110 016746 144054
      036114 012746 000003
      036120 104437
      036122 062706 000010
8677 036126
      036126 012746 000300
      036132 012746 027216
      036136 016746 144024
      036142 012746 000003
      036146 104437
      036150 062706 000010
8678 036154
      036154 012700 000200

```

```

SUBTITLE HARDWARE TEST SPLSPD
*****
;
; SPLIT SPEED TEST
; THIS TEST IS USED TO VERIFY THE SPLIT SPEED CAPABILITIES OF THE DHU11,
; AND THE CORRECT OPERATION OF THE A & B BAUD RATE GROUP SELECTION.
; THE TEST USES THREE SETS OF BAUD RATES (38.4,50; 1200,75; 2000,2400).
; THIS TEST WILL ONLY EXECUTE IF THE STAGGERED LOOPBACK MODE IS SELECTED.
; THE SPECIAL STAGGERED LOOPBACK BERG CONNECTOR MUST BE FITTED.
*****
RGNTST
;
; T8::
; CHECK MODE SELECTED.
; DO NOT EXIT IF STAGGERED LOOPBACK MODE SELECTED.
; EXIT THIS TEST.
; ALLOW LTC INTERRUPTS.
;
; INCREMENT THE ASSEMBLY TIME TEST COUNTER.
; SET UP THE TEST NUMBER. (92)
; INDICATE THAT WE ARE IN A TEST.
; SET ERROR TYPE IN ERROR TABLE.
; SET THE FIRST ERROR NUMBER IN ERROR TABLE.
; SET ERROR MESSAGE ADDRESS IN ERROR TABLE.
; INITIALIZE THE "REPORT ERROR SUMMARY" FLAGS.
; CLEAR THE "AT LEAST ONE ERROR" FLAG.
;
; RESET THE DUT TO A KNOWN STATE, REMOVE STATUS CODES FROM THE FIFO.
; CLEAR TX AND RX INTERRUPT ENABLE BITS.
; THIS SUBROUTINE REPORTS ERROR >>>> 9201 <<<<<.
;
; RESET THE DUT.
; ABORT THE TEST IF FATAL ERROR FOUND IN RESET.
;
; DISABLE ALL INTERRUPTS.
; SET UP DMA TX AND RX INTERRUPT SERVICE ROUTINES.
;
; DISABLE ALL INTERRUPTS.
; SELECT DMA TX INT SERVICE RTN.
; SELECT RX INT SERVICE RTN.
; ALLOW INTERRUPTS.
;
MOV @PRI05,RO
TRAP C$SPRI
MOV @PRI06,C$SPRI
MOV @TXDMA,C$SPRI
MOV TXVECA,C$SPRI
MOV @3,C$SPRI
TRAP C$SVEC
ADD @10,SP
MOV @PRI06,C$SPRI
MOV @RXCHRS,C$SPRI
MOV RXVECA,C$SPRI
MOV @3,C$SPRI
TRAP C$SVEC
ADD @10,SP
MOV @PRI04,RO

```

```

036160 104441 TRAP C$SPRI
8679
8680
8681
8682 036162 012705 177777
8683 036166 004767 167236
8684
8685
8686
8687
8688 036172 012700 005502
8689 036176 004767 161156
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701 036202 012705 005100
8702 036206 012500
8703 036210 012501
8704 036212 004767 162320
8705 036216 012702 005154
8706 036222 012503
8707 036224 012504
8708 036226 012767 177400 143776
8709 036234 004767 166444
8710 036240 012767 021762 147050
8711
8712
8713
8714 036246 004767 164460
8715 036252 103126
8716 036254 012767 021765 147034
8717
8718 036262 004767 164646
8719 036266 004767 162322
8720
8721
8722
8723 036272 004767 164672
8724 036276 005767 143722
8725 036302 001404
8726 036304 032767 000100 143650
8727 036312 001473
8728
8729 036314 012767 021773 146774 6$:
8730
8731
8732
8733 036322 004767 167722
8734 036326 005767 143672

```

```

; ENABLE TRANSMITTERS ON ALL LINES.
MOV 0MAPINS,R5 ;PASS ACTIVE LINE BIT MAP.
JSR PC, TXENBL ;ENABLE TRANSMISSIONS ON ALL LINES.

; CLEAR ERROR TABLE PRIOR TO PERFORMING TX/RX TEST.
MOV 0ERCNTB,R0 ;GET THE BASE ADDRESS OF THE ERROR COUNTER TBL.
JSR PC, CLR16W ;CLEAR THE RX ERROR COUNTERS TABLE.

; PERFORM SPLIT SPEED DMA TX AND RX ON ALL SELECTED LINES AT THE FOLLOWING
; BAUD RATES.
; 38.4K, 50 ; 1200, 75 ; 2000, 2400.

; INITIALISE DMA TX/RX PARAMETERS IN THE CONTROL BLOCK FR EACH OF THE BAUD
; RATES MENTIONED ABOVE.
; 8 BITS/CHAR, 1 STOP BITS, ODD PARITY.
MOV 0SPLPRB,R5 ;GET BASE ADDRESS OF LPR PARAMETER TABLE.
4$: MOV (R5)+,R0 ;GET LPR CONTENTS FOR LINGRP II.
MOV (R5)+,R1 ;GET LPR CONTENTS FOR LINGRP I.
JSR PC, GETTIM ;GET TIME-OUT BASED ON MINIMUM BAUDRATE IN USE.
MOV 0SDP2B,R2 ;SET UP THE START ADR OF THE DATA PATTERN.
MOV (R5)+,R3 ;GET NUMBER OF REPEAT TRANSMISSION ON LINGRP II.
MOV (R5)+,R4 ;GET NUMBER OF REPEAT TRANSMISSION ON LINGRP I.
MOV 0177400,IAM ;SET THE UNUSED BIT MASK FOR 8 BITS/CHAR.
JSR PC, SPLSWP ;SET UP CONTROL BLOCK ETC. FOR TX/RX.
MOV 09202,,ERRNBR ;SET THE ERROR NUMBER TO 9202.

; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9202 THRU 9204 <<<<.
JSR PC, PURIFR ;PURGE THE OUT RECEIVE CHARACTER FIFO.
BCC 60$ ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
MOV 09205,,ERRNBR ;SET ERROR NUMBER TO 9205.

JSR PC, PURRXB ;PURGE THE RX CHAR BUFFER IN MEMORY.
JSR PC, INIDMA ;SEND THE FIRST BATCH OF DATA PATTERNS.

; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9205 THRU 9210 <<<<.
JSR PC, RDCRYS ;READ AND VERIFY THE RX CHARACTERS.
TST ERROR ;HAS AN ERROR BEEN DETECTED ?
BEQ 6$ ;NO, THEN BRANCH.
BIT 0BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
BEQ 50$ ;NO, THEN EXIT THE TEST.

MOV 09211,,ERRNBR ;SET THE ERROR NUMBER TO 9211.

; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9211 THRU 9214 <<<<.
JSR PC, TXRREP ;REPORT FINAL ERRORS FROM RX/RX.
TST ERROR ;HAS AN ERROR BEEN DETECTED ?

```

D2

```

8735 036332 001404          BEQ      8$          ;NO, THEN BRANCH.
8736 036334 032767 000100 143620      BIT      0BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8737 036342 001457          BEQ      50$          ;NO, THEN EXIT THE TEST.
8738
8739 036344 012767 021777 146744 8$:      MOV      09215.,ERRNBR ;SET ERROR NUMBER TO 9215.
8740
8741          ; SWAP PARAMETERS TO ALLOW FOR BOTH CHANNELS TO BE EXERCISED.
8742
8743 036352 010246          MOV      R2,(SP)      ;PUSH THE START ADDRESS ONTO THE STACK.
8744 036354 010002          MOV      R2,R2
8745 036356 010100          MOV      R1,R0
8746 036360 010201          MOV      R2,R1      ;SWAP THE TWO SETS OF
8747 036362 010302          MOV      R3,R2      ;PARAMETERS OVER.
8748 036364 010403          MOV      R4,R3
8749 036366 010204          MOV      R2,R4
8750 036370 012602          MOV      (SP),R2
8751 036372 004767 166306      JSR      PC,SPL SUP  ;RESTORE THE START ADDRESS.
8752
8753          ;
8754          ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9215 THRU 9217 <<<<.
8755
8756 036376 004767 164330      JSR      PC,PUFIFR   ;PURGE THE DUT RECEIVE CHARACTER FIFO.
8757 036402 103052          BCC      60$          ;ABORT THIS TEST IF FIFO WOULD NOT PURGE.
8758 036404 012767 022002 146704      MOV      09218.,ERRNBR ;SET ERROR NUMBER TO 9218.
8759
8760 036412 004767 164516      JSR      PC,PURRXB   ;PURGE THE RX CHAR BUFFER IN MEMORY.
8761 036416 004767 162172      JSR      PC,INIDMA   ;SEND THE FIRST BATCH OF DATA PATTERNS.
8762
8763          ;
8764          ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9218 THRU 9223 <<<<.
8765
8766 036422 004767 164542      JSR      PC,RDCHRS   ;READ AND VERIFY THE RX CHARACTERS.
8767 036426 005767 143572      TST      FERROR      ;HAS AN ERROR BEEN DETECTED ?
8768 036432 001404          BEQ      10$          ;NO, THEN BRANCH.
8769 036434 032767 000100 143520      BIT      0BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8770
8771 036444 012767 022010 146644 10$:      MOV      09224.,ERRNBR ;SET ERROR NUMBER TO 9224.
8772
8773          ;
8774          ; THIS ROUTINE REPORTS ERRORS WITH NUMBERS >>>> 9224 THRU 9227 <<<<.
8775
8776 036452 004767 167570      JSR      PC,IXRREP   ;REPORT FINAL ERRORS FROM RX/RX.
8777 036456 005767 143542      TST      FERROR      ;HAS AN ERROR BEEN DETECTED ?
8778 036462 001404          BEQ      10$          ;NO, THEN BRANCH.
8779 036464 032767 000100 143470      BIT      0BIT06,OPTION ;HAS EXTENDED ERROR REPORTING BEEN REQUESTED ?
8780
8781 036474 020527 005130 10$:      CMP      R5,0$SPL PRF ;CHECK IF ALL PARAMETERS HAVE BEEN DONE.
8782 036500 103642          BLO      4$          ;IF NOT DONE LOOP TO SELECT THE NEXT PARAMETER.
8783
8784          ;
8785          ; DISABLE INTERRUPTS.
8786          ; CLEAR THE INTERRUPT VECTORS.
8787 036502 50$:      SETPRI 0PRI07      ;DISABLE ALL INTERRUPTS.
8788 036502 012700 000340          MOV      0$R0,R0
8789 036506 104441          TRAP    0$PRI
8790 036510 016700 143454          CLRVEC TXVECA      ;RETURN TX INT VECTOR TO UNUSED
8791 036510 016700 143454          MOV      TXVECA,R0

```


12

8789	036514	104436						TRAP	C\$CVEC
8790	036516	012767	022014	146572		MOV	09228,ERRNBR	;SELECT NUMBER 9228 FOR THE NEXT ERROR REPORT.	
8791	036524	004767	165300			JSR	PC,REP\$MR	;REPORT ERROR SUMMARIES IF CALLED FOR.	
8792	036530	005067	143464		60\$:	CLR	CTRLCF	;INDICATE THAT WE ARE NOT WITHIN A TEST.	
8793	036534					ENDTST			
	036534							L10034:	
	036534	104401						TRAP	C\$ETST

```

8795
8796
8797
8798
8799
8800
8801
8802
8803
8804
8805 036536
      036536
8806      000011
8807 036536 012767 000011 143514
8808 036544 012767 177777 143446
8809 036552 016702 143732
8810 036556 012703 002512
8811 036562 020203
8812 036564 001411
8813
8814
8815
8816
8817
8818 036566 012701 012746
8819 036572 104455
      036574 022125
      036576 012572
      036600 015626
8820
8821 036602 012767 002512 143700
8822
8823 036610 005067 143404 60$:
8824 036614
      036614
      036614 104401

```

```

.SBTL. HARDWARE TEST REP8MP
;*****
; - REPORT ANY BMP CODES IN THE QUEUE -
; THIS IS A PSEUDO-TEST USED TO REPORT ANY BMP CODES THAT WERE FOUND
; IN THE DUT'S FIFO DURING PREVIOUS TEST, AND LOGGED IN THE BMP CODE
; QUEUE.
; IT IS UNLIKELY THAT RUNNING THIS PSEUDO-TEST ALONE WILL PRODUCE ANY
; ERROR REPORTS.
;*****
      BGNTST
;*****
; T9:
      TNUM = TNUM + 1 ; INCREMENT THE ASSEMBLY TIME TEST COUNTER.
      MOV  #TNUM,TSTNUM ; SET UP THE TEST NUMBER. (93)
      MOV  #1,CTRLCF ; INDICATE THAT WE ARE IN A TEST.
      MOV  #BMPQBP,R2 ; GET THE CONTENTS OF THE POINTER.
      MOV  #BMPQBP,R3 ; GET THE START ADDRESS OF THE QUEUE.
      CMP  R2,R3 ; SEE IF THE POINTER HAS MOVED FROM THE BASE.
      BEQ  60$ ; EXIT NO CODES IN THE QUEUE.
;*****
; THERE IS AT LEAST ONE BMP CODE IN THE QUEUE. REPORT THE ERROR.
;*****
; REPORT ERROR BMP CODE FOUND IN TEST NN, BMP CODE:NNNNNN"
      MOV  #EM9304,R1 ; PASS THE FIRST MESSAGE TO BE REPORTED.
      ERDF 9301,EM9301,ER9301 ; >>>> ERROR #9301 <<<<<.
;*****
; TRAP C$ERDF
; .WORD 9301
; .WORD EM9301
; .WORD ER9301
;*****
      MOV  #BMPQBP,BMPQBP ; SET POINTER BACK TO THE BEGINING OF THE QUE.
;*****
; INDICATE THAT WE ARE NOT WITHIN A TEST.
      CLR  CTRLCF
      ENDTST
;*****
; 10035:
; TRAP C$ETST

```

8833
8834
8835
8836
8837
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847

.SBTTL. 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.
;--
    
```

8848 036616
036616 000022
036620

BGNHRD

.WORD L10036-L\$HARD/2
L\$HARD: :

8849
8859
8860 036620
036620 000031
036622 036664
036624 160000
036626 177776

;DEVICE CSR ADDRESS QUESTION:
GPRMA HWPTQ1,0,0,160000,177776,YES

.WORD T\$CODE
.WORD HWPTQ1
.WORD T\$LOLIM
.WORD T\$HILIM

8861
8862 036630
036630 001031
036632 036702
036634 000040
036636 000776

;DEVICE INTERRUPT VECTOR QUESTION:
GPRMA HWPTQ2,2,0,40,776,YES

.WORD T\$CODE
.WORD HWPTQ2
.WORD T\$LOLIM
.WORD T\$HILIM

8863
8864 036640
036640 002032
036642 036735
036644 177777
036646 000000
036650 177777

;ACTIVE LINES BIT MAP QUESTION:
GPRMD HWPTQ3,4,0,MAPLNS,0,MAPLNS,YES

.WORD T\$CODE
.WORD HWPTQ3
.WORD MAPLNS
.WORD T\$LOLIM
.WORD T\$HILIM

8865
8866 036652
036652 003032
036654 036763
036656 000377
036660 000001
036662 000005

;TYPE OF LOOPBACK QUESTION:
GPRMD HWPTQ4,6,0,377,1,5,YES

.WORD T\$CODE
.WORD HWPTQ4
.WORD 377
.WORD T\$LOLIM
.WORD T\$HILIM

8867
8868
8869 036664

ENDHRD

L10036: .EVEN

8870
8871
8878 036664 103 123 127
036667 040 101 104
036672 104 122 105
036675 123 122 072
036700 040 000

HWPTQ1: .ASCII /CSR ADDRESS: /

8879	036702	111	116	124	HWPTQ2: .ASCIZ /INTERRUPT VECTOR ADDRESS: /
	036705	105	122	122	
	036710	125	120	124	
	036713	040	126	105	
	036716	103	124	117	
	036721	122	040	101	
	036724	104	104	122	
	036727	105	123	123	
	036732	072	040	000	
8880	036735	101	103	124	HWPTQ3: .ASCIZ /ACTIVE LINE BIT MAP: /
	036740	111	126	105	
	036743	040	114	111	
	036746	116	105	040	
	036751	102	111	124	
	036754	040	115	101	
	036757	120	072	040	
	036762	000			
8881	036763	124	131	120	HWPTQ4: .ASCII /TYPE OF LOOPBACK (1=INTERNAL, 2=H3029 OR H3277, 3=H325/<15><12>
	036766	105	040	117	
	036771	106	040	114	
	036774	117	117	120	
	036777	102	101	103	
	037002	113	040	050	
	037005	061	075	111	
	037010	116	124	105	
	037013	122	116	101	
	037016	114	054	040	
	037021	062	075	110	
	037024	063	060	062	
	037027	071	040	117	
	037032	122	040	110	
	037035	063	062	067	
	037040	067	054	040	
	037043	063	075	110	
	037046	063	062	065	
	037051	015	012		
8882	037053	040	040	040	.ASCIZ /
	037056	040	040	040	4=MODEM, 5=KEYBOARD ECHO); /
	037061	040	040	040	
	037064	040	040	040	
	037067	040	040	040	
	037072	040	040	040	
	037075	040	040	064	
	037100	075	115	117	
	037103	104	105	115	
	037106	054	040	065	
	037111	075	115	105	
	037114	131	102	117	
	037117	101	122	104	
	037122	040	105	103	
	037125	110	117	051	
	037130	072	040	000	

8884

.EVEN

8893
8894
8895
8896
8897
8898
8899
8900
8901
8902
8903
8904
8905
8906
8907
8916
8917
8918
8919
8920
8921
8922
8923
8924
8925
8926
8927
8928
8929
8930
8931
8932
8939

037134
037134 000017
037136
037136 000130
037140 037174
037142 000020
037144
037144 000130
037146 037250
037150 000040
037152
037152 000130
037154 037330
037156 000100
037160
037160 006044
037162
037162 001052
037164 037363
037166 177777
037170 000000
037172 177777
037174
037174
037177
037202 040
037205 111
037210 116
037213 102

105 120
122 124
125 116
124 640
125 116
105 122

.SBTTL SOFTWARE PARAMETER CODING SECTION

; 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.

BGNSFT

.WORD L10037-L\$SOFT/2
L\$SOFT::

;UNIT NUMBER PRINTOUT QUESTION:
GPRML SWPTQ1,0,20,YES

.WORD T\$CODE
.WORD SWPTQ1
.WORD 20

;REPORT NUMB OF BITS TESTED IN DMA ADDR TEST QUESTION:
GPRML SWPTQ2,0,40,YES

.WORD T\$CODE
.WORD SWPTQ2
.WORD 40

;EXTENDED ERROR REPORTING QUESTION:
GPRML SWPTQ3,0,100,YES

.WORD T\$CODE
.WORD SWPTQ3
.WORD 100

; IF EXTENDED ERROR REPORTING IS NOT REQUIRED THEN SKIP THE NEXT QUESTION.

XFERF 2\$

.WORD T\$CODE

;NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE QUESTION:
GPRMD SWPTQ4,2,D,177777,0,177777,YES

.WORD T\$CODE
.WORD SWPTQ4
.WORD 177777
.WORD T\$LOU IM
.WORD T\$HIL IM

2\$; ENDSFT

.EVEN
L10037;

SWPTQ1: .ASCIZ /REPORT UNIT NUMBER AS EACH UNIT IS TESTED;

	037216	040	101	123
	037221	040	105	101
	037224	103	110	040
	037227	125	116	111
	037232	124	040	111
	037235	123	040	124
	037240	105	123	124
	037243	105	104	072
	037246	040	000	
8940	037250	122	105	120
	037253	117	122	124
	037256	040	116	125
	037261	115	102	105
	037264	122	040	117
	037267	106	040	102
	037272	111	124	123
	037275	040	124	105
	037300	123	124	105
	037303	104	040	111
	037306	116	040	104
	037311	115	101	040
	037314	101	104	104
	037317	122	040	124
	037322	105	123	124
	037325	072	040	000
8941	037330	105	130	124
	037333	105	116	104
	037336	105	104	040
	037341	105	122	122
	037344	117	122	040
	037347	122	105	120
	037352	117	122	124
	037355	111	116	107
	037360	072	040	000
8942	037363	116	125	115
	037366	102	105	122
	037371	040	117	106
	037374	040	111	116
	037377	104	111	126
	037402	111	104	125
	037405	101	114	040
	037410	104	101	124
	037413	101	040	105
	037416	122	122	117
	037421	122	123	040
	037424	124	117	040
	037427	122	105	120
	037432	117	122	124
	037435	040	117	116
	037440	040	101	040
	037443	114	111	116
	037446	105	072	040
	037451	000		

SWPTQ2: .ASCIZ /REPORT NUMBER OF BITS TESTED IN DMA ADDR TEST: /

SWPTQ3: .ASCIZ /EXTENDED ERROR REPORTING: /

SWPTQ4: .ASCIZ /NUMBER OF INDIVIDUAL DATA ERRORS TO REPORT ON A LINE: /

8943
8944

.EVEN

```
8953
8954
8955 037452          $PATCH::
8956 037452          .BLKW  24
8957
8964
8965
8966
8967
8968 037522          LASTAD
                                .EVEN
                                .WORD  0
                                .WORD  0
                                037522 000000
                                037524 000000
                                L$LAST::
8969 037526          ENDMOD
8970
8971
8972
8973
8974
8975
8976
8977          000001          .END
```

ACTINS	002172	G	CHCNTB	003442	G	C\$OPEN	000034	EF9013	007141	G	ENDIT	030400	
ADDR	025220		CHKEXT	016316	G	C\$PNTB	000014	EF9019	007206	G	ERCNTB	003302	G
ADR	000020	G	CHKLOS	016416	G	C\$PNTF	000017	EF9020	007225	G	ERLTBL	003602	G
ADRPTR	017270	G	CHRTOT	002476	G	C\$PNTS	000016	EF9101	007306	G	ERRBLK	005322	G
ALFLD	016020	G	CKCHR	016520	G	C\$PNTX	000015	EF9103	007311	G	ERRMSG	005320	G
ASSEMB	000010		CKFRPR	016736	G	C\$QIO	000377	EF9301	007357	G	ERRNBR	005316	G
BCOUNT	002306	G	CKINAC	017150	G	C\$RDBU	000007	EF9302	007435	G	ERRTYP	005314	G
BDRMSG	010020	G	CKTRAP	017256	G	C\$REFG	000047	EMLMSG	013074	G	ERSMRF	002500	G
BITSTD	033676		CKTRPB	017306	G	C\$RESE	000033	EM0101	022070	G	ER0101	013500	G
BITTBL	002364	G	CLKBRL	002272	G	C\$REVI	000003	EM0102	022154	G	ER0503	014032	G
BITO	000001	G	CLKCSR	002270	G	C\$RFLA	000021	EM0103	010047	G	ER1603	014070	G
BIT00	000001	G	CLKHRZ	002276	G	C\$RPT	000025	EM0509	010105	G	ER6201	014162	G
BIT01	000002	G	CLKINT	027146	G	C\$SEFG	000046	EM1601	010111	G	ER9001	014420	G
BIT02	000004	G	CLKVEC	002274	G	C\$SPRI	000041	EM4401	010174	G	ER9002	014520	G
BIT03	000010	G	CLNKST	017336	G	C\$SVEC	000037	EM4402	010224	G	ER9003	014676	G
BIT04	000020	G	CLR16W	017360	G	C\$TPRI	000013	EM4403	010272	G	ER9004	015070	G
BIT05	000040	G	CONMAP	017402	G	DELAY	017456	EM4404	010367	G	ER9005	015204	G
BIT06	000100	G	CSRA	002200	G	DFPTBL	002150	EM4405	010426	G	ER9101	015444	G
BIT07	000200	G	CSRO	000000	G	DIAGMC	000000	EM4406	010522	G	ER9102	015504	G
BIT08	000400	G	CTRLCF	002220	G	DLPRTB	005072	EM4407	010576	G	ER9301	015626	G
BIT09	001000	G	C\$AU	000052		DLP RTE	005100	EM4408	010660	G	EVL	000004	G
BIT1	000002	G	C\$AUTO	000061		DMRW	017570	EM4409	010723	G	EXCNTB	003242	G
BIT10	002000	G	C\$BRK	000022		DMTSTA	002222	EM4410	010760	G	EXTMSG	013143	G
BIT11	004000	G	C\$BSEG	000004		DM168	017516	EM4411	011007	G	E\$END	002100	
BIT12	010000	G	C\$BSUB	000002		DODMA	017712	EM6201	011054	G	E\$LOAD	000035	
BIT13	020000	G	C\$CEFG	000045		DPENDB	003142	EM6202	011106	G	FDATA	002206	G
BIT14	040000	G	C\$CLCK	000062		DPLENB	003202	EM6301	011115	G	FDATE	000006	G
BIT15	100000	G	C\$CLEA	000012		DPRSQB	004642	EM8901	011146	G	FERROR	002224	G
BIT2	000004	G	C\$CLOS	000035		DRSQE	005042	EM9003	011173	G	FFREM	002226	G
BIT3	000010	G	C\$CLP1	000006		DRADRT	002200	EM9004	011215	G	FINACT	020002	G
BIT4	000020	G	C\$CVEC	000036		DROP	030454	EM9006	011233	G	FRPSUP	020062	G
BIT5	000040	G	C\$DCLN	000044		DUMY	033700	EM9007	011306	G	F\$LSA	002206	G
BIT6	000100	G	C\$DODU	000051		EDPFMT	005422	EM9008	011371	G	F\$LSO	000006	G
BIT7	000200	G	C\$DRPT	000024		EDROP	030532	EM9009	011452	G	F\$AU	000015	
BIT8	000400	G	C\$DU	000053		EF.CON	000036	EM9010	011476	G	F\$AUTO	000020	
BIT9	001000	G	C\$EDIT	000003		EF.NEW	000035	EM9011	011522	G	F\$BGN	000040	
BMPCEB	002512	G	C\$ERDF	000055		EF.PWR	000034	EM9012	011532	G	F\$CLEA	000007	
BMPCE	002712	G	C\$ERHR	000056		EF.RES	000037	EM9013	011542	G	F\$DU	000016	
BMPCEP	002510	G	C\$ERRO	000060		EF.STA	000040	EM9014	011551	G	F\$END	000041	
BOE	000400	G	C\$ERSF	000054		EF0003	005530	EM9015	011645	G	F\$HARD	000004	
BREVL	002175	G	C\$ERSO	000057		EF1601	005535	EM9016	011661	G	F\$HW	000013	
BRTBLB	002424	G	C\$ESCA	000010		EF1603	005567	EM9017	011670	G	F\$INIT	000006	
BRTBLE	002464	G	C\$ESEG	000005		EF4401	005631	EM9025	012001	G	F\$JMP	000050	
BUFBAS	003602	G	C\$ESUB	000003		EF6201	005746	EM9026	012075	G	F\$MOD	000000	
BUFEND	004602	G	C\$ETST	000001		EF6202	006061	EM9027	012121	G	F\$MSG	000011	
BUFMID	004202	G	C\$EXIT	000032		EF6203	006157	EM9028	012201	G	F\$PROT	000021	
BUF3QT	004402	G	C\$GETB	000026		EF9001	006254	EM9030	012260	G	F\$PWR	000017	
CALMSI	016072	G	C\$GETW	000027		EF9002	006336	EM9101	012335	G	F\$RPT	000012	
CHB	003122	G	C\$GMAN	000043		EF9003	006410	EM9102	012377	G	F\$SEG	000003	
CBDPAA	003126	G	C\$GPHR	000042		EF9004	006437	EM9104	012466	G	F\$SOFT	000005	
CBDPLA	003130	G	C\$GPL0	000030		EF9005	006467	EM9201	012542	G	F\$SRV	000010	
CBDPNA	003132	G	C\$GPRI	000040		EF9006	006520	EM9301	012572	G	F\$SUB	000002	
CHINCA	003124	G	C\$INIT	000011		EF9007	006544	EM9302	012651	G	F\$SW	000014	
CHLPBA	003136	G	C\$INLP	000020		EF9008	006640	EM9303	012701	G	F\$TEST	000001	
CHLPRA	003122	G	C\$MANI	000050		EF9009	006677	EM9304	012746	G	GETBDR	020326	G
CBMAPA	003134	G	C\$MEM	000031		EF9010	006736	EM9401	013021	G	GETCHR	020454	G
CBUFSA	003140	G	C\$MSG	000023		EF9012	007025	ENDETB	004602	G	GETPRM	030176	

GETTIM	020536	G	LINBIT	020706	G	L\$SPTP	002024	G	NEWCHR	021514	G	PUFIFO	022650	G
GMANWD	002230	G	LNCTRA	002210	G	L\$STA	002030	G	NEWPAS	030156		PUFIFR	022732	G
GPRSOB	002464	G	LNCTRC	000010	G	L\$SW	002162	G	NEWRES	030150		PURRXB	023134	G
G\$CNT0	= 000200		LOE	= 040000	G	L\$TEST	002114	G	NEWSTA	027640		RBUFA	002202	G
G\$DELM	= 000372		LOPBCK	002174	G	L\$TIML	002014	G	NUMLNS	= 000020	G	RBUFO	= 000002	G
G\$DISP	= 000003		LOT	= 000010	G	L\$UNIT	002012	G	ODTSTA	033702		RDCHRS	023170	G
G\$EXCP	= 000400		LPCSLT	= 000036	G	L10000	002160		OOPS	022024	G	RDMAST	023630	G
G\$HILI	= 000002		LPRA	002204	G	L10001	002166		OPTION	002162	G	REPCOD	023670	G
G\$LOLI	= 000001		LPRO	= 000004	G	L10002	013614		O\$APTS	= 000000		REPSMR	024030	G
G\$NO	= 000000		L\$ACP	002110	G	L10003	014066		O\$AU	= 000000		RESETT	024056	G
G\$OFFS	= 000400		L\$APT	002036	G	L10004	014160		O\$BGNR	= 000001		RRXNDN	024170	G
G\$OF SI	= 000376		L\$AU	030540	G	L10005	014416		O\$BGNS	= 000001		RTXNDN	024236	G
G\$PRMA	= 000001		L\$AUT	002070	G	L10006	014516		O\$DU	= 000001		RXBCNT	002716	G
G\$PRMD	= 000002		L\$AUTO	030410	G	L10007	014674		O\$ERRT	= 000001		RXBCTX	= 000030	G
G\$PRML	= 000000		L\$CCP	002106	G	L10010	015066		O\$GNSW	= 000001		RXBEND	003120	G
G\$RADA	= 000140		L\$CLEA	030412	G	L10011	015202		O\$POIN	= 000001		RXBETX	= 000020	G
G\$RAD3	= 000000		L\$CO	002032	G	L10012	015442		O\$SETU	= 000000		RXBFUL	= 000100	G
G\$RAD4	= 000040		L\$DEPO	002011	G	L10013	015502		PARATB	002324	G	RXBIPT	002714	G
G\$RADL	= 000120		L\$DESC	005374	G	L10014	015624		PARATE	002344	G	RXBOPT	002712	G
G\$RADO	= 000020		L\$DESP	002076	G	L10015	016016		PAROA	002324	G	RXBSTA	002720	G
G\$XFER	= 000004		L\$DEVP	002060	G	L10016	027564		PAR1A	002326	G	RXCHRS	027216	G
G\$YES	= 000010		L\$DISP	002124	G	L10020	030406		PAR2A	002330	G	RXCNTB	003542	G
HELP	= 000000		L\$DLY	002116	G	L10021	030410		PAR3A	002332	G	RXDNF	002504	G
HOE	= 100000	G	L\$DTP	002040	G	L10022	030426		PAR4A	002334	G	RXDSBL	024304	G
HWPTQ1	036664		L\$DTYP	002034	G	L10023	030536		PAR5A	002336	G	RXENBL	024400	G
HWPTQ2	036702		L\$DU	030430	G	L10024	030544		PAR6A	002340	G	RXIE0	024474	G
HWPTQ3	036735		L\$DUT	002072	G	L10025	031026		PAR7A	002342	G	RXIE1	024534	G
HWPTQ4	036763		L\$DVTY	005364	G	L10026	031270		PASCNT	002242	G	RXPTRB	003402	G
IBE	= 010000	G	L\$DEF	002052	G	L10027	032222		PCSL0T	= 000016	G	RXTIMO	= 000002	G
IBM	002232	G	L\$ENVI	002044	G	L10030	033706		PDRATB	002344	G	RXTMA	002202	G
IDU	= 000040	G	L\$ERRT	005314	G	L10031	034322		PDRATE	002364	G	RXTOUT	002246	G
IER	= 020000	G	L\$ETP	002102	G	L10032	035000		PIROA	002344	G	RXVECA	002166	G
IESTAT	002234	G	L\$EXP1	002046	G	L10033	035766		PIR1A	002346	G	R0SLOT	= 000002	G
INIDMA	020614	G	L\$EXP4	002064	G	L10034	036534		PDR2A	002350	G	R1SLOT	= 000004	G
ISR	= 000100	G	L\$EXP5	002066	G	L10035	036614		PDR3A	002352	G	R2SLOT	= 000006	G
IXE	= 004000	G	L\$HARD	036620	G	L10036	036674		PDR4A	002354	G	R3SLOT	= 000010	G
I\$AU	= 000041		L\$HIME	002120	G	L10037	037174		PDR5A	002356	G	R4SLOT	= 000012	G
I\$AUTO	= 000041		L\$HPCP	002016	G	MAPLNS	= 177777	G	PDR6A	002360	G	R5SLOT	= 000014	G
I\$CLN	= 000041		L\$HPTP	002022	G	MFUNIT	007535	G	PDR7A	002362	G	SAVBMP	024560	G
I\$DU	= 000041		L\$HW	002150	G	MMENAB	002322	G	PMSFLG	002244	G	SAVPRI	002250	G
I\$HRD	= 000041		L\$ICP	002104	G	MMPRES	002320	G	PMSMSG	013352	G	SAVTEN	002252	G
I\$INIT	= 000041		L\$INIT	027574	G	MMSR0	002314	G	PNT	= 001000	G	SCBCIB	005042	G
I\$MOD	= 000041		L\$LAOP	002026	G	MMSR3	002316	G	PREGRT	005346	G	SCBCIE	005052	G
I\$MSG	= 000041		L\$LAST	037526	G	MODSUP	020734	G	PREG05	005324		SCBRTB	005052	G
I\$PROT	= 000040		L\$LOAD	002100	G	MSFMT1	007566	G	PRFRME	022252	G	SCBRTF	005060	G
I\$PTAB	= 000041		L\$LUN	002074	G	MSFMT2	007626	G	PRI	= 002000	G	SCNSTB	005060	G
I\$PWR	= 000041		L\$MREV	002050	G	MSG1	013616	G	PRI00	= 000000	G	SCNSTE	005064	G
I\$RPT	= 000041		L\$NAME	002000	G	MSG2	013674	G	PRI01	= 000040	G	SCTPTB	005064	G
I\$SEG	= 000041		L\$PRIO	002042	G	MSG3	013753	G	PRI02	= 000100	G	SCTPTE	005072	G
I\$SETU	= 000041		L\$PROT	027566	G	MSLCNT	002312	G	PRI03	= 000140	G	SDPBAS	005130	G
I\$SFT	= 000041		L\$PRT	002112	G	MSLGCT	021074	G	PRI04	= 000200	G	SDPEND	005150	G
I\$SRV	= 000041		L\$REPP	002062	G	MSLOOP	021210	G	PRI05	= 000240	G	SDP2B	005154	G
I\$SUB	= 000041		L\$REV	002010	G	MSSRPT	021224	G	PRI06	= 000300	G	SDP2E	005174	G
I\$TST	= 000041		L\$RPT	027560	G	MSTICK	002310	G	PRI07	= 000340	G	SEPTBL	002162	G
I\$IMP	= 000167		L\$SOFT	037136	G	MUL16U	021440	G	PROTBL	005214	G	SKPST	024626	G
LGRP1M	002236	G	L\$SPC	002056	G	NDERPT	002164	G	PRPARE	022356	G	SPLPRB	005100	G
LGRP2M	002240	G	L\$SPCP	002020	G	NUPMSG	013224	G	PRTLPR	022566	G	SPLPRE	005130	G

SPI SUP	024704	G	TRPAD2	017320	G	TXRON	026220	G	T\$SUBN=	000000	T3	031272	G
STGTRB	005274	G	TSTNUM	002260	G	TXRREP	026246	G	T\$TAGL=	177777	T4	032224	G
STPSW	025204	G	TXAD1A	002212	G	TXRXLB	005234	G	T\$TAGN=	010040	T5	033710	G
SUCSS	033704		TXAD10=	000012	G	TXRXLE	005274	G	T\$TEMP=	000000	T6	034324	G
SVCGBL=	000000		TXAD2A	002214	G	TXVECA	002170	G	T\$TEST=	000011	T7	035002	G
SVCINS=	000001		TXAD20=	000014	G	T\$ARGC=	000003		T\$TSTM=	177777	T8	035770	G
SVCSUB=	000001		TXBFCA	002216	G	T\$CODE=	001052		T\$TSTS=	000001	T9	036536	G
SVCTAG=	000001		TXBFCA=	000016	G	T\$ERRN=	022125		T\$\$AU=	010024	UAM	000200	G
SVCTST=	000001		TXCNTB	003502	G	T\$EXCP=	000000		T\$\$AUT=	010021	UBRFMT	007722	G
SWPTQ1	037174		TXDBLF	002506	G	T\$FLAG=	000040		T\$\$CLE=	010022	UNITN	002176	G
SWPTQ2	037250		TXDMA	027426	G	T\$GMAN=	000000		T\$\$DU=	010023	UNSDIV	026362	G
SWPTQ3	037330		TXDONE	025224	G	T\$HILI=	177777		T\$\$HAR=	010036	UPDCHR	026516	G
SWPTQ4	037363		TXDONF	002502	G	T\$LAST=	000001		T\$\$HW=	010000	VANSUP	026614	G
S\$LSYM=	010000		TXDSBL	025334	G	T\$LOLI=	000000		T\$\$INI=	010020	WAIBIS	027012	G
TERMSG	013437	G	TXENBL	025430	G	T\$LSYM=	010000		T\$\$MSG=	010015	WORD1	002266	G
TIMER1	002500	G	TXENBM	002262	G	T\$LTND=	000011		T\$\$PRO=	010017	WTWLC	027066	G
TIMER2	002302	G	TXFRPR	025524	G	T\$NEST=	177777		T\$\$RPT=	010016	WTWLP	027116	G
TIMER3	002304	G	TXIE0	025620	G	T\$NSO=	000000		T\$\$SOF=	010037	X\$ALWA=	000000	
TNUM	000011	G	TXIE1	025660	G	T\$NS1=	000005		T\$\$SW=	010001	X\$FALS=	000040	
TPABRI	027362	G	TXINTF	002264	G	T\$PTNU=	000000		T\$\$TES=	010035	X\$OFFS=	000400	
TPAFLG	002254	G	TXPTRB	003342	G	T\$SAVL=	177777		T1	030546	X\$TRUE=	000020	
TPARTN	027404	G	TXRINI	025704	G	T\$SEGL=	177777		T2	031030	\$PATCH	037452	G
TPAVEC	002256	G	TXROFF	026160	G								

ABS: 037174 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 50240 WORDS (119 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:05:17
CZDHXAC,RTN,CZDHXAC,STP-SP-SVC34R ML,CZDHXAC,P11

HARDWARE TESTB1
HARDWARE TESTC1
HARDWARE TESTD1
HARDWARE TESTE1
HARDWARE TESTF1
HARDWARE TESTG1
HARDWARE TESTH1
HARDWARE TESTI1
HARDWARE TESTJ1
HARDWARE TESTK1
HARDWARE TESTL1
HARDWARE TESTM1
HARDWARE TESTN1

HARDWARE TESTB2
HARDWARE TESTC2
HARDWARE TESTD2
HARDWARE TESTE2
HARDWARE TESTF2
HARDWARE TESTG2
HARDWARE PARAMETER C.....H2
HARDWARE PARAMETER C.....I2
SOFTWARE PARAMETER C.....J2
SOFTWARE PARAMETER C.....K2
SYMBOL TABLEL2
SYMBOL TABLEM2
SYMBOL TABLEN2