

IEC11-A

IEC11-A POP-11 DIAG
CZIEBA0

COPYRIGHT (c) 1976-84
AH-T884A-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

IDENTIFICATION

PRODUCT CODE: AC T883A MC
PRODUCT TITLE: CZIEBA0 IEC11 A PDP 11 DIAGNOSTIC
PRODUCT DATE: JANUARY 1984
DEPARTMENT: CSS, MUNICH
AUTHOR: BERT HUBER

COPYRIGHT (C) FIRST, 1976, 1984
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE
AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR
OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO
AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.
THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
EQUIPMENT CORPORATION.
DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
3.0	LOADING PROCEDURE
4.0	STARTING PROCEDURE
5.0	OPERATING PROCEDURE
6.0	ERROR HANDLING
7.0	RESTRICTIONS
8.0	EXECUTION TIME AND ORIGINAL EQUIPMENT
9.0	DIAGNOSTIC MONITOR DESCRIPTION
10.0	LISTING

 * DIAGNOSTIC PROGRAM DESCRIPTION *

1.0 A B S T R A C T

THIS PROGRAM IS DESIGNED TO TEST AND EXERCISE THE
 IEC-11A IEC BUS CONTROLLER FOR THE PDP/11.

2.0 R E Q U I R E M E N T S

- A. PDP 11/XX SYSTEM
 CONSOLE TERMINAL WITH STANDARD UNIBUS ADDRESSES.
- B. A MINIMUM OF 5K MEMORY.
- C. ALL STANDARD PROCESSOR TESTS MUST BE RUN SUCCESSFULLY

3.0 L O A D I N G P R O C E D U R E

BOOT XXDP MEDIA, THEN RUN DIAGNOSTIC WITH FOLLOWING
 COMMAND:

R ZIEB??

NOTE: ?? REFERS TO VERSION AND REV LEVEL.

4.0 S T A R T I N G P R O C E D U R E

PROGRAM IS STARTED AT LOC 1000 (OCTAL)
 IF NOT OTHERWISE SPECIFIED THE ABOVE NOTED ADDRESS IS
 ALSO THE PROGRAM RESTART ADDRESS.

5.0 O P E R A T I N G P R O C E D U R E

THE FOLLOWING INITIAL DIALOGUE IS PERFORMED AFTER
 THE PROGRAM IS STARTED :

CDM V3.2 -- CSS DIAGNOSTIC MONITOR

HIGHEST MEMORY ADDRESS:XXXXXX

IEC-11A - IEC BUS CONTROLLER BASIC TESTS

AC T883A MC

```

:
: ENTER FIRST REGISTER ADDRESS OF IEC11-A(DEFAULT IS 160010)
: ENTER AND TERMINATE BY TYPING A <CR>
:

```

```

:
: ENTER VECTOR ADDRESS OF IEC11-A(DEFAULT IS 270)
: ENTER AND TERMINATE BY TYPING A <CR>
:

```

```

:
: ENTER IEC BUS ADDRESS OF IEC11-A(DEFAULT IS 35)
: ENTER AND TERMINATE BY TYPING A <CR>
:

```

```

:
: SET LON SWITCH ON IEC11-A TO POS.1 (ACTIVE)
: TYPE <CR> WHEN READY
:

```

```

:
: SET LON SWITCH ON IEC11-A TO POS.2 (INACTIVE)
: TYPE <CR> WHEN READY
:

```

```

:
: DO YOU WANT A PRINTOUT OF START,END MESSAGES?(Y OR N)
:

```

```

:
: DEPENDING ON Y OR N, THE PROGRAM PRINTS OUT THE START
: AND END OF EACH TEST OR NOT.
:

```

```

:
: NEXT TEST TO RUN?
:

```

```

:
: TYPE <CR> TO GO STEP BY STEP THROUGH ALL IEC TESTS (1 - 14).
: OTHERWISE SPECIFY TEST NUMBER (DECIMAL).
: TO SELECT A SINGLE TEST WHILE TEST PROGRAM IS ACTIVE,
: TYPE (CNTRL C) AND THEN "ABO". AFTERWARDS YOU CAN SELECT
: A SINGLE TEST. THIS TEST RUNS IN AN ENDLESS LOOP UNTIL
: THERE IS AN ERROR MESSAGE OR (CNTRL C) AGAIN.
:

```

```

:
: START TEST 1
:

```

```

:
: END TEST 1
:

```

```

:
: START TEST 2
:

```

```

:
: END TEST 2
:

```

```

:
: START TEST 3 .....ETC.
:

```

```

:
: .
:

```

```

:
: .
:

```

```

:
: .
:

```

```

:
: END TEST 14
:

```

```

:
: END OF TESTING
:

```

```

:
: IEC REGISTER ADDRESS MAY BE ENTERED IN FORMAT 16XXX0
: IEC BUS ADDRESS RANGE: 0 ... 36
:

```

6.0 ERROR HANDLING
.....

ALL ERROR MESSAGES ARE EITHER SELF DEFINING OR OF THE
FOLLOWING GENERAL FORM:

 ERROR XXX
WHERE XXX IS AN OCTAL ERROR NUMBER BETWEEN 1 AND 377.
FOR A DETAILED DESCRIPTION OF THE DIFFERENT ERRORS
OF THIS PROGRAM HAVE A LOOK
AT THE APPROPRIATE PAGES LATER IN THE LISTING.

NOTE:: ERRORS MAY NOT OCCUR IN SEQUENCE.
 FOR EXAMPLE:
 ERROR 45 MIGHT OCCUR BEFORE ERROR 21
 THIS DEPENDS ON THE STRUCTURE OF THE PROGRAM.

7.0 RESTRICTIONS
.....
NONE

8.0 EXECUTION TIME AND ORIGINAL EQUIPMENT
.....

THIS DIAGNOSTIC WAS DESIGNED AND TESTED AT CSS/DP
WITH THE ACCORDING CSS HARDWARE EQUIPMENT.
A NORMAL ERROR FREE RUN ON THAT EQUIPMENT
TAKES ABOUT 4 MINUTES

```

; SBTTL IEC ERROR DESCRIPTIONS
; ERROR 1 SMR CONTENT IS ILLEGAL AFTER A MASTER CLEAR
;          SHOULD BE : 0;          WAS: RO
;
; ERROR 2 CIR CONTENT IS NOT ZERO AFTER A MASTER CLEAR
;          SHOULD BE: 0;          WAS: RO
;
; ERROR 3 IOR CONTENT IS NOT ZERO AFTER A MASTER CLEAR
;          SB: 0;          WAS: RO
;
; ERROR 4 SACS BIT IN CIR NOT SET BY "BIS" FUNCTION
;          RO = CIR CONTENT
;
;          RO = CIR CONTENT
;
; ERROR 6 SACS BIT NOT RESET BY MASTER CLEAR
;          RO = CIR
;
; ERROR 7 SIC PRODUCES "ILLEG. MESS." WITH SACS TRUE
;          RO = CIR CONTENT      R1 = SMR
;
; ERROR 10 SIC DOES NOT BRING SIAS BUT NO 'ILLEG. MESS
;          OCCURS.
;          RO = CIR CONTENT      R1 = SIA
;
; ERROR 11 SIAS BIT 0 TO 1 CHANGE DOES NOT PRODUCE A
;          'STATE CHANGE'
;          RO = CIR      R1 = SMR
;
; ERROR 12 SIAS DOES NOT BRING CACS.
;          RO = CIR      R1 = SMR
;
; ERROR 13 SIAS DOES NOT BECOME ZERO AFTER MORE THAN 200 MS.
;          RO = CIR      R1 = SMR
;
; ERROR 14 NO 'STATE CHANGE' AFTER 'SIAS' 1 TO 0 CHANGE.
;          RO = CIR      R1 = SMR
;
; ERROR 15 CACS IS NO LONGER TRUE WHEN 'SIAS' IS RESET TO 0.
;          RO = CIR      R1 = SMR
;
; ERROR 16 MC DOES NOT RESET CACS
;          RO = SMR
;
; ERROR 17 NO <ILLEGAL MESSAGE> TO <SICS> WHEN <SACS>
;          IS NOT SET.
;          RO = CIR
;
; ERROR 20 NO INTERRUPT OCCURED AFTER SENDING <SIC>.
;          (<ILLEGAL MESSAGE> SHOULD OCCUR AND BRING AN INTERRUPT.)
;          RO = CIR

```

```

; ERROR 21    THE INTERRUPT THAT OCCURED AFTER SETTING <SIC>
;             WITHOUT <SACS> WAS NOT, OR NOT EXCLUSIVELY CAUSED
;             BY <ILLEGAL MESSAGE>.
;             BITS 8-15 OF R5 = 'SHOULD BE' CONTENT OF CIR BITS 8 15
;             R0 = ACTUAL CONTENT OF CIR

; ERROR 22    SETTING OF <SACS> AND <SIC> SHOULD BRING <SIAS>,
;             WHICH IN TURN SHOULD FORCE AN INTERRUPT BY SETTING
;             <STATE CHANGE>. BUT NO INTERRUPT OCCURED.
;             R0 = CIR
;             R1 = SMR

; ERROR 23    THE INTERRUPT THAT OCCURED AFTER SETTING <SACS>
;             AND <SIC> WAS NOT OR NOT EXCLUSIVELY CAUSED BY
;             <STATE CHANGE>.
;             BITS 8-15 OF R5 = 'SHOULD BE' VALUE OF CIR BITS 8 15
;             R0 = ACTUAL CONTENT OF CIR

; ERROR 24    <SRE> IN SMR NOT SET
;             R0 = CIR          R1 = SMR

; ERROR 25    <SRE> PRODUCES <ILLEGAL MESSAGE> WITH <SACS> SET.
;             R0 = CIR CONTENT  R1 = SMR CONTENT

; ERROR 26    <SRE> DOES NOT BRING <SRAS> BUT NO <ILLEGAL MESSAGE> OCCURS.
;             R0 = ICR          R1 = SMR

; ERROR 27    <SRAS> 0 TO 1 CHANGE DOES NOT PRODUCE <STATE CHANGE>.
;             R0 = CIR          R1 = SMR

; ERROR 30    <SRE> IN SMR CAN T BE CLEARED BY A 'BIC' INSTRUCTION.
;             R0 = CIR          R1 = SMR

; ERROR 31    CLEARING OF <SRE> DOES NOT RESET <SRAS> IN SMR.
;             R0 = CIR          R1 = SMR

; ERROR 32    CLEARING OF <SRE> DOES NOT OR NOT ONLY PRODUCE
;             A <STATE CHANGE>.
;             R0 = CIR          R1 = SMR

; ERROR 33    <INTERRUPT ENABLE> IN CIR CAN NOT BE SET.
;             R0 = CIR

; ERROR 34    <INTERRUPT ENABLE> IN CIR CAN NOT BE CLEARED.
;             R0 = CIR

; ERROR 35    <CSBS> NOT SET AFTER SETTING <GTS>.
;             R0 = CIR          R1 = SMR

; ERROR 36    <CSBS> SETTING DOES NOT PRODUCE <STATE CHANGE>.
;             R0 = CIR          R1 = SMR

```

```

; ERROR 37 NO <ILLEGAL MESSAGE> TO SETTING OF <GTS>
; IN 'STAND BY' STATE.
; RO = CIR R1 = SMR

; ERROR 40 <TCA> IN <CSBS> DOES NOT OR NOT ONLY PRODUCE <STATE
; CHANGE>. RO = CIR R1 = SMR

; ERROR 41 <TCA> IN <CSBS> DOES NOT OR NOT ONLY PRODUCE <CACS>.
; RO = CIR R1 = SMR

; ERROR 42 WHEN ADDRESSING THE TALKER IN <CACS>, NO INTERRUPT
; OCCURED.
; THE INTERRUPT SHOULD HAVE BEEN SET BY <DATA ACCEPTED>.
; RO = CIR R1 = SMR R2 = IOR

; ERROR 43 AN INTERRUPT OCCURED BY ADDRESSING THE TALKER,
; BUT IT WAS NOT OR NOT ONLY CAUSED BY <DATA ACCEPTED>.
; RO = CIR R1 = SMR R2 = IOR

; ERROR 44 AFTER SUCCESSFULLY ADDRESSING THE TALKER, IT IS NOT
; POSSIBLE TO SET <CSBS> AND <TACS> CORRECTLY BY <GTS>.
; RO = CIR R1 = SMR R2 = IOR

; ERROR 45 WHEN SENDING A DATA BYTE WITHOUT ADDRESSING THE LISTENER,
; NO INTERRUPT OCCURED.
; RO = CIR R1 = SMR R2 = IOR

; ERROR 46 <TCS> DOES NOT PRODUCE <ILLEG. MESSAGE>
; RO = CIR R1 = SMR

; ERROR 47 LOW ORDER BYTE OF IOR <BITS 0 7> CNA NOT BE SET OR
; READ BACK CORRECTLY.
; SHOULD BE = ALL ONES 'IS' = RO
; BITS 8 15 CAN BE IGNORED BUT SHOULD BE ZERO.

; ERROR 50 WHEN SENDING THE LISTENER ADDRESS, NO INTERRUPT OCCURS.
; RO = CIR R1 = SMR R2 = IOR

; ERROR 51 SENDING THE LISTENER ADDRESS FORCES AN INTERRUPT THAT
; WAS NOT OR NOT ONLY CAUSED BY <DATA ACCEPTED>.
; RO-R2 SAME AS ERROR 50

; ERROR 52 AFTER LISTENER ADDRESSING AND <GTS> NO <LACS> IS SET.
; RO R2 SAME AS ERROR 50

; ERROR 53 NO INTERRUPT TO TALKER ADDRESSING
; RO R2 SEE ERROR 50

; ERROR 54 WHEN SENDING A SECOND DATA BYTE BEFORE READING THE FIRST
; AGAIN <DATA ACCEPTED> CAME UP.
; RO-R2 SEE ERROR 50

```

```

: ERROR 55      FIRST DATA BYTE SENT AND RECEIVED IS NOT THE SAME.
:               RO R2 SEE ERROR 50

: ERROR 56      AFTER READING THE FIRST SENT DATA BYTE, THE SECOND SENT
:               DID NOT BRING <DATA ACCEPTED>.
:               RO R2 SEE ERROR 50

: ERROR 57      SECOND DATA BYTE SENT AND RECEIVED IS NOT THE SAME.
:               RO-R2 SEE ERROR 50

: ERROR 60      SENDING OF DATA BYTE DOES NOT PRODUCE AN INTERRUPT.
:               RO R2 SEE ERROR 50

: ERROR 61      SENDING A DATA BYTE FORCES AN INTERRUPT THAT WAS NOT
:               OR NOT ONLY PRODUCED BY <DATA ACCEPTED>.
:               RO-R2 SEE ERROR 50

: ERROR 62      DATA BYTES SENT AND RECEIVED ARE NOT EQUAL.
:               RO-R2 SEE ERROR 50

: ERROR 63      NO <DATA ACCEPTED> WHEN SENDING A BYTE WITH <LAST BYTE> SET.
:               RO-R2 SEE ERROR 50

: ERROR 64      NO <END> TO <LAST BYTE>
:               RO-R2 SEE ERROR 50

: ERROR 65      DATA CHECK ON: <LAST BYTE>
:               RO-R2 SEE ERROR 50

: ERROR 66      NO <LACS> TO <LTN>
:               RO-R2 SEE ERROR 50

: ERROR 67      NO <LCAS> AFTER THE FOLLOWING COMMAND SEQUENCE.
:               [LTN-GTS-TCA-GTS]
:               RO-R2 SEE ERROR 50

: ERROR 70      <LACS> STILL SET AFTER THE FOLLOWING SEQUENCE:
:               [LTN-GTS-TCA-LUN-GTS]
:               RO R2 SEE ERROR 50

: ERROR 71      <BLOCK DAC> CAN NOT BE SET
:               RO-R2 SEE ERROR 50

: ERROR 72      <BLOCK DAC> CAN NOT BE RESET
:               RO-R2 SEE ERROR 50

: ERROR 73      SENDING A BYTE WITH <BLOCK DAC> FORCES AN INTERRUPT
:               RO-R2 SEE ERROR 50

: ERROR 74      DATA BYTES SENT AND RECEIVED WITH <BLOCK DAC> ARE
:               NOT EQUAL.
:               RO R2 SEE ERROR 50

```

```

; ERROR 75      NO <SRQ> TO <RSV>
;               RO R2 SEE ERROR 50

; ERROR 76      NO <SPAS> TO <SPE>
;               RO R2 SEE ERROR 50

; ERROR 77      DATA CHECK ON STATUS BYTE
;               RO-R2 SEE ERROR 50

; ERROR 100     <SRQ> NOT OFF AFTER STATUS BYTE RECEIVED
;               RO R2 SEE ERROR 50

; ERROR 101     NO <TACS> AFTER <SPD>
;               RO R2 SEE ERROR 50

; ERROR 102     <RPP> CAN NOT BE SET
;               RO-R2 SEE ERROR 50

; ERROR 103     <RPP> CAN NOT BE RESET
;               RO-R2 SEE ERROR 50

; ERROR 104     NO <ILLEG. MESSAGE> WHEN SENDING <SRE>
;               WITHOUT SENDING <SACS> BEFORE.
;               RO-R1 SEE ERROR 50

; ERROR 105     BIT 10 IN CIR NOT SETTABLE BY "BIS"
;               INSTRUCTION (INHIBIT SRQ INTERRUPT)

; ERROR 106     BIT 10 IN CIR NOT RESETTABLE BY "BIC"
;               INSTRUCTION

; ERROR 107     SEE ERROR 105

; ERROR 110     BIT 10 IN CIR NOT RESETTABLE BY MASTER CLEAR

; ERROR 111     BIT 7 (INTERRUPT) SHOULD BE SET WITH
;               CLEAR SRQ INT INHIBIT BIT (BIT 10 IN CIR)

; ERROR 112     BIT 7 (INTERRUPT) NOT CLEAR WITH INHIBIT
;               SRQ INTERRUPT BIT SET (CIR)

; ERROR 113     BIT 7 IN CIR NOT SET AGAIN WITH
;               INHIBIT SRQ INT BIT CLEAR AGAIN

```

```
; ERROR 114    CIR SHOULD CONTAIN #602 (STATE CHGE,INT,LON)
;
;
; ERROR 115    SMR SHOULD CONTAIN ONLY BIT 12 (LACS)
;
;
; ERROR 116    CIR SHOULD CONTAIN #600 (STATE CHANGE,INT)
;              LON MUST BE CLEAR AGAIN
;
; ERROR 117    LACS DISSAPPEARED WITH LON SWITCH IN INACTIVE
;              POSITION (POS. 2) AFTER FORMER ACTIVATION
;              OF THIS SWITCH.
```

```

*****
* PRODUCT CODE: AC T883A-MC
*
* PRODUCT NAME: IEC-11A MAIN TEST
*
* DATE: 20-JUN-76      (V01E00)
*       29-SEP-76      (V01E01)
*       27 JAN-77      (V01E02)
*       01-MAR-77      (V02E00)
*       01-APR-77      (V02E01)
*       01 FEB-80      (V03E02)
*       22 APR 80      (V03E03)
*       12-MAR-84      (ZIEBA0)
*
* MAINTAINER:  CSS/DP MUNICH
*
* AUTHOR: BERT HUBER CSS
*
* UPDATED TO V01E02 BY B.BAUDISCH
* UPDATED TO V02E00 BY B.BAUDISCH
* UPDATED TO V02E01 BY B.BAUDISCH
* UPDATED TO V03E02 BY G. SZENTES
* UPDATED TO V03E03 BY G. SZENTES
* UPDATED TO ZIEBA0 BY J. LEVESQUE
*****

```

9.0 DIAGNOSTIC MONITOR DESCRIPTION

STARTING PROCEDURE

A) USING A PROCESSOR WITH A SWITCH PANEL

PLACE START ADDRESS INTO SWITCH REGISTER
(START ADDRESS IS 1000 OCTAL)

PRESS "LOAD ADDRESS" SWITCH

PRESS "START" SWITCH

PROGRAM SHOULD BEGIN TO PRINT OUT THE IDENTIFICATION
PRINTOUT.

B) USING A LSI11

TYPE IN ST. ADDR., (1000 OCTAL) AND 'G' INTO ONE LINE.

PROGRAM STARTS WITH IDENTIFICATION PRINTOUT.

CAUSING A PROCESSOR WITHOUT A CONSOLE (11/04,11/34)

PRESS "CNTRL" AND "BOOT" BUTTONS. ODT CONSOLE
ROUTINE PRINTS A \$ SIGN.
TYPE IN "L 1000" AND A <CR>.
TYPE AFTER THE "\$ AN S" AND <CR>. PROGRAM SHOULD BEGIN TO
PRINT IDENTIFICATION PRINTOUT.

OPERATING PROCEDURES

TO OPERATE AN ERROR SEE UNDER A)
TO MODIFY A LOCATION SEE UNDER C)
TO DUMP A LOCATION SEE UNDER B)
TO ABORT THE RUNNING PROGRAM SEE UNDER D)

A) ALL INPUT/OUTPUT OPERATIONS ARE DONE VIA THE DIAGNOSTIC
MONITOR. NO SWITCH SETTINGS ARE USED.
ALL COMMANDS, REQUIRED WHEN AN ERROR OCCURS, ARE
ENTERED VIA THE TERMINAL.
THESE COMMANDS SET OR CLEAR BITS IN A MEMORY LOCATION
CALLED PSEUDO-SWITCH REGISTER.
THE FOLLOWING OPTIONS ARE AVAILABLE:

/HT	=	HALT AFTER ERROR (DEFAULT)
/NH	=	NO HALT AFTER ERROR
/PR	=	PRINT ERROR MESSAGE (DEFAULT)
/NP	=	NO ERROR PRINTOUT
/CT	=	COUNT ERRORS
/NC	=	NO ERROR COUNT (DEFAULT)
/SC	=	SCOPE LOOP ON THIS ERROR
/NS	=	NO SCOPE LOOP (DEFAULT)
/CONT	=	CONTINUE SWITCH

WHEN AN ERROR OCCURS, THE FOLLOWING TEXT IS PRINTED:

```

ERROR 000xxx
AT LOCATION: xxxxxx
... HALT AFTER ERROR ...

```

THE ERROR NUMBER CAN BE IN THE RANGE 0 377.
THE LOCATION ADDRESS IS ABSOLUTE.

WHEN AN ERROR IS FOUND, THE PROGRAM IS INTERRUPTED.
THE ERROR NUMBER, ERROR LOCATION AND PSEUDO STOP MESSAGE
ARE PRINTED. THEN THE DIAGNOSTIC MONITOR WAITS FOR
INPUT OF OPTIONS.

PRESS <CNTRL. C> AND THE 'PROMPT STRING' IS PRINTED AT THE BEGINNING OF A NEW LINE.

CDM>
 *** (PROGRAM PRINTS WILL BE UNDERLINED FROM NOW)

NOW YOU CAN SELECT THE OPTIONS YOU NEED TO FIND THE ERROR. THE SYNTAX OF A COMMAND LINE IS AS FOLLOWS:

CDM>ERR=/OPT1/OPT2/OPT3/...OPTN/CONT <CR>

 IF THERE IS A ERROR IN THE INPUT LINE THE FOLLOWING MESSAGE IS PRINTED:

SYNTAX ERROR !

YOU CAN NOW REPEAT THE INPUT.

ANY COMBINATION OF OPTIONS IS LEGAL, BUT PROBABLY NOT USEFUL. WHEN YOU TYPE /SC/NS THE FIRST OPTION IS OVERWRITTEN. THE COMMAND LINE IS INTERPRETED FROM LEFT TO RIGHT.

THE LAST OPTION IN A LINE MUST BE /CONT.

THEN THE MONITOR RETURNS CONTROL TO THE POINT OF PROGRAM WHERE THE ERROR WAS ENCOUNTERED AND THE PROGRAM CONTINUES WITH THE ENTERED OPTIONS.

8) OPTION : DMP
 THIS OPTION CAN BE USED TO CHECK THE CONTENT OF A MEMORY LOCATION OR A REGISTER. IT CAN ONLY BE CALLED AFTER AN ERROR OR WHEN YOU INTERRUPTED THE RUNNING PROGRAM BY PRESSING "CNTRL C".
 EXAMPLE:
 AFTER AN ERROR PRINT YOU PRESSED "CNTRL C" AND THE MONITOR RETURNED THE PROMPT STRING:

CDM>

IF YOU WANT TO CHECK LOCATION 1000 AND IT CONTAINS E.G. 012767, SO YOU HAVE TO TYPE AFTER THE PROMPT STRING

CDM>DMP/001000<CR>

 012727

THE MONITOR WILL PRINT THE CONTENT OF LOCATION 1000.

THE MONITOR RETURNS NOW INTO ITS WAIT ROUTINE AND WAITS FOR FURTHER INPUTS.

- C) OPTION : SET
 THIS OPTION CAN BE USED TO MODIFY THE CONTENT OF A MEMORY LOCATION OR OF A REGISTER.
 IT CAN BE CALLED ONLY AFTER AN ERROR OR WHEN YOU INTERRUPT THE RUNNING PROGRAM. TO CALL THE MONITOR PRESS "CNTRL C" AND THE MONITOR ANSWERS, AS PREVIOUS SEEN, BY PRINTING

CDM>

IF YOU WANT TO CHANGE LOCATION 1000, YOU HAVE TO TYPE ON THE SAME LINE

CDM>SET/001000/000000<CR>

THE LOCATION CONTAINS NOW ZERO. INPUT OF NEW MEMORY LOCATION CONTENT MUST BE SIX OCTAL DIGITS LONG WITH LEADING ZERO'S.

- D) OPTION : ABO
 THIS OPTION CAN BE USED ONLY AFTER AN ERROR OR WHEN YOU INTERRUPT THE RUNNING PROGRAM BY PRESSING "CNTRL C". USING THIS OPTION ALLOWS TO ABORT THE PROGRAM AT CURRENT POINT OF TEST AND EITHER TO GO TO BEGIN OF TEST, OR IF SUPPORTED, TO CALL A SUBTEST.
 AFTER PRESSING "CNTRL C" THE MONITOR RETURNS WITH

CDM>

TYPE NOW ON THE SAME LINE

CDM>ABO<CR>

AND THE PROGRAM WILL GO TO BEGIN OF TEST IF YOU ENTER 0 OR <CR> OR TO A ROUTINE WHERE YOU CAN SELECT A SUBTEST.

- 10.0 LISTING

2	4	LOW CORE
4	53	INITIALISATION
5	103	MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
5	215	\$..MAP MAPPING ROUTINE
7	267	\$..EMT
8	329	\$..TRP
9	399	\$..IOT
10	430	\$..PWR
10	482	\$..RSV
11	508	\$..RRS
12	536	\$..KBI
13	590	\$..INP
14	661	\$..IAY
15	712	\$..IAA
16	759	\$..IAD
18	873	\$..IAE
19	979	\$..RED
20	1162	\$..WRT
21	1230	\$..KBO
22	1252	\$..DPT
23	1285	\$..ADP
24	1317	\$..TOT
26	1418	\$..BUF
27	1437	TEST SELECT ROUTINE
28	1463	LOCAL MACRO DEFINITIONS
30	62	TEST LON SWITCH FEATURE
32	107	TEST 1 EXECUTE MASTER CLEAR
33	209	TEST 2 TEST SACS BIT IN CIR
34	234	TEST 3 TEST CIR BIT 10 INHIBIT SRQ INTERRUPT
35	255	TEST 4 TEST SACS SIC SIAS ILLMSG CACS
36	284	TEST 5 TEST INT ENB
37	314	TEST 6 TEST INTERRUPT WITH STATE CHGE
38	379	TEST 7 TEST STATE CHGE INTERRUPT WITH SRAS
39	402	TEST 8 - IOR DATA TEST
40	444	TEST 9 TEST STATE CHGE INTERRUPT WITH GTS AND SIC
41	514	TEST 10 TEST DATA TRANSFER WITH INTERRUPT
42	534	TEST 10A -- RANDOM TALKER AND LISTENER DATA TEST
43	646	TEST 11 -- TEST FUNCTION OF <LTN> AND <LUN>
44	770	TEST 12 - TEST FUNCTION OF <BLOCK DAC>
45	849	TEST 13 -- TEST FUNCTION OF <RSV>, <SRQ> AND <SPAS>
46	910	TEST 14 --- TEST RPP BIT R/W
47	970	IEC-INTERRUPT HANDLER
48	1064	SUBTEST ADDRESS TABLE
49	1089	MESSAGES
50	1118	CONSTANTS AND PARAMETERS
51	1120	
52	1178	

1
2 000000
3
4
5
6 000000

```
.ENABL AMA
.ENABL ABS
.LIST ME
.SBTTL LOW CORE
.MCALL INIT
INIT
.LIST ME
;
; COPYRIGHT (C) 1976
; DIGITAL EQUIPMENT CORPORATION, MAYNARD MASSACHUSETTS
;
; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ON A
; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE
; INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR
; ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE
; MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH
; SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE
; TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN
; IN DEC.
;
; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
; EQUIPMENT CORPORATION.
;
; DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
; ITS SOFTWARE ON EQUIPEMENT WHICH IS NOT SUPPLIED BY DEC.
;
;
```

```

7 000000      .PSECT  CSSMON
8
9      ;
10     ;      MZ001
11     ;      *****
12     ;      THIS CHANGE ALLOWS TO USE A MACRO THAT EITHER CLEARS
13     ;      THE PSW BY A "CLR" OR A "MTPS".
14     ;
15     ;      GS001
16     ;      *****
17     ;      THIS CHANGE ALLOWS TO START THE PROGRAM AT LOC. 200
18     ;
19 000000      .ASECT
20              .=0
21              .REPT  4
22              .+2
23              IOT
24              .ENDR
25 000020      003056      $..IOT
26 000022      000000      0
27 000024      003126      $..PWR
28 000026      000000      0
29 000030      002326      $..EMT
30 000032      000000      0
31 000034      002524      $..TRP
32 000036      000000      0
33              .REPT  30
34              .+2
35              IOT
36              .ENDR
37 000200      000137      001000      JMP 10$
38              000137      .REPT  137
39              .+2
40              IOT
41              .ENDR
42 001000      000137      001204      10$: JMP START
43              000100      .REPT  100
44              0
45              .ENDR

```

```

;POINT TO VECTOR +2
;HANDLE ILLEGAL TRAP
;
;ILLEGAL TRAP-HANDLER
;PRIORITY = 0
;POWERFAIL-RESTART-HANDLER
;PRIORITY = 0
;EMT-HANDLER
;PRIORITY = 0
;TAP-HANDLER
;PRIORITY = 0
;
;POINTER TO VECTOR +2      ;**GS001**
;HANDLE ILLEGAL TRAP      ;**GS001**
;
;
;INITIALIZE SYSTEM          ;**GS001**
;
;POINTER TO VECTOR +2      ;**GS001**
;HANDLE ILLEGAL TRAP      ;**GS001**
;
; INITIALIZE SYSTEM        ;**GS001**
;
;STACK AREA

```

```

47      ;
48      ;      MZ001
49      ;      *****
50      ;      AS VERY FIRST INSTRUCTION DO A "RESET" TO CLEAR
51      ;      ALL HARDWARE THAT IS IN AN UNDEFINED STATE
52      ;
53      ;      .SBTTL  INITIALISATION
54 001204 000005      START: RESET      ;CLEAR THE WORLD      ;MZ001
55 001206 012706 001204      MOV      #START,SP      ;SETUP SP
56 001212 012737 001242 000004      MOV      #$.TYP,4      ;PREP FOR TEST
57 001220 005037 000006      CLR      6      ;
58 001224 005037 177776      CLR      @@PS      ;IS IT A PDP OR A LSI ?
59 001230 000240      NOP      ;
60 001232 012737 177777 007016      MOV      @-1,$PTYP      ;IT IS A PDP
61 001240 000404      BR      TYP..E      ;SKIP LSI PART
62 001242 005037 007016      $.TYP: CLR      $$PTYP      ;INDICATE A LSI
63 001246      STCKUP
64 001252 012737 000006 000004      TYP..E: MOV      @6,4      ;REBUILD
65 001260 012737 000004 000006      MOV      @4,6      ;
66 001266 012737 003424 000060      MOV      #$.KBI,@@60      ;SETUP KEYBOARD VECTOR
67 001274 012737 000340 000062      MOV      @340,@@62      ;
68 001302 012700 007026      MOV      @KBBUFF,R0      ;RESET ALL FLAGS
69 001306 005020      1$: CLR      (R0),      ;
70 001310 020027 007202      CMP      R0,@PROMFS      ;
71 001314 001374      BNE      1$      ;
72 001316 012737 001000 007172      MOV      @1000,SCOPAD      ;IF NO SCOPE LABEL IS SET IN THE PROGRAM
73      ;      ;THEN GOTO START AFTER ERR=7SC
74 001324      PSET      @0      ;SET PS TO 0      ;MZ001
75 001350      WRITE     MVERSN      ;SAY HELLO TO EVERYBODY
76 001360 012737 007026 007146      MOV      @KBBUFF,KBBPNT      ;SETUP INPUT-BUFFER-POINTER
77 001366 004737 001516      JSR      PC,M..TST      ;CALL MEM. TEST WITH MEM. MANAGEM. SETUP ;BB001
78 001372 000414      BR      M..SAV      ;GO TO SAVE HIGHEST MEMORY ADDRESS      ;BB001
79 001374      M..TSE:      ;HERE IF NO MEMORY MANAGEMENT FOUND      ;BB001
80 001374 012700 160000      MOV      @160000,R0      ;SETUP MAXIMUM CORE-ADDRESS
81 001400 005001      CLR      R1      ;SETUP FOR MEMEND+2 FOR UNMAPPED PROC. ;BB001
82 001402 012737 001410 000004      MOV      @. +6,@@4      ;SETUP TRAPHANDLING
83 001410 012706 001204      MOV      #START,SP      ;REBUILD STACKPOINTER
84 001414 005740      TST      -(R0)      ;TOP OF CORE?
85 001416      REMVEC      @2      ;REBUILD TRAP TO 4 VECTOR
86 001424      M..SAV:      ;HERE IF HIGHEST MEMORY ADDRESS FOUND      ;BB001
87 001424 010037 007022      MOV      R0,MEMEND      ;SAVE HIGHEST MEMORY ADDR.
88 001430 010137 007024      MOV      R1,MEMEND+2      ;SAVE EXTENSION BITS
89 001434 006201      ASR      R1      ;PUT ADDR16 INTO CARRY      ;BB001
90 001436 006000      ROR      R0      ;ROTATE ADDR16 INTO MSB OF R0      ;BB001
91 001440 006201      ASR      R1      ;PUT ADDR17 INTO CARRY      ;BB001
92 001442 006000      ROR      R0      ;ROTATE ADDR17 INTO MSB OF R0      ;BB001
93 001444 000241      CLC      ;RESET CARRY      ;BB001
94 001446 006000      ROR      R0      ;SHIFT OUT THE LS-OCT DIGIT OF PHYS ADDR ;BB001
95 001450      DUMP      OCT,R0      ;DUMP ADDRESS
96 001460      WRITE     NO6      ;PRINT LS-OCT-DIGIT ALWAYS AS "6"      ;BB001
97 001470 052737 000100 177560      BIS      @BIT6,@@TKS      ;ENABLE ITY INTERRUPTS
98 001476 012737 000042 000040      MOV      @42,@@40      ;PREP THIS LOCATION FOR USING XXDP
99 001504 005737 007152      TST      OUTFLG      ;SYNCHRONISE OUTPUT
100 001510 001375      BNE      . 4      ;
101 001512 000137 010614      JMP      CDM..E      ; NOW WE ARE ON THE AIR

```

```

103          .SBTTL  MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
104
105          ; SUBROUTINE:  M..TST
106          ; *****
107
108          ; THIS ROUTINE CHECKS THE AVAILABILITY OF A MEMORY MANAGEMENT UNIT
109          ; IN PDP 11 PROCESSORS AND SETS KISAR0...KISAR5 TO THE LOW 24K-WORDS
110          ; OF THE COMPUTER MEMORY .
111          ; KISAR7 IS USED TO MAP THE SO-CALLED I/O PAGE .
112          ; KISAR6 IS USED TO MAP THE HIGHEST MEMORY ADDRESS CHECK POINTER .
113
114          ; INPUT:          (SP)      *RETURN ADDRESS
115
116          ; OUTPUT:         R0        *LOW 16 ADDRESS BITS OF HIGHEST MEMORY ADDRESS
117          ;                  R1        *BITS 0,1 ADDRESS BITS 16,17
118
119          ; AUTHOR:         BERNHARD BAUDISCH CSS/DP MUNICH 19 DEC 78
120
121          ;
122          M..TST:
123 001516      012737 001776 000250      MOV      @M..TRP,@0250      ;REFERENCE LABEL
124 001524      005037 000252              CLR      @0252          ;SETUP SEGMENTATION TRAP CATCHER
125 001530      012737 001776 000004      MOV      @M..TRP,@04      ;SETUP TRAP-4 CATCHER
126 001536      005037 000006              CLR      @06          ;...
127 001542      004737 002006              CALL     M..MMM          ;SETUP DEFAULT MAPPING,ENABLE MM
128 001546      012737 007400 172354      MOV      @7400,KISAR6      ;MAP WITH KISAR6 THROUGH MEMORY
129 001554      012737 000001 007020      MOV      @1,,$MTYP        ;REMEMBER THAT WE HAVE MEMORY MANAGEMENT!
130 001562      012700 157776              MOV      @157776,R0      ;GET HIGHEST ADDRESS WITHIN APR6 RANGE
131 001566      012737 001766 000004      MOV      @M..HIM,@04      ;SET UP TRAP 4 CATCHER
132 001574      005710              20$:    TST      @R0          ;SEE IF ADDRESS PRESENT
133 001576      103031              BCC      M..DET          ;CC IF ADDRESS FOUND
134 001600      162737 000200 172354      SUB      @200,KISAR6      ;MAP TO NEXT LOWER 4K WORD BANK
135 001606      002401              BLT      30$            ;THIS CASE SHOULD NEVER HAPPEN
136 001610      000771              BR       20$            ;CHECK NEXT LOWER 4-K WORDS
137 001612              30$:
138 001612              PSET      @0          ;HERE ONLY IF MAIN PROCESSOR HARDWARE ERROR
139 001636              WRITE     MEMDEF      ;SET PRIORITY TO 0
140 001646      012716 001374              MOV      @M..TSE,(SP)      ;MEMORY MANAGEMENT DEFECT
141 001652      042737 000001 177572      BIC      @BIT0,SRO      ;MODIFY RETURN ADDRESS TO UNMAPPED SYSTEM
142 001660      000422              BR       M..RET          ;DISABLE MM
143 001662              M..DET:
144 001662              PUSH      KISAR6      ;AND GO AHEAD LIKE UNMAPPED SYSTEM
145 000005      .REPT      5          ;HIGHEST MEMORY ADDRESS FOUND
146              ASL      (SP)          ;GET MAP REGISTER CONTENTS
147              .ENDR
148 001700      103402              BCS      40$            ;PUT ADDRESS BIT 17 INTO PSW CARRY
149 001702      005001              CLC          ;
150 001704      000402              BR       45$            ;CS IF ADDRESS BIT 17 SET
151 001706      012701 000001      40$:    MOV      @BIT0,R1      ;HERE IF BIT17 NOT SET
152 001712      006316      45$:    ASL      (SP)          ;...
153 001714      006101              ROL      R1          ;FLAG BIT 17 FOUND
154 001716      042700 160000      BIC      @160000,R0      ;GET ADDRESS BIT 16
155 001722      050016              BIS      R0,(SP)          ;INSERT BIT IN MEMORY EXTENSION MASK
156 001724      012600              MOV      (SP)+,R0      ;MASK OUT APR SELECTION BITS
157 001726      012737 000252 000250      M..RET: MOV      @252,@0250      ;BUILD 16 BIT BASE ADDRESS
158 001734      012737 000004 000252      MOV      @4,@0252      ;-->R0 FOR OUTPUT TO CALLER
159 001742      012737 000006 000004      MOV      @6,@04      ;RESET SEGMENTATION TRAP CATCHER
159 001742      012737 000006 000004      MOV      @6,@04      ;RESET TRAP-4 CATCHER

```

```

160 001750 012737 000004 000006      MOV    #4,006      ;...
161 001756 000207                    RTS    PC          ;RETURN TO CALLER
162 001760                    U..DET:  ;HERE IF MEMORY MANAGEMENT ADDRESS NOT FOUND
163 001760 012716 001374      MOV    #M..TSF,(SP)      ;MODIFY RETURN ADDRESS
164 001764 000760                    BR     M..RET       ;RETURN COMMON
165 001766                    M..HIM:  ;TRAP-4 AND SEGMENTATION TRAP CATCHER
166 001766 052766 000001 000002      BIS    #BIT0,2(SP)  ;SET CARRY IN RETURN PSW SETUP
167 001774 000002                    RTI                ;RETURN WITH CARRY SET INDICATOR
168                                ;
169 001776                    M..TRP:  ;
170 001776 022626                    CMP    (SP)+,(SP)+  ;CLEAN STACK OF INT. VALUES
171 002000 005726                    TST    (SP)+        ;CLEAN STACK FROM RETURN ADDRESS
172 002002 000137 001760      JMP     U..DET            ;NO MEMORY MANAGEMENT UNIT AVAILABLE
173                                ;
174                                ;
175                                ;
176                                ;      SUBROUTINE M..MMM
177                                ;      *****
178                                ;
179                                ;      THIS ROUTINE IS CALLED TO SET UP THE APR AND PDR
180                                ;      REGISTER WITH DEFAULT VALUES AND TO ENABLE MEMORY
181                                ;      MANAGEMENT.
182                                ;
183                                ;      INPUT:  NONE
184                                ;      OUTPUT: ALL PDRS ARE PRESET WITH 77406
185                                ;              APRO IS PRESET WITH 0
186                                ;              APR1    "          200
187                                ;              APR2    "          400
188                                ;              .
189                                ;              .
190                                ;              APR7    "          7600
191                                ;
192                                ;      MM IS ENABLED
193                                ;
194                                ;
195 002006                    M..MMM:  ;
196 002006                    PUSH    R0,R1,R2          ;SAVE
197 002014 012700 077406      MOV    #77406,R0         ;GET PAGE DESCRIPTOR REGISTER SETUP
198 002020 012701 000010      MOV    #8.,R1           ;SETUP FOR 8 PDR'S
199 002024 012702 172300      MOV    #KISDR0,R2        ;GET KISDR0 ADDRESS
200 002030 010022      10$:  MOV    R0,(R2)+          ;SETUP NEXT PDR
201 002032 005301      DEC     R1                     ;COUNT SETUPS
202 002034 001375      BNE     10$                    ;NE IF MORE TO DO
203 002036 012702 172340      MOV    #KISAR0,R2        ;GET KISAR BASE ADDRESS
204 002042 005022      CLR     (R2)+                  ;SETUP KISAR 0
205 002044 012722 000200      MOV    #200,(R2)+        ;SETUP KISAR 1
206 002050 012722 000400      MOV    #400,(R2)+        ;SETUP KISAR 2
207 002054 012722 000600      MOV    #600,(R2)+        ;SETUP KISAR 3
208 002060 012722 001000      MOV    #1000,(R2)+       ;SETUP KISAR 4
209 002064 012722 001200      MOV    #1200,(R2)+       ;SETUP KISAR 5
210 002070 012722 001400      MOV    #1400,(R2)+       ;SETUP KISAR 6
211 002074 012712 007600      MOV    #7600,R2         ;SETUP KISAR 7 TO I/O PAGE
212 002100 052737 000001 177572      BIS    #BIT0,SRO   ;ENABLE MEMORY MANAGEMENT
213 002106                    POP     R2,R1,R0         ;RESTORE REGS.
214 002114 000207      RETURN                ;BACK TO CALLER
215                    .SBTTL  $..MAP MAPPING ROUTINE

```

```

217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237 002116
238 002122
239 002136 000003
240 002140 000137 002316
241 002144 004737 002006
242 002150 013700 007022
243
244
245
246 002170 042700 176000
247 002174 013701 007024
248
249
250
251 002224 060001
252 002226 042737 000001 177776
253 002234 020166 000006
254 002240 002023
255 002242
256 002262 052700 000001
257 002266
258 002306 000403
259 002310 016637 000006 172354
260 002316
261 002322 012616
262 002324 000207
263
264
265

```

```

: SUBROUTINE: $.MAP
: *****
:
: THIS ROUTINE IS USED TO MAP WITH APR6 TO THE PHYSICAL MEMORY
: ABOVE 28. K AND COMPARE DMA READ AND WRITE DATA
:
: INPUT: (SP)      =RETURN ADDRESS
:         2(SP)    =100 OCTAL BYTE BLOCK OFFSET
:         4(SP)    =MAPPING POINTER
:
: OUTPUT: KERNEL APR6 IS MAPPED TO BUFFER START ADDRESS
:          MAPPING POINTER IS SET TO BASE ADDRESS OF APR6=140000
:          CARRY IS SET IF NO MEMORY AVAILABLE
:          BREAKPOINT TRAP, IF YOU TRY TO MAP TO CDM (0 4.K)
:
: AUTHOR: ALBERT BREM CSS MUC 16-JAN-79
:
$.MAP: PUSH    RO,R1      ;SAVE REGISTERS
        IF 6(SP) GE #200 GOTO 10$;DONT MAP TO SYSTEM AREA
        BPT          ;DONT MAP TO SYSTEM AREA
        JMP         2$    ;LEAVE
10$:    CALL      M..MMH   ;ENABLE MM,SETUP DEFAULT MAPPING
        MOV       MEMEND,RO ;SAVE HIGHEST MEMORY LOCATION
        .REPT     6
        ROR       RO      ;COMPUTE 100 OCTAL BYTE BLOCKS
        .ENDR
        BIC       #176000,RO ;"
        MOV       MEMEND+2,R1 ;SAVE MEMORY EXTENSION
        .REPT     10
        ASL       R1      ;COMPUTE ADDITIONAL BLOCKS
        .ENDR
        ADD       RO,R1    ;TOTAL BLOCKS OF 32.WORDS MEMORY
        BIC       #BIT0,#PS ;CLEAR CARRY
        CMP       R1,6(SP) ;IS MEMORY AVAILABLE?
        BGE       1$      ;YES
        PREAD     RO      ;SAVE CURRENT PSW
        BIS       #BIT0,RO ;SET CARRY FOR ERROR INDICATION
        PSET      RO      ;
        BR        2$      ;DONT TOUCH APR6
        MOV       6(SP),KISAR6 ;MAP TO BUFFER AREA
        POP       R1,RO    ;RESTORE REGISTERS
        MOV       (SP)+,(SP) ;CLEAN STACK OF ARGUMENT
        RETURN          ;BACK TO CALLER
:
:
:

```

```

267          .SBTTL  $.EMT
268          ;
269          ; SUBROUTINE:  $.EMT
270          ; .....
271          ;
272          ; THIS ROUTINE HANDLES ALL EMT-TRAPS THAT MIGHT OCCUR
273          ; IN A DIAGNOSTIC PROGRAM, AND DIRECTS THE PROGRAM TO
274          ; CONTINUE AT THE APPROPRIATE FUNCTION SUBROUTINES.
275          ;
276          ; INPUT:          2(SP) = PS
277          ;                  (SP) = PC
278          ; OUTPUT:         NONE
279          ;
280          ; AUTHOR:         BERT HUBER CSS/DP MUNICH 14-JULI 76
281          ;
282          ;
283          ; ENABL LSB
284 002326 012746 177776  $.EMT: MOV  4-2,-(SP)      ;REBUILD EMT-ADDR.
285 002332 066616 000002      ADD  2(SP),(SP)      ;
286 002336 017616 000000      MOV  8(SP),(SP)      ;GET EMT-CODE
287 002342 042716 177400      BIC  4177400,(SP)    ;EXTRACT EMT-NR.
288 002346 006316          ASL  (SP)              ;CONVERT TO TABLE-OFFSET
289 002350 021627 000004      CMP  (SP),4EMTTND-EMTTBL 2;VALID EMT-NR.?
290 002354 101413          BLOS  1$              ;IF LOS, YES
291 002356          WRITE MILEMT                  ;ILLEGAL EMT
292 002366          DUMP  OCT,2(SP)                ;
293 002400 000137 001000      JMP  841000          ;RESTART PROGRAM
294 002404 062716 002416  1$:  ADD  4EMTTBL,(SP)    ;GET FUCTION CODE ADDR.
295 002410 017616 000000      MOV  8(SP),(SP)      ;
296 002414 000136          JMP  8(SP)+            ;ENTER FUCTION SUBROUTINE
297 002416 002424      EMTTBL: EMT0
298 002420 002432          EMT1
299 002422 002470          EMT2
300 002424      EMTTND:
301          ;
302          ; SUBROUTINE TO SAVE SCOPE-LOOP ADDR.
303          ;
304 002424 011637 007172      EMT0:  MOV  (SP),SCOPAD    ;SAVE ADDR. OF SCOPE-LOOP
305 002430 000002          RTI                        ;RETURN TO CALLER
306          ;
307          ; SUBROUTINE TO SETUP A VECTOR
308          ;
309 002432 017646 000010      EMT1:  MOV  810(SP),-(SP) ;GET POINTER TO VECTOR ADDR.
310 002436 016676 000010      MOV  10(SP),8(SP)      ;SETUP VECTOR
311 002444 062716 000002      ADD  42,(SP)          ;POINT TO NEW PS
312 002450 016636 000006      MOV  6(SP),8(SP)+      ;SETUP NEW PS
313 002454 012666 000004      MOV  (SP)+,4(SP)      ;CLEAN UP STACK
314 002460 012666 000004      MOV  (SP)+,4(SP)      ;
315 002464 005726          TST  (SP)+              ;
316 002466 000002          RTI                        ;RETURN TO CALLER
317          ;
318          ; SUBROUTINE TO RESET A VECTOR
319          ;
320 002470 017646 000004      EMT2:  MOV  84(SP),-(SP)  ;GET POINTER TO VECTOR A
321 002474 011646          MOV  (SP),-(SP)          ;DUPLICATE IT
322 002476 062716 000002      ADD  42,(SP)          ;BUILD ADDRESS OF VECTOR
323 002502 012636          MOV  (SP)+,8(SP)+        ;RESET VECTOR

```

L2

IEC11 A DIAGNOSTIC
\$.EMT

MACRO M1200 30 MAR 84 16:02 PAGE 7 1

SEQ 24

324	002504	012776	000004	177774	MOV	#4,8-4(SP)	;RESET VECTOR +2
325	002512	016666	000002	000004	MOV	2(SP),4(SP)	;CLEAN STACK
326	002520	012616			MOV	(SP)+,(SP)	;
327	002522	000002			RTI		;RETURN TO CALLER

```

329          .SBTTL  $..TRP
330          ;*
331          ; SUBROUTINE:  $..TRP
332          ; .....
333          ;
334          ; THIS ROUTINE HANDLES ALL TRAPS TO LOC 34 THAT MIGHT
335          ; OCCUR IN A CSS-DIGNOSTIC PROGRAM.
336          ; AS THE TRAP INSTRUCTION IN THIS DIAGNOSTICS IS ONLY USED TO
337          ; INDICATE AN ERROR CONDITION, THIS TRAP-HANDLER IS A (PSEUDO-)
338          ; SWITCH-REGISTER CONTROLLED ERROR HANDLER. FOR THE DEFINITION
339          ; OF ALL AVAILABLE 'SWITCH'-OPTIONS, REFER TO SUBROUTINE "$..IAE"
340          ; LATER IN THIS LISTING.
341          ;
342          ; INPUT:          2(SP)  =      PS
343          ;                (SP)  =      PC
344          ;   LOW ORDER BYTE OF TRAP INSTRUCTION = ERROR NUMBER
345          ;
346          ; OUTPUT:        DEPENDING ON OPTIONS SELECTED
347          ; AUTHOR:        BERT HUBER, CSS/DP MUNICH  14 JUL-76
348          ; -
349          ;.PSECT CSSMON
350          ;.ENABL  LSB
351          $..TRP: MOV  0-2, -(SP)      ;REBUILD TRAP ADDRESS
352                   ADD  2(SP), (SP)    ;
353                   MOV  0(SP), (SP)    ;GET TRAP CODE
354                   BIC  0177400, (SP)  ;EXTRACT ERROR NUMBER
355                   TST  $.ER          ;FIRST ERROR OCCURENCE ?
356                   BEQ  2$            ;IF EQ, FIRST PASS
357                   TST  PSDSWR        ;SCOPE LOOP ?
358                   BPL  3$            ;
359                   CMP  (SP), $.ER    ;SAME SCOPE LOOP ?
360                   BEQ  3$            ;IF EQ, YES
361                   WRITE MILSCP       ;ILLEGAL SCOPE LOOP
362                   DUMP OCT, 2(SP)    ;DUMP ADDRESS OF ERROR +2
363                   GOTO START        ;RESTART PROGRAM
364                   MOV  (SP), $.ER    ;STORE ERROR NUMBER
365                   BIT  0BIT14, PSDSWR ;INHIBIT PRINTOUT ?
366                   BNE  4$            ;IF NE, YES
367                   WRITE MERROR      ;PRINT ERROR MESSAGE PART 1
368                   DUMP DEC, (SP)    ;DUMP ERROR NUMBER
369                   WRITE MERRP2     ;PRINT ERROR MESSAGE PART 2
370                   DUMP OCT, 2(SP)    ;DUMP ERROR PC +2
371                   BIT  0BIT13, PSDSWR ;COUNT ERRORS ?
372                   BEQ  5$            ;IF EQ, NO
373                   INC  02(SP)       ;COUNT !
374                   BNE  5$            ;IF EQ, OVERFLOW
375                   WRITE MERCOV      ;ERROR COUNTER OVERFLOW
376                   DUMP DEC, (SP)    ;DUMP ERROR NUMBER
377                   BIT  0BIT14, PSDSWR ;PRINT ? ?
378                   BNE  6$            ;IF NE, NO
379                   BIT  0BIT15, PSDSWR ;SCOPE LOOP ?
380                   BNE  6$            ;IF SCOPE, NO PRINT
381                   BIT  0BIT12, PSDSWR ;HALT AFTER ERROR SELECTED
382                   BNE  6$            ;IF NE, YES
383                   WRITE MERHLT      ;STOP MESSAGE
384                   BIT  0BIT12, PSDSWR ;HALT ?
385                   BNE  7$            ;IF EQ, YES

```

IEC11-A DIAGNOSTIC
\$..TRP

MACRO M1200 30-MAR 84 16:02 PAGE 8-1

SEQ 26

386	003004	032737	004000	007160		BIT	#BIT11.PSDSWR	;IMMEDIATE CONTINUE ?
387	003012	001770				BEQ	6\$	;IF NE,YES
388	003014	042737	004000	007160	7\$:	BIC	#BIT11.PSDSWR	;SCOPE LOOP ?
389	003022	005737	007160			TST	PSDSWR	;
390	003026	100005				BPL	8\$	;IF PL, NO
391	003030					STCKUP		
392	003034	005726				TST	(SP)+	;ADJUST STACKPOINTER
393	003036	000177	004130			JMP	@SCOPAD	;ENTER SCOPE LOOP
394	003042	005037	007166		8\$:	CLR	\$..ER	;RESET ERROR NUMBER
395	003046	005726				TST	(SP)+	;READJUST STACK
396	003050	062716	000002			ADD	#2,(SP)	;BUILD RETURN ADDRESS
397	003054	000002				RTI		;RETURN TO CALLER

```

399          .SBTTL $..IOT
400
401      ; SUBROUTINE: $..IOT
402      ; .....
403
404      ; THIS SUBROUTINE HANDLES ILLEGAL TRAPS OR INTERRUPTS.
405
406      ; IF SUCH A TRAP OR INTERRUPT OCCURS, A MESSAGE IS
407      ; PRINTED, SHOWING THE VECTOR TRAPPED TO, AND THE PROGRAMS PC
408      ; WHERE IT WAS INTERRUPTED.
409      ; THE PROGRAM HALTS THEN, AND MUST BE RESTARTED.
410
411      ; INPUT:      6(SP)  - PS OF PROGRAM
412      ;             4(SP)  - PC OF PROGRAM
413      ;             2(SP)  - PS OF VECTOR
414      ;             (SP)  - PC OF VECTOR *2
415
416      ; OUTPUT:     MESSAGE TO THE OPERATOR
417
418      ; AUTHOR:     BERT HUBER, CSS/DP MUNICH 22 JUL-76
419
420      ; .PSECT CSSMON
421      ; .ENABL LSB
422      $..IOT: WRITE MILTR1      ; SEND FIRST MESSAGE PART
423      SUB #2,(SP)      ; COMPUTE VECTOR
424      DUM OCT,(SP)      ; DUMP VECTOR ADDRESS
425      WRI MILTR2      ; SEND SECOND MESSAGE PART
426      DUMP OCT,4(SP)    ; DUMP PROGRAM PC
427      BR      ; HANG UP SYSTEM
428      .DSABL LSB

```

162716 000002

000777

```
430          .SBTTL  $.PWR
431
432      ; SUBROUTINE:  $.PWR
433      ; .....
434
435      ; THIS ROUTINE IS DIRECTLY ENTERED AFTER EACH POWER-FAIL TO
436      ; HANDLE THE NECESSARY REGISTER SAVE OPERATIONS, AND IS USED
437      ; AFTER THE POWER-UP INTERRUPT TO RESTORE THE SAVED REGISTERS
438      ; AFTER A SUCCESSFUL POWER-FAIL RESTART, A SHORT MESSAGE IS PRINTED.
439
440      ; AUTHOR:      BERT HUBER, CSS/DP MUNICH 27-JULY-76
441      ;             MZ001  SAVE THE OUTFLAG IN 3: AND RESTORE IT
442      ;             ..... AFTER PRINTOUT OF THE POWERFAIL MESSAGE
443      ;
444      ; .PSECT CSSMON
445      ; .ENABL  LSB
446      $.PWR: PUSH  R0,R1,R2,R3,R4,R5
447      MOV      SP,(PC).      ;SAVE STACK-POINTER
448      .WORD    0              ;STACKPOINTER-SAVE-AREA
449      MOV      @2:,@24        ;SETUP VECTOR FOR RESTART
450      MOV      OUTFLG,(PC).   ;SAVE PRINT FLAG      ;MZ001
451      .WORD    0              ;SAVE FOR FLAG      ;MZ001
452      MOV      INFLAG,(PC).   ;SAVE INPUT FLAG
453      .WORD    0              ;SAVE WORD FOR INPUT FLAG
454      BR       .              ;WAIT FOR POWER DOWN
455      CLR      R2              ;DELAY
456      INC      R2              ;THE
457      CMP      R2,@177777     ;POWER UP START
458      BNE      .-6            ;FOR 1SECOND
459      MOV      1:SP           ;REBUILD STACKPOINTER
460      CLR      OUTFLG         ;ALLOW PRINTOUT OF PWRFL MES.      ;MZ001
461      CLR      INFLAG         ;
462      MOV      @1:$.PWR,@24   ;REBUILD VECTOR SETUP
463      POP      R5,R4,R3,R2,R1,R0
464      MOV      1:$.MSP,(PC).  ;SAVE OLD MESSAGE POINTER
465      .WORD    0              ;SAVE WORD FOR OLD MESSAGE POINTER
466      WRITE    MPWRFL         ;SEND POWER-FAIL-MESSAGE      ;MZ001
467      TST      OUTFLG         ;WAIT FOR OUTPUT COMPLETE      ;MZ001
468      BNE      4:            ;      ;MZ001
469      SET6     @0TKS          ;ENABLE FURTHER TTY-INTERRUPTS
470      MOV      3:OUTFLG      ;RESTORE FLAG IF IT WAS SET      ;MZ001
471      TST      OUTFLG         ;IF AN OUTPUT BEFORE PWRFL WAS RUNNING
472      BEQ      7:            ;
473      MOV      8:$.MSP       ;THEN RESTORE OLD MESSAGE POINTER
474      SET6     @0TPS          ;AND ENABLE INT IN TPS
475      MOVB     @12,@0TPB     ;AND START INTERRUPT
476      TST      5:            ;IF AN INP FLG BEFORE PWRFL WAS NOT SET
477      BEQ      6:            ;GOTO 6:
478      WRITE    PROMES         ;ELSE WRITE >
479      MOV      5:INFLAG      ;RESTORE INPUT FLAG IF IT WAS SET
480      RTI              ;CONTINUE WITH PROGRAM
481      .DSABL  LSB
482      .SBTTL  $.RSV
483
484      ; SUBROUTINE:  $.RSV
485      ; .....
486
```

```

487 ; THIS ROUTINE SAVES R0 R5 ONTO THE STACK
488 ;
489 ;
490 ; INPUT:      (SP)      -      CALLER PC
491 ;
492 ; OUTPUT:      (SP)      -      R5 CONTENT
493 ;              2(SP)      -      R4 CONTENT
494 ;              4(SP)      -      R3 CONTENT
495 ;              6(SP)      -      R2 CONTENT
496 ;              10(SP)     -      R1 CONTENT
497 ;              12(SP)     -      R0 CONTENT
498 ;              14(SP)     -      CALLER PC
499 ;
500 ; AUTHOR:      BERT HUBER, CSS/DP MUNICH 12 APR 76
501 ;
502 ; .PSECT CSSMON
503 ; .ENABL LSB
504 003360 $.RSV: PUSH R0,R1,R2,R3,R4,R5
505 003374 000176 000014 JMP @14(SP) ;RETURN TO CALLER
506 ; .DSABL LSB

```

```

508          .SBTTL  $.RRS
509          ;
510          ; SUBROUTINE:  $.RRS
511          ; .....
512          ;
513          ; THIS SUBROUTINE RESTORES R5 R0 FROM THE STACK
514          ;
515          ; INPUT:          (SP)  = CALLER PC
516          ;                2(SP)  = SAVED R5
517          ;                4(SP)  = SAVED R4
518          ;                6(SP)  = SAVED R3
519          ;                10(SP) = SAVED R2
520          ;                12(SP) = SAVED R1
521          ;                14(SP) = SAVED R0
522          ;                16(SP) = OLD CALLER PC
523          ;
524          ; OUTPUT:        REGISTERS ARE RESTORED AND STACK IS CLEANED
525          ;
526          ; AUTHOR:        BERT HUBER, CSS/DP MUNICH  12 APR-76
527          ;
528          ; .PSECT CSSMON
529          ; .ENABL LSR
530 003400 005726  $.RRS: TST (SP), ;POINT TO SAVED R5
531 003402      POP  R5,R4,R3,R2,R1,R0
532 003416 005726      TST (SP), ;SKIP OLD CALLER PC
533 003420 000176 177760  JMP @ 20(SP) ;RETURN TO CALLER
534          .DSABL LSR

```

```

536          .SBTTL $..KBI
537
538          ; SUBROUTINE: $..KBI
539          ; .....
540
541          ; THIS ROUTINE IS ENTERED IMMEDIATELY AFTER AN INTERRUPT
542          ; FROM TTY-KEYBOARD. ALL REGISTERS ARE SAVED ACROSS CALL.
543
544          ; INPUT:      2(SP)= PS
545          ;              (SP) = PC
546
547          ; OUTPUT:     NONE
548
549          ; AUTHOR:     BERT HUBER, CSS/DP MUNICH 5-APR 76
550          ; -
551          ;.PSECT CSSMON
552          ;.ENABL LSB
553 003424 005737 007152 $..KBI: TST OUTFLG          ;;;OUTPUT RUNNING ?
554 003430 001422          BEQ 4$          ;;;IF NE, IGNORE INTERRUPT
555 003432          PUSH RO          ;;;SAVE
556 003434 005037 177560 CLR 00TKS          ;;;INHIBIT INTERRUPT
557 003440 013700 177562 MOV 00TKB,RO          ;;;GET CHARACTER
558 003444 012737 000100 177560 MOV 0100,00TKS          ;;;RE-ENABLE
559 003452 042700 000200 BIC 0200,RO          ;;;MASK OUT ASCII
560 003456 022700 000003 CMP 03,RO          ;;; IS IT <CNTRL C> ?
561 003462 001003 BNE 13$          ;;;IF NOT, RETURN
562 003464 012737 007777 007170 MOV 07777,SPEFLG          ;;;SET A SPECIAL FLAG
563          ;;;THIS FLAG IS USED TO FINISH
564          ;;;A RUNNING PRINTOUT AND THEN TO
565          ;;;REMEMBER THAT <CNTRL C> WAS HIT
566
567 003472          13$: POP RO          ;;;REBUILD
568 003474 000436 BR 3$          ;;; RETURN TO CALLER
569 003476 004737 003360 4$: CALL $..RSV          ;;;SAVE ALL REGISTERS
570 003502 013700 177562 MOV 00TKB,RO          ;;;GET INPUT BYTE
571 003506 042700 000200 BIC 0200,RO          ;;;BUILD REAL ASCII
572 003512 122700 000003 CMPB 03,RO          ;;;<CNTRL C> FUNCTION ?
573 003516 001003 BNE 1$          ;;;IF NE, NO
574 003520          ;;SPE:
575 003520 004737 006136 CALL $..PRO          ;;;SEND PROMPT STRING
576 003524 000420 BR 2$          ;;;RETURN
577 003526 005737 007150 1$: TST INFLAG          ;;;INPUT FLAG SET ?
578 003532 001415 BEQ 2$          ;;;IF EQ, NO
579 003534 004737 003574 5$: CALL $..INP          ;;;PROCESS INPUT
580 003540 020027 000015 CMP RO,015          ;;;WAS IT A <CR> ?
581 003544 001010 BNE 2$          ;;;IF NE, NO
582 003546 005037 007150 CLR INFLAG          ;;;RESET INPUT MODE FLAG
583 003552 012701 007026 MOV 0KBBUFF,R1          ;;;REBUILD BUFFER POINTER
584 003556 010137 007146 MOV R1,KBBPNT          ;;;
585 003562 004737 004010 CALL $..IAY          ;;;ANALYZE INPUT
586 003566 004737 003400 2$: CALL $..RRS          ;;;RESTORE REGISTERS
587 003572 000002 3$: RTI          ;;;RETURN TO CALLER
588          .DSABL LSB

```

```

590          .SBTTL $..INP
591          ;
592          ;SUBROUTINE: $..INP
593          ;.....
594          ;
595          ;THIS SUBROUTINE IS THE INPUT HANDLER FOR ALL TERMINALS
596          ;USED IN ALL CSS-MUNICH DIAGNOSTIC SOFTWARE GENERATED NOW
597          ;AND LATER.
598          ;THE SPECIAL FUCTIONS HANDLED BY THIS ROUTINE ARE:
599          ;    <CNTRL U> FOR LINE RUBOUT
600          ;    <DELETE> FOR CHARACTER RUBOUT
601          ;ALL OTHER INPUT IS HANDLED AS NORMAL ASCII-TEXT AND STORED
602          ;IN THE KEYBOARD INPUT BUFFER.
603          ;
604          ;INPUT:      (SP)      -    CALLER PC
605          ;              RO      -    CURRENT INPUT CHARACTER
606          ;
607          ;OUTPUT:     NONE
608          ;
609          ;AUTHOR:     BERT HUBER CSS/DP MUNICH
610          ;
611          ;.IDENT /V1.0/
612          ;.PSECT CSSMON
613          ;.ENABL LSB
614 003574 013701 007146 $..INP: MOV R1,KBPPNT, R1          ;;;GET CURRENT BUFFER POINTER
615 003600 012702 006122      MOV R2,$..KBO, R2          ;;;POINT TO KEYBOARD-ECHO
616 003604 022701 007146      CMP R1,KBEND, R1          ;;;INPUT-BUFFER FULL?
617 003610 001015          BNE 1$          ;;;IF NE, NO
618 003612 022700 000025      CMP R2,25, R0          ;;;<CNTRL U>?
619 003616 001415          BEQ 2$          ;;;IF EQ, YES.
620 003620 122700 000177      CMPB R1,177, R0          ;;;<DELETE>?
621 003624 001407          BEQ 1$          ;;;IF EQ, YES.
622 003626 005037 007150      CLR INFLAG          ;;;RESET INPUTFLAG
623 003632          WRITE MKBOVF          ;;;PRINT OVERFLOW MESS.
624 003642 000451          BR 7$          ;;;PROMPT FOR INPUT
625 003644 122700 000025 1$: CMPB R2,25, R0          ;;;<CNTRL U>?
626 003650 001436          BEQ 6$          ;;;IF NE, NO
627 003652 122700 000177 2$: CMPB R1,177, R0          ;;;<DELETE>?
628 003656 001016          BNE 4$          ;;;IF NE, NO
629 003660 005737 007162      TST RUBFLG          ;;;FIRST RUBOUT?
630 003664 001005          BNE 3$          ;;;IF NE, NO
631 003666 010637 007162      MOV SP,RUBFLG          ;;;FLAG RUBOUT MODE
632 003672 012700 000057      MOV R1,'/, R0          ;;;PRINT "/"
633 003676 004712          CALL @R2          ;;;
634 003700 020127 007026 3$: CMP R1,KBUFF          ;;;FULL LINE DELETED?
635 003704 001434          BEQ INP..E          ;;;IF EQ, YES
636 003706 114100          MOVB (R1),R0          ;;;NO, REMOVE CHARACTER
637 003710 004712          CALL @R2          ;;;AND ECHO IT
638 003712 000431          BR INP..E          ;;;WAIT FOR NEXT INP 'I
639 003714 005737 007162 4$: TST RUBFLG          ;;;RUBOUT MODE?
640 003720 001407          BEQ 5$          ;;;IF EQ, NO
641 003722          PUSH R0          ;;;YES, SAVE CURRENT INPUT
642 003724 012700 000057      MOV R1,'/, R0          ;;;NO OF RUBOUT
643 003730 004712          CALL @R2          ;;;ECHO "/"
644 003732 005037 007162      CLR RUBFLG          ;;;RESET RUBOUT MODE FLAG
645 003736          POP R1          ;;;GET CURRENT INPUT AGAIN
646 003740 004712 5$: CALL @R2          ;;;ECHO IT

```

647	003742	110021		MOVB	R0,(R1)+	;;;SAVE IN INPUT BUFFER
648	003744	000414		BR	INP..E	;;;WAIT FOR NEXT INPUT
649	003746	012700	000136	6\$:	MOV	#136,R0
650	003752	004712			CALL	BR2
651	003754	012700	000125		MOV	#'U,R0
652	003760	004712			CALL	BR2
653	003762	005037	007162		CLR	RUBFLG
654	003766	012701	007026	7\$:	MOV	#KBBUFF,R1
655	003772	004737	006136		CALL	\$. .PRO
656	003776	010137	007146	INP..E:	MOV	R1,KBBPNT
657	004002	116100	177777		MOVB	-1(R1),R0
658	004006	000207			RETURN	
659					.DSABL	LSB

```

;;;SAVE IN INPUT BUFFER
;;;WAIT FOR NEXT INPUT
;;;ECHO "+"
;;;
;;;ECHO "U"
;;;
;;;LEAVE RUBOUT MODE
;;;RESET INPUT BUFFER POINTER
;;;PROMPT FOR INPUT
;;;SAVE POINTER
;;;SAVE LAST CHARACTER
;;;RETURN TO CALLER

```

```

661          .SBTTL $..IAY
662          ;*
663          ;SUBROUTINE: $..IAY
664          ;*****
665          ;
666          ;LEGAL INPUT TYPES ARE:
667          ;
668          ;      ERR=          FOR ERROR-HANDLING
669          ;      DMP=          FOR CORE-DUMP OPTIONS
670          ;      SET=          FOR DEBUGGING OPTIONS
671          ;      NONFORMATTED INPUT FOR SPECIAL REQ.
672          ;
673          ;INPUT: (SP)=      RETURN PC
674          ;OUTPUT:          NONE
675          ;
676          ;AUTHOR:          BERT HUBER, CSS/DP MINICH      28-JUN 76
677          ;
678          ;.PSEC; CSSMON
679          ;ENABL  LSB
680 004010 122711 000015 $..IAY: CMPB 0'15,(R1)      ;;;<CR>?
681 004014 001440      BEQ  IAY..E      ;;;IF EQ., SKIP ANALIZATION
682 004016 005737 007174 3$:  TST  $.$ASF      ;;;DONT ANALYZE IF ASCII INPUT RUNNING
683 004022 001035      BNE  IAY..E      ;;;IF NE, YES.
684 004024 122711 000104      CMPB 0'D,(R1)      ;;;'DMP" REQUESTED?
685 004030 001003      BNE  1$      ;;;IF NE, NO
686 004032 004737 004210      CALL $..IAD      ;;;ANALYZE DUMP REQUEST
687 004036 000427      BR  IAY..E      ;;;RETURN TO CALLER
688 004040 122711 000123 1$:  CMPB 0'S,0R1      ;;;'SET" REQUESTED?
689 004044 001003      BNE  2$      ;;;IF NE, NO
690 004046 004737 004402      CALL $..IAS      ;;;ANALYZE SET REQUEST
691 004052 000421      BR  IAY..E      ;;;RETURN TO CALLER
692 004054 122711 000105 2$:  CMPB 0'E,0R1      ;;;"ERR"-REQUEST?
693 004060 001003      BNE  4$      ;;;IF NE, NO
694 004062 004737 004472      CALL $..IAE      ;;;ANALYZE ERROR REQUEST
695 004066 000413      BR  IAY..E      ;;;RETURN TO CALLER
696 004070 122711 000101 4$:  CMPB 0'A,0R1      ;;;"ABO" REQUESTED ?
697 004074 001003      BNE  5$      ;;;IF NE, NO
698 004076 004737 004120      CALL $..IAA      ;;;ANALYZE ABO-REQUEST
699 004102 000405      BR  IAY..E      ;;;RETURN TO CALLER
700 004104 005737 007156 5$:  TST  INPREQ      ;;;NO SYNTAX ERROR IF SPECIAL INPUT RUNNING
701 004110 001002      BNE  IAY..E      ;;;
702 004112 004737 006262      CALL $..STX      ;;;SYNTAX ERROR
703 004116 000207      IAY..E: RETURN      ;;;RETURN TO CALLER
704          .DSABL  LSB

```

```

706      ;
707      ;
708      ;
709      ;
710      ;
711      ;
712      ;      .SBTTL  $..IAA
713      ;
714      ;SUBROUTINE:  $..IAA
715      ;.....
716      ;
717      ;LEGAL INPUT IS "ABO"
718      ;
719      ;      INPUT:  NONE
720      ;      OUTPUT: IF IN THE PROGRAM THE LABEL ""$..SPC"" IS USED,
721      ;               THE STACK WILL BE INITIALIZED AND A JUMP TO
722      ;               THIS LABEL IN THE DEVICE TEST PROGRAM IS PERFORMED.
723      ;               IF THIS LABEL IS NOT USED, THE STACK IS INITIALIZED
724      ;               AND A JUMP TO THE BEGIN OF THE DEVICE TEST PROGRAM
725      ;               IS PERFORMED.
726      ;
727      ;      MZ001
728      ;      .....
729      ;      THE MACRO "PSET"ALLOWS TO RUN THE COM EITHER ON PDP
730      ;      OR ON LSI.
731      ;
732      ;AUTHOR:      M. ZILLER,      CSS/DP MUNICH  OCT. 1976
733      ;
734      ;
735      ;      .PSECT CSSMON
736      ;      .ENABL  LSB
737      $..IAA: RESET      ;STOP ANY ACTIONS
738      SET6      @TKS      ;SWITCH ON TTY INT.
739      MOV      @START,SP  ;INITIALIZE STACK
740      .IF DF $..SPC
741      MOV      @$.SPC,-(SP) ;SET THE LABEL ""$..SPC"" FOR DESTINATION
742      .ENDC
743      .IF NDF $..SPC
744      MOV      @COM..E.(SP) ;SET END OF MONITOR FOR DESTINATION
745      .ENDC
746      PSET      @0      ;CLEAR PS      ;MZ001
747      MOV      @KBBUFF,RO ;RESET BUFFERS
748      CLR      (RO).
749      CMP      RO,@PROMES
750      BNE      1$
751      MOV      @KBBUFF,KBBPNT ;REBUILD BEGIN ADDRESS
752      RETURN    ;GOTO POINT OF CONTINUATION
      .DSABL  LSB

```

```

754      ;.PSECT CSSMON
755      ;
756      ;
757      ;
758      ;
759      .SBTTL $.IAD
760      ;
761      SUBROUTINE      :      $.IAD
762      *****      *****
763      ;
764      ;      MZ001
765      ;      *****
766      ;      MODIFICATION TO ALLOW THE <CR> AFTER INPUT
767      ;      TO COMPLETE.
768      ;
769      ;
770      ;      THIS SUBROUTINE IS ENTERED WHEN A DUMP REQUEST IS ANALYZED.
771      ;      DEPENDING ON HOW THE REQUEST IS TERMINATED, THERE ARE TWO
772      ;      WAYS FOR RETURN:
773      ;
774      ;      A)      WHEN TERMINATED BY <CR> THE CONTENT OF A LOCATION IS
775      ;      PRINTED AND THE RETURN IS PERFORMED
776      ;      B)      WHEN INPUT IS TERMINATED BY A ""/""THE MONITOR WAITS
777      ;      FOR INPUT AND LOADS THIS INPUT INTO THE EXAMINED LOCATION.
778      ;
779      ;INPUT      :      (SP)      *RETURN ADDRESS
780      ;OUTPUT      :      MODIFIED LOCATION IF REQUESTED
781      ;
782      ;AUTHOR:      M. ZILLER, SS/DP MUNICH      OCTOBER 1976
783      ;
784      ;-
785      ;.PSECT CSSMON
786      .ENABL      LSB
787      $.IAD:      INC      R1      ;POINT TO "M"
788      004210      005201      000115      CMPB      (R1),0'M      ;IS IT A 'M'
789      004216      001027      BNE      77$      ;NO IF NE
790      004220      122127      000120      CMPB      (R1),0'P      ;IS IT A 'P'
791      004224      001024      BNE      77$      ;NO IF NE
792      004226      122127      000057      CMPB      (R1),0'/'      ;IS IT A '/'
793      004232      001021      BNE      77$      ;NO IF NE
794      004234      004737      004304      CALL      $.NOR      ;RETRIEFE INPUT
795      004240      103416      BCS      77$      ;IS CS. INPUT FORMAT ERROR
796      004242      105737      177564      11$:      TSTB      @0TPS      ;IS THE <CR> FINISHED ? ;MZ001
797      004246      100375      BPL      11$      ;      ;MZ001
798      004250      012737      000012      177566      MOV      @12,@0TPB      ;PRINT A <LF
799      004256      105737      177564      12$:      TSTB      @0TPS      ;DONE ?
800      004262      100375      BPL      12$      ;WAIT FOR COMLETION
801      004264      [JMP      OCT,(RO)      ;AND PRINT
802      004274      000402      BR      IAD..E      ;GO TO END
803      004276      004737      006262      77$:      CALL      $.STX      ;WRITE SYNTAX ERROR MESSAGE
804      004302      000207      IAD..E:      RETURN      ;RETURN TO MAIN
805      ;

```

```

807
808
809
810
811 004304 005000
812 004306 012702 000006
813 004312 121127 000015
814 004316 001425
815 004320 121127 000057
816 004324 001422
817 004326 142711 000060
818 004332 100421
819 004334 121127 000007
820 004340 101016
821 004342 006300
822 004344 006300
823 004346 006300
824 004350 152100
825 004352 005302
826 004354 001356
827 004356 121127 000015
828 004362 001403
829 004364 121127 000057
830 004370 001002
831 004372 000241
832 004374 000207
833 004376 000261
834 004400 000207
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849 004402 005201
850 004404 122127 000105
851 004410 001024
852 004412 122127 000124
853 004416 001021
854 004420 122127 000057
855 004424 001016
856 004426 004737 004304
857 004432 103413
858 004434 010037 004470
859 004440 122127 000057
860 004444 001006
861 004446 004737 004304
862 004452 103403
863 004454 010077 000010

;
; THIS IS THE ROUTINE GET THE OCTAL VALUE OF THE ADDRESS
;
;..NOR: CLR R0 ;CLEAR STORAGE OF NUMBER
; MOV #6,R2 ;CLEAR DIGIT COUNTER
1$: CMPB @R1,@15 ;IS IT A TERMINATOR ?
; BEQ NOR..E ;IF EQ, YES
; CMPB @R1,@' / ;
; BEQ NOR..E ;
; BICB #60,@R1 ;BUILD REAL BINARY
; BMI NOR..ER ;IF MI, IT WAS LOWER THAN 60
; CMPB @R1,@7 ;IS IT REAL OCTAL ?
; BMI NOR..ER ;IF HI,NO REAL OCTAL
; ASL R0 ;MULTIPLY
; ASL R0 ;WITH 10 IN
; ASL R0 ;OCTAL
; BISB (R1),R0 ;SET THE TRUE BITS
; DEC R2 ;COUNT DIGITS BUILT
; BNE 1$ ;ALL DONE?
; CMPB @R1,@15 ;IS THE LAST + 1 BYTE A TERMINATOR?
; BEQ NOR..E ;IF YES THEN SYNTAX OK
; CMPB @R1,@' / ;TERMINATOR IN SET COMMAND?
; BNE NOR..ER ;IF NOT THEN SYNTAX ERROR
; CLC ;INDICATE SUCCESSFUL OPERATION
; RETURN ;
; NOR..ER: SEC ;INDICATE FAILURE
; RETURN ;
;
; SUBROUTINE: $..IAS
; ..IAS:
;
; THIS SUBROUTINE IS ENTERED IF THE INPUT ANALYZER DETECTS A SET-COMMAND
;
; INPUT: (SP) = RETURN ADDRESS
; R1 = BEGIN OF INPUT STRING
;
; OUTPUT: = MODIFIED LOCATION
;
;..IAS: INC R1 ;POINT TO "E"
; CMPB (R1),@'E ;CORRECT
; BNE 76$ ;NO IF NE
; CMPB (R1),@' I ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CMPB (R1),@' / ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CALL $..NOR ;BUILD ADDRESS TO MODIFY
; BCS 76$ ;IF CS, FORMAT ERROR
; MOV R0,2$ ;SAVE
; CMPB (R1),@' / ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CALL $..NOR ;BUILD DATA TO INSERT
; BCS 76$ ;IF CS, FORMAT ERROR
; MOV R0,@2$ ;AND LOAD

```

M3

IEC11 A DIAGNOSTIC
\$..IAD

MACRO M1200 30 MAR 84 16:02 PAGE 1 / 1

SEQ 38

864 004460 000402
865 004462 004737 006262
866 004466 000207
867 004470 000000
868

76\$: BR IAS..E
CALL \$..STX
IAS..E: RETURN
2\$: .WORD 0
.DSABL LSB

.GO TO END
;WRITE ERROR MESSAGE
;RETURN
;SORAGE FOR ADDRESS TO MODIFY

```

870      ;
871      ;
872      ;
873      .SBTTL $.IAE
874      ;
875      ;SUBROUTINE: $.IAE
876      ;*****
877      ;
878      ;THIS ROUTINE CHECKS THE CORRECT SYNTAX OF
879      ;AN"ERR"- INPUT STRING AND PROCESSES THE
880      ;REQUESTED 'SWITCHREGISTER OPTIONS'.
881      ;
882      ;LEGAL SWITCH OPTIONS ARE:
883      ;
884      ;      /HT      *      HALT AFTER ERROR (DEFAULT)
885      ;      /NH      *      NO HALT AFTER ERROR
886      ;      /PR      *      PRINT ERROR MESSAGE (DEFAULT)
887      ;      /NP      *      INHIBIT ERROR PRINTOUT
888      ;      /CT      *      COUNT ERROR
889      ;      /NC      *      NO ERROR COUNTING (DEFAULT)
890      ;      /SC      *      SCOPE LOOP ON THIS ERROR
891      ;      /NS      *      NO SCOPE LOOP (DEFAULT)
892      ;      /CONT    *      IMMEDIATE CONTINUE SWITCH
893      ;
894      ;
895      ;INPUT:      R1      *START ADDR.OF TEXT STRING
896      ;OUTPUT:     BIT-SETUP IN SOFTWARE SWITCH REGISTER
897      ;
898      ;AUTHOR:     BERT HUBER, CSS/DP MUNICH 28-JULY 76
899      ;
900      ;.PSECT CSSMON
901      .ENABL LSB
902 004472 010602      $.IAE: MOV SP,R2      ;;;FLAG FOR VERY FIRST CHARACTER
903 004474 012703 007160 MOV #PSDSWR,R3 ;;;GET "SWR" ADDRESS
904 004500 005004      CLR R4      ;;;USE AS CHARACTER 1
905 004502 062701 000003 ADD #3,R1 ;;;SKIP IDENTIFIER
906 004506 122127 000075 CMPB (R1),#0 ;;;CORRECT SYNTAX?
907 004512 001137      BNE 77$ ;;;BRANCH IF NOT
908 004514 122127 000057 1$: CMPB (R1),#0 / ;;;IS NEXT CHAR. A '/'
909 004520 001403      BEQ 2$ ;;;IF EQ, YES
910 004522 005702      TST R2 ;;;VERY FIRST?
911 004524 001132      BNE 77$ ;;;IF YES: SYNTAX ERROR
912 004526 000526      BR 11$ ;;;ASSUME END OF STRING
913 004530 005002      2$: CLR R2 ;;;NO LONGER VERY FIRST
914 004532 005204      INC R4 ;;;SET FOR 'C1' FLAG
915 004534 121127 000116 CMPB (R1),#N ;;;IS IT THE /N OPTION?
916 004540 001002      BNE 3$ ;;;BRANCH IF NOT
917 004542 005201      INC R1 ;;;ADVANCE TO NEXT CHARACTER
918 004544 005004      CLR R4 ;;;RESET 'C1' FLAG
919 004546 121127 000123 3$: CMPB (R1),#S ;;;IS IT A 'S'?
920 004552 001426      BEQ 5$ ;;;BRANCH IF YES
921 004554 121127 000103 CMPB (R1),#C ;;;IS IT A 'C'?
922 004560 001437      BEQ 7$ ;;;BRANCH IF YES
923 004562 121127 000110 CMPB (R1),#H ;;;IS IT A 'H'?
924 004566 001472      BEQ 9$ ;;;BRANCH IF YES
925 004570 121127 000120 CMPB (R1),#P ;;;IS IT A 'P'?
926 004574 001106      BNE 77$ ;;;IF NO: SYNTAX ERROR

```

927	004576	005704				TST	R4	IS IT A 'P' ?
928	004600	001004				BNE	48	BRANCH IF YES
929	004602	052713	040000			BIS	#BIT14,(R3)	SET SR-BIT FOR /NP
930	004606	005201		308:		INC	R1	ADVANCE TO NEXT CHARACTER
931	004610	000741				BR	18	CHECK FOR NEXT SWITCH
932	004612	005201		48:		INC	R1	NEXT CHARACTER
933	004614	122127	000122			CMPB	(R1),#0,R	IS IT A '/PR' ?
934	004620	001074				BNE	778	IF NO: SYNTAX ERROR
935	004622	042713	040000			BIC	#BIT14,(R3)	SET '/PR' OPTION
936	004626	000732				BR	18	CHECK NEXT SWITCH
937	004630	005704		58:		TST	R4	IS IT 'S' ?
938	004632	001003				BNE	68	BRANCH IF YES
939	004634	042713	100000			BIC	#BIT15,(R3)	SETUP FOR '/NS' OPTION
940	004640	000762				BR	308	TRY NEXT
941	004642	005201		68:		INC	R1	NEXT CHARACTER
942	004644	122127	000103			CMPB	(R1),#0,C	IS IT A '/SC' ?
943	004650	001060				BNE	778	IF NO: SYNTAX ERROR
944	004652	052713	100000			BIS	#BIT15,(R3)	SETUP FOR '/SC' OPTION
945	004656	000716				BR	18	CHECK NEXT SWITCH
946	004660	126127	000001	000117	78:	CMPB	1(R1),#0	CHECK FOR '/CONT
947	004666	001016				BNE	708	IF NE , OTHER
948	004670	126127	000002	000116		CMPB	2(R1),#0,N	CHECK FOR '/CONT'
949	004676	001045				BNE	778	IF NO: SYNTAX ERROR
950	004700	126127	000003	000124		CMPB	3(R1),#0,T	CHECK FOR '/CONT'
951	004706	001041				BNE	778	IF NO: SYNTAX ERROR
952	004710	052737	004000	007160		BIS	#BIT11,PSDSWR	SET IMMEDIATE CONTINUE
953	004716	062701	000005			ADD	#5,R1	ADJUST POINTER
954	004722	000430				BR	118	END THIS SCAN
955	004724	005704			708:	TST	R4	IS IT A '/CT' ?
956	004726	001003				BNE	88	BRANCH IF YES
957	004730	042713	020000			BIC	#BIT13,(R3)	SETUP FOR '/NC' OPTION
958	004734	000724				BR	308	TRY NEXT SWITCH
959	004736	005201		88:		INC	R1	ADVANCE TO NEXT CHARACTER
960	004740	122127	000124			CMPB	(R1),#0,T	IS IT A '/CT' OPTION ?
961	004744	001022				BNE	778	IF NO: SYNTAX ERROR
962	004746	052713	020000			BIS	#BIT13,(R3)	SETUP FOR '/CT' OPTION
963	004752	000660				BR	18	CHECK NEXT SWITCH
964	004754	005704			98:	TST	R4	IS IT '/HT' ?
965	004756	001003				BNE	108	BRANCH IF YES
966	004760	052713	010000			BIS	#BIT12,(R3)	SETUP FOR '/NP' OPTION
967	004764	000710				BR	308	TRY NEXT SWITCH
968	004766	005201		108:		INC	R1	ADVANCE TO NEXT CHARACTER
969	004770	122127	000124			CMPB	(R1),#0,T	IS IT A '/HT' ?
970	004774	001006				BNE	778	IF NO: SYNTAX ERROR
971	004776	042713	010000			BIC	#BIT12,(R3)	SETUP FOR '/HT' OPTION
972	005002	000644				BR	18	TRY NEXT SWITCH
973	005004	124127	000015	118:		CMPB	(R1),#15	WAS LAST BYTE A <CR>?
974	005010	001402				BEQ	IAE..E	BRANCH IF YES
975	005012	004737	006262	778:		CALL	8..STX	SYNTAX ERROR
976	005016	000207			IAF..E:	RETURN		RETURN TO CALLER
977						.DSABL	L58	

```

979          .SBTTL  $.RED
980
981      ; SUBROUTINE  $.RED
982      ; .....
983
984
985      ; **** AB00
986      ;         THE CARRY BIT IS SET IF A ZERO OCTAL INPUT (ONLY CR,NO DATA)
987      ;         HAPPENS
988
989
990      ; THIS SUBROUTINE HANDLES ALL TYPES OF SPECIAL INPUT REQUESTS
991      ; SETUP BY THE PROGRAMMER THROUGH A ' READ ' MACRO.
992      ; LEGAL INPUT TYPES THAT CAN BE HANDLED ARE:
993
994      ; BINARY FORMAT (EX: 1001101101010001)
995      ; .....
996      ; INPUT CAN BE UP TO 16-DIGITS OF THE FORM 0 OR 1.
997      ; LESS THAN 16 DIGIT INPUT IS FILLED WITH LEADING ZEROES
998      ; INPUT IS RETURNED IN RO.
999      ; THE PREVIOUS CONTENT OF RO WILL BE OVERWRITTEN.
1000
1001      ; OCTAL FORMAT (EX:176523)
1002      ; .....
1003      ; INPUT CAN BE UP TO 6 DIGITS OF THE FORM 0-7.
1004      ; IN CASE OF A 6-DIGIT-WIDE INPUT,THE LEFTMOST(HIGH ORDER)
1005      ; DIGIT IS TRUCATED TO THE RIGHTMOST (LOW ORDER)BIT.
1006      ; IN CASE OF LESS THAN 6 DIGITS INPUT, LEADING ZERO DIGITS
1007      ; ARE INSERTED.
1008      ; THE RESULT IS RETURNED IN RO,HEREBY OVERWRITING
1009      ; THE PREVIOUS CONTENT OF RO.
1010
1011      ; DECIMAL FORMAT (EX: 96003)
1012      ; .....
1013      ; INPUT CAN BE UP TO 5 DIGITS OF THE FORM 0 9.
1014      ; LESS THEN FIVE DIGITS INPUT ARE FILLED UP WITH
1015      ; LEADING ZEROE DIGITS.
1016      ; MAXIMUM NUMBER ALLOWED IS 32656.
1017      ; INPUT IS RETURNED IN RO, HEREBY OVERWRITING THE
1018      ; PREVIOUS CONTENT OF RO.
1019
1020      ; ASCI FORMAT (EX: B6T#EL)
1021      ; .....
1022
1023      ; INPUT CAN BE ANY LEGAL ASCII CHARACTER.
1024      ; IF INPUT INTO A BUFFER IS REQUESTED, UP TO 80
1025      ; CHARACTERS CAN BE INPUT.
1026      ; IF INPUT RETURN IN RO IS REQUESTED,ONLY THE FIRST
1027      ; CHARACTER WILL BE RETURNED IN THE LOW ORDER BYTE
1028      ; OF RO.
1029      ; OLD CONTENT OF RO IS OVERWRITTEN IN THIS CASE.
1030      ; IF THE ONLY INPUT IS <CR>, ZERO WILL BE RETURNED IN RO
1031      ; AS WELL AS IN THE FIRST BYTE OF THE SPECIFIED BUFFER.
1032
1033      ; AUTHOR:          BERT HUBER, CSS/DP MUNICH          29 JUN 76
1034
1035      ; $.RED: CALL  $.PRO          ;PRINT PROMPT STRING

```

1035 005020 004737 006136

1036	005024	010637	007156		MOV	SP,INPREQ	SET SPECIAL REQUEST FLAG
1037	005030	026627	000002	000004	CMP	2(SP),#4	DETECT SPECIAL ASCII INPUT RUNNING
1038	005036	003402			BLE	100#	NO ASCII INPUT
1039	005040	010637	007174		MOV	SP,\$.ASF	SET SPECIAL ASCII INPUT FLAG
1040	005044			100#:			
1041	005044	005737	007150	1#:	TST	INFLAG	WAIT FOR INPUT COMPLETE
1042	005050	001375			BNE	1#	
1043	005052				PUSH	R1,R2,R3,R4,R5	SAVE REGS
1044	005064	012701	007026		MOV	#KBBUF,R1	GET INPUT POINTER
1045	005070	016602	000014		MOV	14(SP),R2	GET INPUT MODE INDICATOR
1046	005074	000172	005100		JMP	82#(R2)	SELECT REQ. INPUT
1047	005100	005112		2#:	\$.OCT		HANDLE OCTAL INPUT
1048	005102	005270			\$.DEC		HANDLE DECIMAL INPUT
1049	005104	005504			\$.BIN		HANDLE BINARY INPUT
1050	005106	005444			\$.BUF		HANDLE ASCII INPUT INTO BUFFER
1051	005110	005464			\$.ASC		HANDLE ASCII INPUT INTO BUFFER
1052	005112	005000		\$.OCT:	CLR	R0	CLEAR INPUT POINTER
1053	005114	005037	005244		CLR	6#	RESET COUNTER
1054	005120	121127	000015		CMPB	BR1,#15	ZERO INPUT ?
1055	005124	001014			BNE	3#	IF NE, SKIP
1056	005126				POP	R5,R4,R3,R2,R1	***AB00: RESTORE GEN. REGISTERS
1057	005140	011666	000002		MOV	(SP),2(SP)	***AB00: REAJUST RETURN ADDRESS
1058	005144	005726			TST	(SP)	***AB00:
1059	005146	005037	007156		CLR	INPREQ	***AB00: RESET SPECIAL REQUEST COUNTER
1060	005152	000261			SEC		***AB00: SET CARRY AS ZERO INPUT INDICATOR
1061	005154	000207			RETURN		***AB00: AND GO BACK TO CALLER
1062	005156	006300		3#:	ASL	R0	ADJUST INPUT CHARACTER
1063	005160	006300			ASL	R0	
1064	005162	006300			ASL	R0	
1065	005164	023727	005244	000006	CMP	6#,#6	ALL INPUT HANDLED ?
1066	005172	001406			BEQ	4#	MORE THAN 6 DIGITS INPUT
1067	005174	121127	000060		CMPB	BR1,#60	LEGAL OCTAL DIGIT ?
1068	005200	103403			BLO	4#	IF LO, NO
1069	005202	121127	000067		CMPB	BR1,#67	LEGAL OCTAL DIGIT ?
1070	005206	101412			BLOS	5#	IF LOS, YES
1071	005210			4#:	WRITE	MILOCT	ILLEGAL OCTAL INPUT
1072	005220				POP	R5,R4,R3,R2,R1	RESTORE REGS
1073	005232	000672			BR	\$.RED	RESTART INPUT SEQUENCE
1074	005234	142711	000060	5#:	BICB	#60,BR1	BUILD BINARY
1075	005240	152100			BISB	(R1),R0	INSERT INTO BUFFER
1076	005242	005227			INC	(PC)	COUNT INPUTS HANDLED
1077	005244	000000		6#:	.WORD	0	DIGIT COUNTER
1078	005246	121127	000015		CMPB	BR1,#15	END OF INPUT DETECTED?
1079	005252	001341			BNE	3#	IF NE, NO
1080	005254	023727	005244	000001	CMP	6#,#1	SET CARRY IF ONLY CR WAS THE INPUT
1081	005262	001001			BNE	7#	
1082	005264	000261			SEC		SET CARRY
1083	005266			7#:			
1084	005266	000557			BR	RED..E	END OF INPUT
1085	005270	005000		\$.DFC:	CLR	R0	CLEAR TRANSFER BUFFER
1086	005272	005004			CLR	R4	RESET COUNTER
1087	005274	122721	000015	1#:	CMPB	#15,(R1)	END OF INPUT ?
1088	005300	001402			BEQ	2#	IF EQ, YES
1089	005302	005204			INC	R4	DIGIT COUNTER
1090	005304	000773			BR	1#	KEEP COUNTING
1091	005306	012701	007026	2#:	MOV	#KBBUF,R1	SET INPUT BUFFER POINTER
1092	005312	020427	000005		CMP	R4,#5	MORE THAN 5 DIGITS ?

1093	005316	101412			BLOS	4:		; IF LOS, NO
1094	005320				WRITE		MILDEC	; ILLEGAL DECIMAL INPUT
1095	005330				POP		R5,R4,R3,R2,R1	; REST. REGS
1096	005342	000626			BR		\$.RED	; RESTART INPUT
1097	005344	005704			TST	4:	R4	; ZERO DIGIT COUNT ?
1098	005346	001527			BEQ		RED..E	; IF EQ, YES
1099	005350	006304			ASL		R4	; BUILD TABLE OFFSET
1100	005352	016402	005360		MOV		5:(R4),R2	; SETUP CORRECT PARAMETER
1101	005356	000406			BR		6:	; SKIP TABLE
1102	005360	000000	000001	000012	5:		.WORD	0,1,10..,100..,1000..,10000.
	005366	000144	001750	023420				
1103	005374	006204			6:		ASR	R4 ; REBUILD R4 CONTENT
1104	005376	005003					CLR	R3 ; RESET SAVE BUFFER
1105	005400	112103			MOVB		(R1)..,R3	; GET FIRST DIGIT
1106	005402	020327	000060		CMP		R3,#60	; CHECK LOWER DIGIT RANGE
1107	005406	103744			BLO		3:	; IF LO, ERROR
1108	005410	020327	000071		CMP		R3,#71	; CHECK UPPER DIGIT RANGE
1109	005414	101341			BHI		3:	; IF HI, ERROR
1110	005416	042703	177760		BIC		#177760,R3	; CONVERT TO OCTAL
1111	005422	005703			TST		R3	; ZERO DIGIT?
1112	005424	001405			BEQ		13:	; IF EQ, YES
1113	005426	000241			7:		CLC	; RESET CARRY
1114	005430	060200			ADD		R2,R0	; BUILD OCTAL NUMBER
1115	005432	103732			BCS		3:	; ERROR IF OVERFLOW
1116	005434	005303			DEC		R3	; ALL DONE?
1117	005436	001373			BNE		7:	; IF NE, NO
1118	005440	005304			13:		DEC	R4 ; NEXT DIGIT
1119	005442	000740					BR	4: ;
1120	005444	122711	000015		\$.BUF:		CMPB	#15,(R1) ; ZERO DIGIT ?
1121	005450	001001					BNE	1: ; SKIP IF NOT
1122	005452	005011					CLR	(R1) ; RESET FIRST INPUT WORD
1123	005454	010100			1:		MOV	R1,R0 ; TRANSFER BUFFER ADDRESS
1124	005456	005037	007174				CLR	\$.ASF ; CLEAR SPECIAL ASCII FLAG
1125	005462	000461					BR	RED..E ; END OF INPUT
1126	005464	005000			\$.ASC:		CLR	R0 ;
1127	005466	005037	007174				CLR	\$.ASF ; CLEAR SPECIAL ASCII INPUT FLAG
1128	005472	122711	000015				CMPB	#15,R1 ; ZERO INPUT?
1129	005476	001453					BEQ	RED..E ; IF EQ, YES
1130	005500	111100					MOVB	R1,R0 ; GET FIRST BYTE
1131	005502	000451					BR	RED..E ; END OF INPUT
1132	005504	005000			\$.BIN:		CLR	R0 ;
1133	005506	005037	005624				CLR	6: ; RESET COUNTER
1134	005512	122711	000015				CMPB	#15,R1 ; ZERO INPUT?
1135	005516	001443					BEQ	RED..E ; END OF INPUT
1136	005520	022737	000021	005624	1:		CMP	#17..,6: ; TOO MANY DIGITS?
1137	005526	001013					BNE	2: ; IF NE, NO
1138	005530				5:		WRITE	MILBIN ; ILLEGAL INPUT MESSAGE
1139	005540						POP	R5,R4,R3,R2,R1 ; REST. REGS
1140	005552	000137	005020				JMP	\$.RED ; REENTER
1141	005556	121127	000060		2:		CMPB	R1,#60 ; ZERO DIGIT ?
1142	005562	001407					BEQ	3: ; BRANCH IF YES
1143	005564	121127	000061				CMPB	R1,#61 ; ONE DIGIT ?
1144	005570	001011					BNE	4: ; BRANCH IF NOT
1145	005572	006300					ASL	R0 ; ADJUST TRANSFER BUFFER
1146	005574	052700	000001				BIS	#BIT0,R0 ; INSERT INTO TRANSFER BUFFER
1147	005600	000401					BR	7: ;
1148	005602	006300			3:		ASL	R0 ; ADJUST TRANSFER BUFFER

1149	005604	005201		7:	INC	R1		;POINT TO NEXT INPUT
1150	005606	005237	005624		INC	6:		;COUNT INPUTS
1151	005612	000742			BR	1:		;CHECK NEXT
1152	005614	121127	000015	4:	CMPB	BR1, #15		;END OF INPUT ?
1153	005620	001343			BNE	5:		;NO, ILLEGAL CHARACTER
1154	005622	000401			BR	RED..E		
1155	005624	000000		6:	.WORD	0		;COUNTER
1156	005626			RED..E:	POP	R5,R4,R3,R2,R1		;REST. REGS
1157	005640	011666	000002		MOV	(SP),2(SP)		;READJUST STACK RETURN ADDRESS
1158	005644	005726			TST	(SP).		; "
1159	005646	005037	007156		CLR	INPREQ		;RESET SPECIAL REQUEST POINTER
1160	005652	000207			RETURN			;RETURN TO CALLER

```

1162          .SBTTL      $.WRT
1163
1164          ;SUBROUTINE:  $.WRT
1165          ;*****      *****
1166
1167          ;THIS ROUTINE IS CALLED TO SETUP A MESSAGE PRINTOUT ON CONSOLE
1168          ;ALL PRINTING IS CONTROLLED BY INTERRUPT EXEPT <CR>, <LF>.
1169
1170          ;INPUT:        2(SP)      *      ADDRESS OF MESSAGE TEXT
1171          ;                (SP)      *      CALLER PC
1172
1173          ;OUTPUT:       MESSAGE ON TERMINAL
1174
1175          ;AUTHOR:       BERT HUBER CSS/DP MUNICH      28 JUN 76
1176
1177          ;MODIFIED BY:  JOHN LEVESQUE   CSS ISG   23-JAN 83
1178
1179          ;
1180          ;              MODIFICATION REQUIRED TO HANDLE XON AND XOFF
1181          ;
1182          ;-
1183          ;.PSECT CSSMON
1184          ;.ENABL  LSB
1185          $.WRT: TST      OUTFLG      ;OUTPUT RUNNING ?
1186          BNE     $.WRT      ;IF NE, YES - WAIT
1187          TST     INFLAG      ;INPUT RUNNING ?
1188          BNE     $.WRT      ;IF NE, YES - WAIT
1189          MOV     SP,OUTFLG    ;SET OUTPUTFLAG
1190          MOV     2(SP),(PC).  ;GET MESSAGE POINTER
1191          $.MSP: .WORD      0      ;MESSAGE POINTER
1192          MOV     @1$,@#64     ;SETUP VECTOR
1193          MOV     @340,@#66    ;
1194          MOV     (SP), (SP)   ;UPDATE STACK
1195          BIS     @100,@#TPS    ;ENABLE INTERRUPTS
1196          TST     OUTFLG      ;OUTPUT RUNNING?      ;**GS0001**
1197          BNE     .-4          ;IF NE,YES->WAIT      ;**GS0001**
1198          RETURN
1199          1$:  CMPB     @23,@#TKB  ;CHECK FOR XOFF
1200          BEQ     3$          ;XOFF CONDITION WAIT TO CLEAR
1201          BR      20$         ;NO XOFF THEN CONTINUE
1202          3$:  CMPB     @21,@#TKB  ;CHECK FOR XON
1203          BNE     1$          ;NO XON GO WAIT FOR IT
1204          20$:
1205          CMPB     @$.MSP,@' ]  ;END OF MESSAGE ?
1206          BEQ     WRT..E      ;;;IF EC YES
1207          CMPB     @$.MSP,@' [  ;;;<CR>,<LF>?
1208          BNE     30$         ;;;IF NE, NO
1209          BIC     @100,@#TPS    ;;;DISABLE INTERRUPTS
1210          MOV     @15,@#TPB     ;;;SEND <CR>
1211          TSTB    @#TPS        ;;;WAIT FOR DONE
1212          BPL     .-4          ;;;
1213          INC     $.MSP        ;;;POINT TO NEXT CHARACTER
1214          MOV     @12,@#TPB     ;;;SEND <LF>
1215          BIS     @100,@#TPS    ;;;RE-ENABLE INTERRUPT
1216          RTI
1217          30$: MOVB     @$.MSP,@#TPB ;;;SEND NEXT CHARACTER
1218          INC     $.MSP        ;;;POINT TO NEXT
1219          RTI                  ;;;RETURN TO CALLER

```

1219	006060	042737	000100	177564	WRT..E:	BIC	#100.8#TPS	;;;DISABLE OUTPUT INTERRUPT
1220	006066	005037	007152			CLR	OUTFLG	;;;RESET OUTPUT FLAG
1221	006072	005737	007170			TSI	SPEFLG	;;;THIS FLAG IS SET ONLY
1222	006076	001410				BEQ	4\$	;;;WHEN DURING A PRINTOUT
1223	006100	005037	007170			CLR	SPEFLG	;;;THE <CNTRL C> COMMAND WAS
1224	006104	005037	007156			CLR	INPREQ	;;;NECESSARY TO INTERRUPT THE
1225	006110	004737	003360			CALL	\$..RSV	;;;RUNNING PRINTOUT
1226	006114	000137	003520			JMP	\$\$\$SPE	;;;NOW NEW SPEC. INPUT IS PROCESSED
1227	006120	000002		4\$:		RTI		;;;RETURN TO MAIN PROGRAM
1228						.DSABL	LSB	

```

1230          .SBTTL  $..KBO
1231          ;*
1232          ;SUBROUTINE:  $..KBO
1233          ;.....
1234          ;
1235          ;KEYBOARD ECHO ROUTINE FOR TEMINAL INPUT
1236          ;
1237          ;INPUT:          (SP)      -      CALLER PC
1238          ;                  RO      -      PATTERN TO ECHO
1239          ;
1240          ;OUTPUT:         PRINT PATTERN TO TERMINAL
1241          ;
1242          ;AUTHOR:          BERT HUBER CSS/DP MUNICH      28-JUN 76
1243          ;
1244          ;.PSECT CSSMON
1245          .ENABL  LSB
1246 006122 105737 177564  $..KBO:  TSTB  @TPS      ;ECHO RUNNING ?
1247 006126 100375      BPL  $..KBO      ;IF PL, WAIT
1248 006130 010037 177566  MOV  RO,@TPB    ;ECHO CHARACTER
1249 006134 000207      RETURN      ;RETURN TO CALLER
1250          .DSABL  LSB

```

```

1252          .SBTTL  $..PRO
1253          ;
1254          ;SUBROUTINE:  $..PRO
1255          ;*****
1256          ;
1257          ;THIS ROUTINE IS CALLED TO PRINT THE DEFAULT KEYBOARD
1258          ;PROMPT STRING ON THE TERMINAL, AND TO SET THE INPUT FLAG.
1259          ;
1260          ;INPUT:          (SP)      -      CALLER PC
1261          ;
1262          ;OUTPUT:         PROMPT STING ON TERMINAL
1263          ;
1264          ;AUTHOR:         BERT HUBER CSS/DP MUNICH          28-JUN 76
1265          ;               MZ001
1266          ;               *****
1267          ;               THE USE OF THIS MACRO'S ALLOWS TO USE ONE VERSION OF CDM
1268          ;               ON LSI AND PDP.
1269          ;
1270          ;-
1271          .IDENT  /V1.0/
1272          ;.PSECT  CSSMON
1273          .ENABL  LSB
1274 006136 005037 007150  $..PRO: CLR      INFLAG      ;;;RESET INPUT FLAG
1275 006142          PREAD  STATUS      ;SAVE STATUS      ;MZ001
1276 006166          PSET   #0          ;CLEAR PS        ;MZ001
1277 006212          WRITE  PROMES      ;;;WRITE PROMPT STRING
1278 006222 005737 007152 1$:      TST    OUTFLG      ;WAIT FOR OUTPUT COMPLETE
1279 006226 001375          BNE     1$
1280 006230          PSET   STATUS      ;RELOAD STATUS   ;MZ001
1281 006254 010637 007150          MOV    SP,INFLAG      ;;;SET INPUT FLAG
1282 006260 000207          RETURN      ;RETURN TO CALLING PROGRAM
1283          .DSABL  LSB

```

```

1285          .SBTTL $..STX
1286          ;*
1287          ;SUBROUTINE: $..STX
1288          ;*****
1289          ;
1290          ;THIS ROUTINE PRINTS THE 'SYNTAX ERROR' MESSAGE
1291          ;
1292          ;INPUTS:          (SP)= CALLER PC
1293          ;
1294          ;OUTPUT:         MESSAGE ON TERMINAL
1295          ;
1296          ;AUTHOR:         BERT HUBER CSS/DP MUNICH      28 JUN '76
1297          ;
1298          ;
1299          ;      MZ001
1300          ;      *****
1301          ;      THE USE OF THIS MACRO'S ALLOWS TO RUN ONE VERSION OF CDM
1302          ;      ON LSI AND PDP.
1303          ;
1304          ;-
1305          ;.PSECT CSSMON
1306          .ENABL LSB
1307 006262 000240 $..STX: NOP          ;;;DUMMY
1308 006264          PREAD STATUS ;SAVE STATUS ;MZ001
1309 006310          PSET  #0      ;CLEAR PS   ;MZ001
1310 006334          WRITE MSYERR ;WRITE SYNTAX ERROR MESSAGE
1311 006344 005737 007152 1$: TST OUTFLG ;WAIT FOR OUTPUT COMPLETE
1312 006350 001375          BNE 1$      ;
1313 006352          PSET STATUS ;RELOAD PS ;MZ001
1314 006376 000207          RETURN ;;;RETURN TO CALLER
1315          .DSABL LSB

```

```
1317 .SBTTL $.DMP
1318 ;
1319 ;SUBROUTINE: $.DMP
1320 ;*****
1321 ;
1322 ;THIS ROUTINE PRINTS ON THE TERMINAL PRINTER NUMBERS
1323 ;IN BINARY, DECIMAL AND OCTAL FORMAT, DEPENDING ON THE
1324 ;TYPE-ARGUMENT OF THE CALLING DUMP MACRO.
1325 ; THIS MONITOR WAS MODIFIED IN JANUARY 1983 TO INCLUDE XON AND XOFF
1326 ; TERMINAL FEATURES. ALSO INCLUDED SOME MACRO DIRECTIVES TO ALLOW
1327 ; ASSEMBLY ON VAX.
1328 ;
1329 ;
```

```

1331      ;
1332      ;INPUT:      4(SP)      =      VALUE TO DUMP
1333      ;              2(SP)      =TYPE OF DUMP      :  0 = OCTAL
1334      ;              ;              2 = DECIMAL
1335      ;              ;              4 = BINARY
1336      ;
1337      ;              (SP)      =RETURN PC
1338      ;
1339      ;OUTPUT:      DUMP ON TERMINAL
1340      ;
1341      ;AUTHOR:      BERT HUBER CSS/DP MUNICH      28-JUN-76
1342      ;
1343      ;.PSECT CSSMON
1344      ;.ENABL LSB
1345      $..DMP: PUSH RO      ;SAVE RO
1346      MOV 4(SP),RO      ;GET TYPE ARGUMENT
1347      TST OUTFLG      ;OUTPUT RUNNING ?
1348      BNE . 4      ;IF NE, WAIT
1349      JMP @D..TYP(RO)      ;SELECT FUNCTION ROUTINE
1350      D..TYP: $..DOC
1351      $..DDC
1352      $..DBN
1353      $..DOC: MOV @6,-(SP)      ;SETUP DIGIT COUNT
1354      MOV 10(SP),-(SP)      ;GET VALUE TO DUMP
1355      MOV @'0,RO      ;SETUP BASIC ASCII VALUE
1356      ASL (SP)      ;BUILD FIRST DIGIT
1357      BCC 1$      ;IF CC, DIGIT IS A '0'
1358      INC RO      ;OTHERWISE DIGIT IS A ?
1359      CALL $..KBO      ;DUMP DIGIT
1360      DEC 2(SP)      ;COUNT DIGITS DUMPED
1361      BEQ 4$      ;IF EQ, ALL DONE
1362      MOV @'0,RO      ;SETUP NEXT DIGIT BASE
1363      ASL (SP)      ;BUILD NEXT DIGIT
1364      BCC 2$      ;IF CC, DIGIT INCLUDES ?????
1365      ADD @4,RO      ;DIGIT = BASE+4
1366      ASL (SP)      ;ANALYZE NEXT BIT
1367      BCC 3$      ;IF CC, DIGIT INCLUDES ???
1368      CMPB (RO)+,(RO)+      ;DIGIT = DIGIT + 2
1369      ASL (SP)      ;ANALYZE LAST BIT OF DIGIT
1370      BCC 1$      ;IF CC, DIGIT MUST BE EVEN
1371      INC RO      ;OTHERWISE DIGIT IS ODD
1372      BR 1$      ;GO AND DUMP
1373      4$: CMP (SP)+,(SP)+      ;READJUST STACK
1374      BR DMP..E      ;END OF DUMP
1375      ;.DSABL LSB
1376      ;.ENABL LSB
1377      $..DDC: MOV @5,-(SP)      ;SETUP A DIGIT COUNT
1378      MOV 10(SP),-(SP)      ;GET VALUE TO DUMP
1379      MOV @'0,RO      ;BUILD LEADING ZERO
1380      CALL $..KBO      ;AND PRINT
1381      MOV @'0,RO      ;SETUP BASIC ASCII VALUE
1382      1$: CMP @$POINT,(SP)      ;CMP VALUE WITH ACT. DEC. DIGIT
1383      BHI 3$      ;VALUE IS SMALLER
1384      INC RO      ;COUNT NUMBER OF DIVISIONS
1385      SUB @$POINT,(SP)      ;DIVIDE BY ACT. DEC. DIGIT
1386      BR 2$      ;DO NEXT
1387      2$: CALL $..KBO      ;PRINT THIS DEC. DIGIT
1388      ADD @2,$POINT      ;POINT TO NEXT DEC. VALUE

```

```

1388 006574 005366 000002          DEC      2(SP)      ;COUNT NUMBER OF CONF. DIGITS
1389 006600 001357          BNE      1$              ;ALL DONE
1390 006602 012737 006616 006614    MOV      @TABLE,$POINT ;REBUILD FOR NEXT USER
1391 006610 022626          CMP      (SP)+,(SP)+      ;CORRECT STACK FOR RETURN
1392 006612 000430          BR       DMP..F           ;END OF DEC. DUMP
1393 006614 006616          $POINT: $TABLE
1394 006616 023420          $TABLE: 10000.
1395 006620 001750          1000.
1396 006622 000144          100.
1397 006624 000012          10.
1398 006626 000001          1
1399          .DSABL      LSB
1400          .ENABL      LSB
1401 006630 012766 000020 000004    $..DBN: MOV      @16..4(SP) ;SETUP PRINT POINTER
1402 006636 000241          1$:      CLC              ;RESET CARRY
1403 006640 006166 000006          ROL      6(SP)      ;SELECT FIRST CHARACTER
1404 006644 103410          BCS      3$              ;IF " 1 " BRANCH
1405 006646 012700 000060          MOV      @60,R0      ;SETUP A "0" CHARACTER
1406 006652 004737 006122          2$:      CALL     $..KBO ;DUMP CHARACTER
1407 006656 005366 000004          DEC      4(SP)      ;COUNT CHARACTERS DUMPED
1408 006662 001365          BNE      1$              ;IF NE, CONTINUE
1409 006664 000403          BR       DMP..E           ;END OF DUMP
1410 006666 012700 000061          3$:      MOV      @61,R0 ;SETUP A " 1 " CHARACTER
1411 006672 000767          BR       2$              ;DUMP IT
1412          .DSABL      LSB
1413 006674          DMP..E: POP      R0              ;RESTORE R0
1414 006676 062706 000006          ADD      @6,SP      ;READJUST STACK
1415 006702 000176 177772          JMP      @-6(SP)    ;RETURN TO CALLER
1416          .DSABL      LSB

```

1418
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1428
 1429
 1430
 1431 006706
 1432 006716
 1433 006730
 1434 006740
 1435 006752 000207

```

      .SBTIL $..ADP
;
; THIS SUBROUTINE IS USED TO PRINT OUT ADDITIONAL INFORMATION
; ABOUT AN ERROR. MOST TIMES IT WILL BE USED TO PRINT THE
; DATA IN CASE OF DATA ERROR.
;
;
; INPUTS : (SP) *ADDRESS OF FIRST ARGUMENT
;
; OUTPUT:  NONE
;
; AUTHOR:  MICHAEL ZILLER CSS DP MUNICH MAY 1977
$..ADP: WRITE MER1 ;"SHOULD BE"
        DUMP OCT,$SHLD ;DUMP DATA
        WRITE MER2 ;"WAS"
        DUMP OCT,$WAS ;DUMP DATA
        RETURN ;RETURN TO MAIN
  
```

```

1437          .SBTTL 1..TOT
1438          ;
1439          ; THIS ROUTINE IS ENTERED AFTER THE MACRO "ADRTST" HAS BEEN EVOCED
1440          ; AND A BUS TIMEOUT HAS HAPPENED
1441          ;
1442          ; INPUT: 4(SP) = PC FOR PROGRAM CONTINUATION
1443          ;          RO = REGISTER ADDRESS (E.G. 164000) WHICH SHOULD BE TESTED
1444          ;
1445          ; OUTPUT:      THE ERROR MESSAGE:
1446          ;              THIS ADDRESS IS NOT AVAILABLE ON UNIBUS: 164000
1447          ; AUTHOR:      ALBERT BREM, CSS MUC      FEB 1979
1448          ;
1449          ;
1450          ;
1451          1451 006754 062706 000004          1..TOT ADD    #4,SP          ;CORRECT SP
1452          1452 006760          WRITE    $MNOAV          ;SAY ADDRESS NOT AVAILABLE ON UNIBUS
1453          1453 006770          DUMP     OCT,RO          ;DUMP REGISTER ADDRESS
1454          1454 007000 005737 007152          TST     OUTFLG          ;SYNCHRONIZE PRINTOUT
1455          1455 007004 001375          BNE     .-4          ;
1456          1456 007006 012737 000006 000004          MOV     #6,B#4          ;REBUILD TRAP CATCHER
1457          1457 007014 000136          JMP     @B(SP).          ;JUMP TO CONTINUATION LABEL
1458          ;
1459          ;
1460          ;
1461          ;

```

```

1463          .SBTTL 1..BUF
1464          ;
1465          ;SUBSECTION TO CONTAIN ALL CDM BUFFERS, CONSTANTS AND
1466          ;'NOT MACRO DEFINED' PARAMETERS.
1467          ;
1468          ;AUTHOR:          BERT HUBER CSS/DP MUNICH
1469          ;
1470          ;.PSECT CSSMON
1471          .ENABL LSB
1472          ;MEMORY MANAGEMENT DEFINITIONS
1473          172360 KDSARO=172360
1474          172320 KSDSRO=172320
1475          172340 KISARO=172340
1476          172352 KISAR5=172352
1477          172354 KISAR6=172354
1478          172356 KISAR7=172356
1479          172300 KISDR0=172300
1480          172314 KISDR6=172314
1481          172316 KISDR7=172316
1482          172200 SISDR0=172200
1483          177660 UDSARO=177660
1484          177620 UDSDR0=177620
1485          177640 UISARO=177640
1486          177650 UISAR4=177650
1487          177652 UISAR5=177652
1488          177654 UISAR6=177654
1489          177656 UISAR7=177656
1490          177600 UISDR0=177600
1491          177610 UISDR4=177610
1492          177612 UISDR5=177612
1493          177614 UISDR6=177614
1494          177616 UISDR7=177616
1495          170200 UBMPR=170200
1496          140000 CMODE=140000
1497          030000 PMODE=30000
1498          177572 SR0=177572
1499          172516 SR3=172516
1500 007016 000000 000000 000000 000000 000000 000000 000000 000000
1501 007020 000000 000000 000000 000000 000000 000000 000000 000000
1502 007022 000000 000000 000000 000000 000000 000000 000000 000000
1503
1504
1505
1506 007026 KBBUFF: .BLKB 80.
1507 007146 KBBEND:
1508 007146 007026 KBBPNT: .WORD KBBUFF
1509 007150 000000 INFLAG: .WORD 0
1510 007152 000000 OUTFLG: .WORD 0
1511 007154 000000 ERRFLG: .WORD 0
1512 007156 000000 INPREQ: .WORD 0
1513 007160 000000 PSDSWR: .WORD 0
1514 007162 000000 RUBFLG: .WORD 0
1515 007164 000000 STATUS: .WORD 0
1516 007166 000000 $.ER: .WORD 0
1517 007170 000000 SPEFLG: .WORD 0
1518 007172 001000 SCOPAD: 1000
1519 007174 000000 $.ASF: .WORD 0

```

```

; IF 0:=LSI; IF 1:=PDP
; IF 0:NO MM; IF 1:WITH MM
; TOP OF CORE AND CORE EXTENSION
; STARTING FROM HERE TILL PROMES
; THE CONTENTS WILL BE CLEARED
; WITH "ABO"
; TERMINAL INPUT BUFFER
; REFERENCE LABEL
; INPUT BUFFER POINTER
; INPUT "RUNNING" FLAG
; OUTPUT "RUNNING" FLAG
; "ERROR HANDLING REQ." FLAG
; "WAIT FOR SPECIAL INPUT" FLAG
; SOFTWARE SWITCH REGISTER
; RUBOUTMODE-INDICATOR FLAG
; BUFFER PS-CONTENT IF SAVED
; HOLDS CURRENT ERROR NUMBER
; USED IN $.KBI AND $.WRT
; SAVE AREA FOR SCOPE LOOP ADDRESS
; SPECIAL ASCII INPUT FLAG

```

Address	Hex	ASCII	Comment
1520	007176	000000	
1521	007200	000000	
1522			
1523	007202	133	103
1524	007210	133	040
1525	007213	133	113
1526	007256	133	111
1527	007316	133	111
1528	007360	133	111
1529	007421	133	111
1530	007445	133	102
1531	007503	133	123
1532	007524	133	123
1533	007601	133	123
1534	007652	133	105
1535	007662	040	040
1536	007704	133	120
1537	007737	133	105
1538	010002	133	056
1539	010032	133	117
1540	010101	133	111
1541	010144	040	040
1542	010156	133	133
1543	010252	110	111
1544	010304	133	040
1545	010333	040	040
1546	010351	133	133
1547	010426	133	115
1548	010472	133	111
1549	010542	133	127
1550	010612	066	135
1551			
1552			
1553			
1554	010614		

```

1      .TITLE  IEC11 A DIAGNOSTIC
2      .NLIST  CND
3      .LIST  MEB
4
5      ;
6      ;
7      ;
8      ;
9      ;
10     ;
11     ;
12     ;
13     ;
14     ;
15     ;
16     ;
17     ;
18     ;
19     ;
20     ;
21     ;
22     ;
23     ;
24     ;
25     ;
26     ;

010614 012746 021606      WRITE  M1      ;IDENTIFY TEST
010614 004737 005654      MOV     #M1,-(SP)      ;SAVE #M1 ONTO STACK
010620      CALL     $.WRT      ;INITIATE PRINTOUT

010624 012746 021510      RGAIN: WRITE  MES1      ;ENTER FIRST REG OF IEC11 A
010624 004737 005654      MOV     #MES1,-(SP)      ;SAVE #MES1 ONTO STACK
010630      CALL     $.WRT      ;INITIATE PRINTOUT
010634 005046      READ  OCT,RO      ;INPUT REQUEST
010634 004737 005020      CLR     (SP)      ;INDICATE OCTAL INPUT REQUIRED
010642      CALL     $.RED      ;ACCEPT INPUT
010642 005700      IF RO NE #0 GOTO 1$      ;
010644 001402      TST     RO      ;IS RO ZERO ?
010646 000137 010656      BEQ     .+6      ;
010652 012700 160010      JMP     1$      ;
010656 032700 000007      MOV     #160010,RO      ;SAVE DEFAULT VALUE
010662 001360      BIT     #7,RO      ;CHECK THREE LSB OF INPUT
010664      BNE     RGAIN      ;IF NOT CLEAR,REQUEST AGAIN
010664 020027 160010      IF RO LO #160010 GOTO RGAIN      ;
010670      CMP     RO,#160010      ;COMPARE RO AND #160010
010672 103001      BHIS     .+4      ;
010674 000754      BR      RGAIN      ;
010674 020027 177776      IF RO HI #177776 GOTO RGAIN      ;
010700      CMP     RO,#177776      ;COMPARE RO AND #177776
010702 101401      BLOS     .+4      ;
010704 000750      BR      RGAIN      ;
010704 012737 006754 000004      ADRTST  RGAIN      ;
010712 012746 010624      MOV     #$.TOT,#04      ;PROVIDE TIMEOUT ENTRY
010716 005710      MOV     #RGAIN,-(SP)      ;SAVE RETURN ADDRESS ON STACK
010720 012737 000006 000004      TST     (RO)      ;IS ADDRESS AVAILABLE ?
010726 005726      MOV     #6,#04      ;RESTORE TIMEOUT VECTOR
010730 010037 024204      TST     (SP)+      ;CORRECT STACK
010734 062700 000002      MOV     RO,CIR      ;SAVE CIR ADDRESS
010740      ADD     #2,RO      ;BUILD NEXT ADDRESS
010740 012737 006754 000004      ADRTST  RGAIN      ;
010746 012746 010624      MOV     #$.TOT,#04      ;PROVIDE TIMEOUT ENTRY
010752 005710      MOV     #RGAIN,-(SP)      ;SAVE RETURN ADDRESS ON STACK
010754 012737 000006 000004      TST     (RO)      ;IS ADDRESS AVAILABLE ?
010762 005726      MOV     #6,#04      ;RESTORE TIMEOUT VECTOR
010764 010037 024206      TST     (SP)+      ;CORRECT STACK
010770 062700 000002      MOV     RO,SMR      ;SAVE SMR REG ADDRESS
010774      ADD     #2,RO      ;BUILD NEXT ADDRESS
010774 012737 006754 000004      ADRTST  RGAIN      ;
011002 012746 010624      MOV     #$.TOT,#04      ;PROVIDE TIMEOUT ENTRY
011006 005710      MOV     #RGAIN,-(SP)      ;SAVE RETURN ADDRESS ON STACK
011010 012737 000006 000004      TST     (RO)      ;IS ADDRESS AVAILABLE ?
011016 005726      MOV     #6,#04      ;RESTORE TIMEOUT VECTOR
011020 010037 024212      TST     (SP)+      ;CORRECT STACK
011024 062700 000001      MOV     RO,IOR      ;SAVE IOR REG ADDRESS
011030 010037 024210      ADD     #1,RO      ;BUILD HIGH-BYTE ADDRESS
011034 062700 000001      MOV     RO,IORHB      ;OF IOR
011040      ADD     #1,RO      ;BUILD NEXT ADDRESS
011040 012737 006754 000004      ADRTST  RGAIN      ;
011040      MOV     #$.TOT,#04      ;PROVIDE TIMEOUT ENTRY

```

011046	012746	010624		MOV	ARGAIN, (SP)	;SAVE RETURN ADDRESS ON STACK
011052	005710			TST	(RO)	;IS ADDRESS AVAILABLE ?
011054	012737	000006	000004	MOV	#6, #44	;RESTORE TIMEOUT VECTOR
011062	005726			TST	(SP)	;CORRECT STACK
27 011064	010037	024214		MOV	RO, VSR	;SAVE VSR REG ADDRESS
28 011070				VECAIN: WRITE	MES2	;REQUEST VECTOR ADDRESS OF IEC11 A
011070	012746	021666		MOV	#MES2, (SP)	;SAVE #MES2 ONTO STACK
011074	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
29 011100				READ	OCT, RO	;INPUT REQUEST
011100	005046			CLR	-(SP)	;INDICATE OCTAL INPUT REQUIRED
011102	004737	005020		CALL	\$.RED	;ACCEPT INPUT
30 011106				IF RO NE #0 GOTO 2:		
011106	005700			TST	RO	;IS RO ZERO ?
011110	001402			BEQ	.+6	
011112	000137	011122		JMP	2:	
31 011116	012700	000270		MOV	#270, RO	;SAVE DEFAULT VALUE
32 011122				IF RO LO #40 GOTO VECAIN		
011122	020027	000040		CMP	RO, #40	;COMPARE RO AND #40
011126	103001			BHIS	.+4	
011130	000757			BR	VECAIN	
33 011132				IF RO HI #774 GOTO VECAIN		
011132	020027	000774		CMP	RO, #774	;COMPARE RO AND #774
011136	101401			BLOS	.+4	
011140	000753			BR	VECAIN	
34 011142	020077	013046		CMP	RO, #VSR	;IS INPUT EQUAL TO VSR ?
35 011146	001406			BEQ	SAV	;IF YES, SAVE INPUT AND CONTINUE
36 011150				WRITE	MES6	;IF NOT, WRITE MESSAGE
011150	012746	022034		MOV	#MES6, -(SP)	;SAVE #MES6 ONTO STACK
011154	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
37 011160	000137	011070		JMP	VECAIN	;REENTER
38 011164	010037	024224		SAV: MOV RO, VEC		;SAVE IEC11-A VECTOR ADDRESS
39 011170				ADDRIN: WRITE	MES5	;REQUEST IEC11-A BUS ADDRESS
011170	012746	021750		MOV	#MES5, -(SP)	;SAVE #MES5 ONTO STACK
011174	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
40 011200				READ	OCT, RO	;INPUT
011200	005046			CLR	(SP)	;INDICATE OCTAL INPUT REQUIRED
011202	004737	005020		CALL	\$.RED	;ACCEPT INPUT
41 011206				IF RO NE #0 GOTO 3:		
011206	005700			TST	RO	;IS RO ZERO ?
011210	001402			BEQ	.+6	
011212	000137	011222		JMP	3:	
42 011216	012700	000035		MOV	#35, RO	;SAVE DEFAULT VALUE
43 011222				IF RO LO #1 GOTO ADDRIN		
011222	020027	000001		CMP	RO, #1	;COMPARE RO AND #1
011226	103001			BHIS	.+4	
011230	000757			BR	ADDRIN	
44 011232				IF RO HI #36. GOTO ADDRIN		
011232	020027	000044		CMP	RO, #36.	;COMPARE RO AND #36.
011236	101401			BLOS	.+4	
011240	000753			BR	ADDRIN	
45 011242	010037	024226		MOV	RO, IECAAD	;SAVE IEC11 A BUS ADDRESS
46 011246	012737	000100	024216	2: MOV	#100, TALKER	;PRE-ADDRESS TALKER
47 011254	012737	000040	024220	MOV	#40, LISTNR	;PRE ADDRESS LISTENER
48 011262	050037	024216		BIS	RO, TALKER	;SAVE TALKER ADDRESS
49 011266	050037	024220		BIS	RO, LISTNR	;SAVE LISTENER ADDRESS
50 011272	004737	011740		JSR	PC, TEST0	;TEST LON SWITCH FEATURE; **V02E00**
51 011276	017700	012712		8: MOV	#VSR, RO	;CHECK VECTOR ADDRESS

52	011302	020027	000774		CMP	R0, #774	;LEGAL VECTOR ?
53	011306	101003			BHI	30;	;IF HI NO ;**V01E02**
54	011310	020027	000040		CMP	R0, #40	;LEGAL VECTOR ;**V01E02**
55	011314	103005			BHIS	4;	;IF HIS YES ;**V01E02**
56	011316			30:	WRITE	MES2;	;REQUEST ANOTHER VECTOR ADDRESS
	011316	012746	021666		MOV	#MES2, (SP)	;SAVE #MES2 ONTO STACK
	011322	004737	005654		CALL	I..WRT	;INITIATE PRINTOUT
57	011326	000763			BR	8;	;CHECK AGAIN
58	011330	020027	000040	4:	CMP	R0, #40;	;CHECK RANGE
59	011334	1037 0			BLO	30;	;IF LO, INVALID
60	011336	010037	024222		MOV	R0, VECTOR	;GET VECTOR ADDRESS

```

62      .SBTTL TEST SELECT ROUTINE
63      ;...SPC: REMVEC  #VECA          ;SAVE #VECA ONTO STACK
        MOV      #VECA, (SP)          ;RESET #VECA VECTOR
        EMT      2                    ;WAIT FOR ANY OUTPUT COMPLETE
64      011342 012746 024224          TST      OUTFLG
        011346 104002                  BNE      .-4
65      011350 005737 007152          RESET          ;CLEAR THE WORLD
66      011354 001375                  SET6      #@TKS          ;RE-ENABLE TTY
67      011356 000005                  BIS      #BIT6, @TKS
        011360 052737 000100 177560    WRITE     TPMSG          ;ASK FOR PRINTING START MESSAGES
        011366 012746 024070          MOV      #TPMSG, -(SP)      ;SAVE #TPMSG ONTO STACK
        011372 004737 005654          CALL     $.WRT              ;INITIATE PRINTOUT
68      011376 012746 000010          READ      ASC              ;Y OR N?
        011402 004737 005020          MOV      #10, -(SP)         ;INDICATE ASCII INPUT IN RO REQUIRED
69      011406 020027 000131          CALL     $.RED              ;ACCEPT INPUT
        011412 001402                  IF RO NE #Y GOTO SFLAG      ;SET FLAG
        011414 000137 011426          CMP      RO, #Y            ;COMPARE RO AND #Y
70      011420 005037 011670          BEQ      .+6
        011424 000402                  JMP      SFLAG
        011426 005237 011670          CLR      MSGFLG            ;PRINT OUT START/END MESSAGES
71      011432 012746 024044          BR       LAB11
        011436 004737 005654          SFLAG: INC      MSGFLG      ;NO PRINT OUT
        011442 012746 000002          LAB11: WRITE     MESNEX      ;REQUEST NEXT TEST TO RUN
        011446 004737 005020          MOV      #MESNEX, -(SP)      ;SAVE #MESNEX ONTO STACK
72      011452 020027 000016          CALL     $.WRT              ;INITIATE PRINTOUT
        011456 101401                  READ      DEC, RO          ;WAIT FOR INPUT
        011460 000764                  MOV      #2, (SP)          ;INDICATE DECIMAL INPUT REQUIRED
73      011462 012746 000002          CALL     $.RED              ;ACCEPT INPUT
        011466 000137 011536          IF RO HI #14. GOTO LAB11    ;COMPARE RO AND #14.
74      011472 012737 177777 011666    CMP      RO, #14.
        011500 006300                  BLOS     .+4
        011502 010027                  BR       LAB11
        011504 000000                  IF RO EQ #0 GOTO TESTAL ; ;IS RO ZERO ?
75      011506 013737 011504 011664    TST      RO
        011514 062737 011626 011664    BNE      .+6
        011522 017737 000136 011664    JMP      TESTAL
        011530 004777 000130          MOV      #177777, TSTREP    ;LOOP ON SINGLE TESTS
        011534 000764                  ASL      RO                ;BUILD WORD OFFSET INTO TABLE
        011536 012737 000000          MOV      RO, (PC)+          ;AND SAVE
        011544 012727 011530          MOV      0, OFFSET          ;HOLDS CURRENT TEST NUMBER
        011550 000000          OFFSET: .WORD 0
        011552 013737 011550 011664    SINGLE: MOV     OFFSET, OFFCNT
        011556 017737 000100 011664    ADD      #TT, OFFCNT
        011560 004777 000072          MOV      #OFFCNT, OFFCNT
        011562 023727 011550 011662    MOV      PC, #OFFCNT
        011566 001404                  JSR      PC, #OFFCNT      ;GOTO SELECTED TEST
        011572 062737 000002 011550    BR       SINGLE          ;REPEAT TEST
        011576 000760                  TESTAL: MOV     #0, TSTREP    ;NO LOOP ON SINGLE TESTS
        011580 012746 024166          MOV      #TT+2, (PC)+      ;GET ADDRESS OF FIRST TEST
        011584 013737 011550 011664    SEQPNT: .WORD 0          ;POINTER FOR TEST IN PROGRESS
        011588 017737 000100 011664    SEQ:      MOV     SEQPNT, OFFCNT
        011592 004777 000072          MOV      #OFFCNT, OFFCNT    ;LOAD ADDRESS OF NEXT TEST
        011596 023727 011550 011662    JSR      PC, #OFFCNT      ;GOTO EXECUTE TEST
        011600 001404                  CMP      SEQPNT, #TTE 4    ;ALL DONE ?
        011602 062737 000002 011550    BEQ      TESTL1          ;IF EQ, YES
        011606 000760                  ADD      #2, SEQPNT        ;POINT TO NEXT TEST
        011610 012746 024166          BR       SEQ              ;AND DO NEXT
        011612 012746 024166          TESTL1: WRITE     MESEND      ;END OF TEST
        011612 012746 024166          MOV      #MESEND, (SP)      ;SAVE #MESEND ONTO STACK

```

IEC11-A DIAGNOSTIC
TEST SELECT ROUTINE

MACRO M1200 30 MAR 84 16:02 PAGE 30-1

SEQ 61

	011616	004737	005654		CALL	\$.WRT		;INITIATE PRINTOUT
98	011622	000137	011536		JMP	TESTAL		
99	011626	000000	012106	012250	TT:	.WORD	0,TEST1,TEST2,TEST3,TEST4,TEST5,TEST6,TEST7	
	011634	012456	012700	013302				
	011642	013444	013754					
100	011646	014372	014516	015400		.WORD	TEST8,TEST9,TEST10,TEST11,TEST12,TEST13,TEST14	
	011654	017046	017510	020176				
	011662	021114						
101	011664	000000			OFFCNT:	.WORD	0	;
102	011666				TTE:			
103	011666	000000			TSTREP:	.WORD	0	
104	011670	000000			MSGFLG:	.WORD	0	;MESSAGE FLAG

```

107          .SBTTL  LOCAL MACRO DEFINITIONS
108          ;
109          ; THIS SECTION CONTAINS ONLY MACRO DEFINITIONS
110          ; USED IN THE IEC-11A DIAGNOSTIC.
111          ;
112          .MACRO  MCLEAR
113          BIS     @BIT5,@CIR      ;EXECUTE MASTER CLEAR
114          .ENDM  MCLEAR
115          ;
116          .MACRO  SACS
117          BIS     @BIT0,@CIR      ;SET SACS BIT IN CIR
118          .ENDM  SACS
119          ;
120          .MACRO  INHSRQ
121          BIS     @BIT10,@CIR     ;SET INHIBIT SRQ INTERRUPT
122          .ENDM  INHSRQ          ; **V02E00**
123          ;
124          .MACRO  INTENB
125          BIS     @BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
126          .ENDM  INTENB
127          ;
128          ;
129          .MACRO  TSTEND  A,?L
130          TST     MSGFLG          ;PRINTOUT OR NOT?
131          BNE     L
132          WRITE   MTE'A;          ;TEST'A END MESSAGE
133          RETURN;                  ;RETURN TO CALLER
134          .ENDM  TSTEND
135          ;
136          .MACRO  SIC
137          BIS     @BIT6,@SMR      ;SEND INTERFACE CLEAR
138          .ENDM  SIC
139          ;
140          .MACRO  SAVCIR
141          MOV     @CIR,R0          ;SAVE CIR CONTENT
142          .ENDM  SAVCIR
143          ;
144          .MACRO  SAVSMR
145          MOV     @SMR,R1          ;SAVE SMR CONTENT
146          .ENDM  SAVSMR
147          ;
148          .MACRO  SAVIOR
149          MOV     @IOR,R2          ;SAVE IOR CONTENT
150          .ENDM  SAVIOR
151          ;
152          .MACRO  GTS
153          BIS     @BIT12,@SMR     ;GO TO STAND BY
154          .ENDM  GTS
155          ;
156          .MACRO  LTN
157          SET4    @SMR
158          .ENDM  LTN
159          ;
160          .MACRO  LUN
161          SET5    @SMR
162          .ENDM  LUN
163          ;

```

```

164      .MACRO SRE
165      SET7 @SMR      ;SEND REMOTE ENABLE
166      .ENDM SRE
167      ;
168      .MACRO TCA
169      BIS @BIT1,@SMR ;TAKE CONTROL ASYNC
170      .ENDM TCA
171      ;
172      .MACRO TCS
173      SET0 @SMR      ;TAKE CONTROL SYNCR.
174      .ENDM TCS
175      ;
176      .MACRO SAVIEC
177      SAVCIR
178      SAVSMR
179      SAVIOR
180      .ENDM SAVIEC
181      ;
182      .MACRO LOOP      A,B,?C
183      DEC (PC)+      ;COUNT LOOPS
184      .WORD A      ;
185      IF C NE #0 GOTO B
186      MOV @A,C      ;SETUP NEW LOOP COUNT
187      .ENDM LOOP
188
189
190
191
192 011672 013746 011734  RANDOM: MOV RA, (SP)
193 011676 013700 011736  MOV RB,R0
194 011702 006316  ASL @SP
195 011704 005500  ADC R0
196 011706 006200  ASR R0
197 011710 005516  ADC @SP
198 011712 061600  ADD @SP,R0
199 011714 005616  SBC @SP
200 011716 060016  ADD R0,@SP
201 011720 005600  SBC R0
202 011722 012637 011734  MOV (SP)+,RA
203 011726 010037 011736  MOV R0,RB
204 011732 000207  RETURN
205 011734 135753  RA: .WORD 135753
206 011736 024642  RB: .WORD 024642
207      ;

```

					TEST0:	SBTTL	TEST LON SWITCH FEATURE	
209						SCOPE		
210	011740	104000				WRITE	SLON	; SET LON SW MESSAGE ; **V02E00**
211	011742					MOV	#SLON, (SP)	; SAVE #SLON ONTO STACK ; **V02E00**
	011742	012746	022112			CALL	\$.WRT	; INITIATE PRINTOUT
	011746	004737	005654			READ	OCT,RO	
212	011752					CLR	-(SP)	; INDICATE OCTAL INPUT REQUIRED
	011752	005046				CALL	\$.RED	; ACCEPT INPUT
	011754	004737	005020			MCLEAR		; PRODUCE CLEAN STATUS ; **V02E00**
213	011760					BIS	#BITS, #CIR	; EXECUTE MASTER CLEAR
	011760	052777	000040	012216		CMP	#CIR, #602	; STATE CHGE, INT, LON ; **V02E00**
214	011766	027727	012212	000602		BEQ	1\$	; BRANCH IF CORRECT ; **V02E00**
215	011774	001404				ERROR	114.	; CIR CONT NOT #602 ; **V02E00**
216	011776					TRAP	114.	; ERROR 114.
	011776	104562				.WORD	0	; ERROR COUNT
	012000	000000				JMP	TEST0	; **V02E00**
217	012002	000137	011740			CMP	#SMR, #BIT12	; LACS SET EXCLUSIVELY? ; **V02E00**
218	012006	027727	012174	010000	1\$:	BEQ	2\$	; BRANCH IF O.K. ; **V02E00**
219	012014	001404				ERROR	115.	; LACS NOT OR NOT ONLY ; **V02E00**
220	012016					TRAP	115.	; ERROR 115.
	012016	104563				.WORD	0	; ERROR COUNT
	012020	000000				JMP	TEST0	; **V02E00**
221	012022	000137	011740			WRITE	CLON	; RESET LON SW MSG ; **V02E00**
222	012026				2\$:	MOV	#CLON, (SP)	; SAVE #CLON ONTO STACK
	012026	012746	022216			CALL	\$.WRT	; INITIATE PRINTOUT
	012032	004737	005654			READ	OCT,RO	
223	012036					CLR	-(SP)	; INDICATE OCTAL INPUT REQUIRED
	012036	005046				CALL	\$.RED	; ACCEPT INPUT
	012040	004737	005020			CMP	#600, #CIR	; STATE CHGE, INT ; **V02E00**
224	012044	022777	000600	012132		BEQ	3\$	; BRANCH IF O.K. ; **V02E00**
225	012052	001404				ERROR	116.	; CIR CONTENT NOT #600 ; **V02E00**
226	012054					TRAP	116.	; ERROR 116.
	012054	104564				.WORD	0	; ERROR COUNT
	012056	000000				JMP	TEST0	; **V02E00**
227	012060	000137	011740			CMP	#SMR, #BIT12	; LACS STILL SET? ; **V02E00**
228	012064	027727	012116	010000	3\$:	BEQ	4\$	; BRANCH IF YES ; **V02E00**
229	012072	001404				ERROR	117.	; ERROR IF NO ; **V02E00**
230	012074					TRAP	117.	; ERROR 117.
	012074	104565				.WORD	0	; ERROR COUNT
	012076	000000				JMP	TEST0	; **V02E00**
231	012100	000137	011740			RETURN		; RETURN FROM SUBTEST ; **V02E00**
232	012104	000207			4\$:			

```

234 .S8TTL TEST 1 EXECUTE MASTER CLEAR
235 012106 005737 011670 TEST1: TST MSGFLG ;PRINTOUT OR NOT?
236 012112 001004 BNE LOPT1
237 012114 012746 022324 WRITE MT1S ;TEST1 START MESSAGE
      012120 004737 005654 MOV #MT1S, (SP) ;SAVE #MT1S ONTO STACK
      012124 104000 CALL $.WRT ;INITIATE PRINTOUT
238 012124 104000 LOPT1: SCOPE ;SCOPE LOOP MARKER
239 012126 052777 000040 012050 MCLEAR
      012126 017700 012046 BIS #BIT5, @CIR ;EXECUTE MASTER CLEAR
240 012134 017700 012046 MOV @SMR, RO ;SAVE SMR CONTENT
241 012140 001404 BEQ 1$ ;IF EQ. YES
242 012142 104401 ERROR 1. ;ILLEGAL SMR AFTER MC
      012142 104401 TRAP 1. ;ERROR 1.
      012144 000000 .WORD 0 ;ERROR COUNT
243 012146 000137 012106 JMP TEST1 ;
244 012152 017700 012026 1$: SAVCIR
      012152 017700 012026 MOV @CIR, RO ;SAVE CIR CONTENT
245 012156 001404 BEQ 2$ ;IF EQ. NO BIT SET
246 012160 104402 ERROR 2. ;ILLEGAL CIR AFTER MC
      012160 104402 TRAP 2. ;ERROR 2.
      012162 000000 .WORD 0 ;ERROR COUNT
247 012164 000137 012106 JMP TEST1 ;
248 012170 017700 012016 2$: MOV @IOR, RO ;SAVE IOR CONTENT
249 012174 001404 BEQ 3$ ;IF EQ. NO BIT SET
250 012176 104403 ERROR 3. ;ILLEGAL IOR AFTER MC
      012176 104403 TRAP 3. ;ERROR 3.
      012200 000000 .WORD 0 ;ERROR COUNT
251 012202 000137 012106 JMP TEST1 ;
252 012206 005327 012210 3$: LOOP 1, LOPT1 ;COUNT LOOPS
      012206 177777 DEC (PC). ;
      012210 005737 012210 64$: TST 64$ ;IS 64$ ZERO ?
      012212 001401 BEQ .+4 ;
      012216 000741 BR LOPT1 ;
      012222 012737 177777 012210 MOV #1, 64$ ;SETUP NEW LOOP COUNT
253 012230 005737 011670 TSTEND 1
      012230 001004 TST MSGFLG ;PRINTOUT OR NOT?
      012234 012746 022371 BNE 66$
      012236 004737 005654 MOV #MTE1, (SP) ;SAVE #MTE1 ONTO STACK
      012242 000207 CALL $.WRT ;INITIATE PRINTOUT
      012246 000207 66$: RETURN ;RETURN TO CALLER

```

255					.SBTIL	TEST 2	TEST SACS BIT IN CIR
256	012250	005737	011670		TEST2: TST	MSGFLG	PRINTOUT OR NOT?
257	012254	001004			BNE	LOPT2	
258	012256				WRITE	MT25	TEST2 START MESSAGE
	012256	012746	022405		MOV	#MT25, (SP)	SAVE #MT25 ONTO STACK
	012262	004737	005654		CALL	1..WRT	INITIATE PRINTOUT
259	012266				LOPT2: MCLEAR		PRODUCE CLEAR CONTROLLER STATE
	012266	052777	000040	011710	BIS	#BIT5, BCIR	EXECUTE MASTER CLEAR
260	012274	104000			SCOPE		SCOPE LOOP MARKER
261	012276				SACS		
	012276	052777	000001	011700	BIS	#BIT0, BCIR	SET SACS BIT IN CIR
262	012304				SAVCIR		
	012304	017700	011674		MOV	BCIR, R0	SAVE CIR CONTENT
263	012310				TST	R0	
	012310	032700	000001		BIT	#BIT0, R0	
264	012314	001004			BNE	11	IF NE, SET
265	012316				ERROR	4.	SACS BIT NOT SET
	012316	104404			TRAP	4.	ERROR 4.
	012320	000000			.WORD	0	ERROR COUNT
266	012322	000137	012250		JMP	TEST2	
267	012326	104000			11: SCOPE		SCOPE LOOP MARKER
268	012330				CLRO	BCIR	TRY TO RESET SACS BIT
	012330	042777	000001	011646	BIC	#BIT0, BCIR	
269	012336				SAVCIR		
	012336	017700	011642		MOV	BCIR, R0	SAVE CIR CONTENT
270	012342				TST	R0	SACS RESET ?
	012342	032700	000001		BIT	#BIT0, R0	
271	012346	001404			BEQ	21	IF EQ, YES
272	012350				ERROR	5.	SACS NOT RESET BY BIC
	012350	104405			TRAP	5.	ERROR 5.
	012352	000000			.WORD	0	ERROR COUNT
273	012354	000137	012250		JMP	TEST2	
274	012360	104000			21: SCOPE		SCOPE LOOP MARKER
275	012362				SACS		
	012362	052777	000001	011614	BIS	#BIT0, BCIR	SET SACS BIT IN CIR
276	012370				MCLEAR		
	012370	052777	000040	011606	BIS	#BIT5, BCIR	EXECUTE MASTER CLEAR
277	012376				SAVCIR		IS SACS CLEAR AFTER MC ?
	012376	017700	011602		MOV	BCIR, R0	SAVE CIR CONTENT
278	012402	001404			BEQ	31	IF EQ, YES
279	012404				ERROR	6.	SACS NOT RESET BY MC
	012404	104406			TRAP	6.	ERROR 6.
	012406	000000			.WORD	0	ERROR COUNT
280	012410	000137	012250		JMP	TEST2	
281	012414				31: LOOP	1, LOPT2	
	012414	005327			DEC	(PC).	COUNT LOOPS
	012416	177777			641: .WORD	1	
	012420	005737	012416		TST	641	IS 641 ZERO ?
	012424	001401			BEQ	..4	
	012426	000717			BR	LOPT2	
	012430	012737	177777	012416	MOV	#-1, 641	SETUP NEW LOOP COUNT
282	012436				TSTEND	2	
	012436	005737	011670		TST	MSGFLG	PRINTOUT OR NOT?
	012442	001004			BNE	661	
	012444	012746	022452		MOV	#MTE2, (SP)	SAVE #MTE2 ONTO STACK
	012450	004737	005654		CALL	1..WRT	INITIATE PRINTOUT
	012454	000207			661: RETURN;		RETURN TO CALLER

```

284 012456 005737 011670
285 012462 001004
286 012464
287 012464 012746 022466
012470 004737 005654
288 012474 052777 000040 011502
012474 104000
289 012502 052777 002000 011472
290 012504 032777 002000 011464
012512 001004
291 012512 104551
012520 000000
292 012522 000137 012456
012532 042777 002000 011444
293 012532 032777 002000 011436
012540 001404
294 012540 104552
012550 000000
295 012552 000137 012456
012560 104000
296 012562 052777 002000 011414
012570 032777 002000 011406
297 012576 001004
300 012600 104553
012602 000000
301 012604 000137 012456
012610 052777 000040 011366
302 012616 032777 002000 011360
012624 001404
303 012626 104556
012630 000000
304 012632 000137 012456
012636 005327
012640 177777
012642 005737 012640
012646 001401
012650 000711
012652 012737 177777 012640
310 012660 005737 011670
012660 001004
012664 012746 022554

```

TEST3: SBTTL TEST 3 TEST CIR BIT 10 INHIBIT SRQ INTERRUPT
 TST MSGFLG ;PRINTOUT OR NOT?
 BNE LOPT2A
 WRITE MT2AS ;GIVE TEST3 START MSG ; **V02E00**
 MOV #MT2AS, -(SP) ;SAVE #MT2AS ONTO STACK
 CALL \$. WPT ;INITIATE PRINTOUT
 LOPT2A: MCLEAR ;PRODUCE CLEAR CONTR STATE ; **V02E00**
 BIS #BIT5, @CIR ;EXECUTE MASTER CLEAR
 SCOPE ;SCOPE LOOP ENTRY ; **V02E00**
 INMSRQ ;SET SRQ INT INHIBIT ; **V02E00**
 BIS #BIT10, @CIR ;SET INHIBIT SRQ INTERRUPT
 TST10 @CIR ;IS BIT 10 IN CIR SET? ; **V02E00**
 BIT #BIT10, @CIR
 BNE 1: ;BRANCH IF SET ; **V02E00**
 ERROR 105. ;BIT 10 IN CIR NOT SET ; **V02E00**
 TRAP 105. ;ERROR 105.
 .WORD 0 ;ERROR COUNT
 JMP TEST3
 1: CLR10 @CIR ;RESET BIT 10 IN CIR ; **V02E00**
 BIC #BIT10, @CIR ; **V02E00**
 TST10 @CIR ;CLEAR AGAIN ? ; **V02E00**
 BIT #BIT10, @CIR
 BEQ 2: ;BRANCH IF YES ; **V02E00**
 ERROR 106. ;BIT 10 NOT CLEAR ; **V02E00**
 TRAP 106. ;ERROR 106.
 .WORD 0 ;ERROR COUNT
 JMP TEST3
 2: SCOPE ; **V02E00**
 INMSRQ ;SET SRQ INT INHIBIT ; **V02E00**
 BIS #BIT10, @CIR ;SET INHIBIT SRQ INTERRUPT
 TST10 @CIR ;LOOK IF SET AGAIN ; **V02E00**
 BIT #BIT10, @CIR
 BNE 3: ;BRANCH IF O.K. ; **V02E00**
 ERROR 107. ;BIT 10 IN CIR NOT SET AGAIN ; **V02E00**
 TRAP 107. ;ERROR 107.
 .WORD 0 ;ERROR COUNT
 JMP TEST3
 3: MCLEAR ;RESET BIT10 IN CIR BY MC ; **V02E00**
 BIS #BIT5, @CIR ;EXECUTE MASTER CLEAR
 TST10 @CIR ;LOOK IF DONE ; **V02E00**
 BIT #BIT10, @CIR
 BEQ 4: ;BRANCH IF O.K. ; **V02E00**
 ERROR 110. ;BIT10 IN CIR NOT CLEAR ; **V02E00**
 TRAP 110. ;ERROR 110.
 .WORD 0 ;ERROR COUNT
 JMP TEST3
 4: LOOP 1, LOPT2A ;LOOP IN STEP? ; **V02E00**
 DEC (PC). ;COUNT LOOPS
 .WORD 1
 64: TST 64: ;IS 64: ZERO ?
 BEQ .+4
 BR LOPT2A
 MOV #1, 64: ;SETUP NEW LOOP COUNT
 TSTEND 2A ;RETURN TO CALLER
 TST MSGFLG ;PRINTOUT OR NOT?
 BNE 66: ; **V02E00**
 MOV #MTE2A, (SP) ;SAVE #MTE2A ONTO STACK

D6

TEST 1: A DIAGNOSTIC MACRO M1200 30 MAR 84 16:02 PAGE 36 1
TEST 5: TEST CIR BIT 10 INHIBIT SRQ INTERRUPT

SEQ 68

012672 004737 005654
012676 000207

661: CALL 1..WRT
RETURN;

INITIATE PRINTOUT
RETURN TO CALLER

314					.SBTTL	TEST 4	TEST SACS SIC SIAS ILLMSG CACS
315	012700	005737	011670		TEST4: TST	MSGFLG	;PRINTOUT OR NOT?
316	012704	001004			BNE	LOPT3	
317	012706				WRITE	MT35	;START MESSAGE TEST 4
	012706	012746	022570		MOV	#MT35, (SP)	;SAVE #MT35 ONTO STACK
	012712	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
318	012716				LOPT3: MCLEAR		
	012716	052777	000040	011260	BIS	#BIT5, @CIR	;EXECUTE MASTER CLEAR
319	012724				SACS		
	012724	052777	000001	011252	BIS	#BIT0, @CIR	;SET SACS BIT IN CIR
320	012732	104000			SCOPE		;SCOPE LOOP MARKER
321	012734				SIC		;SEND INTERFACE CLEAR
	012734	052777	000100	011244	BIS	#BIT6, @SMR	;SEND INTERFACE CLEAR
322	012742	000240			NOP		;WAIT SOME MICRO SECONDS
323	012744	000240			NOP		
324	012746	000240			NOP		
325	012750				SAVCIR		
	012750	017700	011230		MOV	@CIR, R0	;SAVE CIR CONTENT
326	012754				SAVSMR		;SAVE SMR CONTENT
	012754	017701	011226		MOV	@SMR, R1	;SAVE SMR CONTENT
327	012760				CLR8	@CIR	;RESET "STATE CHANGE" BIT
	012760	042777	000400	011216	BIC	#BIT8, @CIR	
328	012766				TST12	R0	;ILLEGAL MESSAGE ?
	012766	032700	010000		BIT	#BIT12, R0	
329	012772	001404			BEQ	18	;IF EQ, NO
330	012774				ERROR	7.	;ILLEGAL MESSAGE
	012774	104407			TRAP	7.	;ERROR 7.
	012776	000000			.WORD	0	;ERROR COUNT
331	013000	000137	012700		JMP	TEST4	
332	013004				18: TST13	R1	;SIAS SET ?
	013004	032701	020000		BIT	#BIT13, R1	
333	013010	001004			BNE	28	;IF NE, YES
334	013012				ERROR	10.	;SIAS NOT SET
	013012	104412			TRAP	10.	;ERROR 10.
	013014	000000			.WORD	0	;ERROR COUNT
335	013016	000137	012700		JMP	TEST4	
336	013022				28: TST8	R0	;STATE CHANGE ?
	013022	032700	000400		BIT	#BIT8, R0	
337	013026	001004			BNE	38	;IF NE, YES
338	013030				ERROR	11.	;NO STATE CHANGE TO SIAS
	013030	104413			TRAP	11.	;ERROR 11.
	013032	000000			.WORD	0	;ERROR COUNT
339	013034	000137	012700		JMP	TEST4	
340	013040				38: TST8	R1	;CACS SET ?
	013040	032701	000400		BIT	#BIT8, R1	
341	013044	001004			BNE	108	;IF NE, YES
342	013046				ERROR	12.	;CACS NOT SET BY SIAS
	013046	104414			TRAP	12.	;ERROR 12.
	013050	000000			.WORD	0	;ERROR COUNT
343	013052	000137	012700		JMP	TEST4	
344	013056	005037	013074		108: CLR	118	
345	013062				48: TST13	@SMR	;SIAS RESET ?
	013062	032777	020000	011116	BIT	#BIT13, @SMR	
346	013070	001407			BEQ	58	;IF EQ, YES
347	013072	005227			INC (PC).		;WAIT A WHILE
348	013074	000000			118: .WORD	0	
349	013076	001371			BNE	48	

```

350 013100          ERROR 13.          ;SIAS NOT RESET
    013100 104415   TRAP 13.          ;ERROR 13.
    013102 000000   .WORD 0           ;ERROR COUNT
351 013104 000137 012700          ;
352 013110          5$: SAVCIR          ;
    013110 017700 011070   MOV @CIR,R0 ;SAVE CIR CONTENT
353 013114          SAVSMR          ;SAVE SMR CONTENT
    013114 017701 011066   MOV @SMR,R1 ;SAVE SMR CONTENT
354 013120          TST8 R0           ;STATE CHANGE ?
    013120 032700 000400   BIT @BIT8,R0
355 013124 001004          BNE 6$      ;IF NE, YES
356 013126          ERROR 14.        ;NO STATE CHANGE TO SIAS n' SET
    013126 104416   TRAP 14.        ;ERROR 14.
    013130 000000   .WORD 0         ;ERROR COUNT
357 013132 000137 012700          ;
358 013136          6$: TST8 R1        ;CACS STILL SET ?
    013136 032701 000400   BIT @BIT8,R1
359 013142 001004          BNE 7$      ;IF NE, YES
360 013144          ERROR 15.        ;CACS RESET WITH SIAS RESET
    013144 104417   TRAP 15.        ;ERROR 15.
    013146 000000   .WORD 0         ;ERROR COUNT
361 013150 000137 012700          ;
362 013154 104000          7$: SCOPE    ;SCOPE LOOP MARKER
363 013156          MCLEAR          ;EXECUTE MASTER CLEAR
    013156 052777 000040 011020   BIS @BIT5,@CIR ;SAVE NEW SMR CONTENT
364 013164          SAVSMR          ;SAVE SMR CONTENT
    013164 017701 011016   MOV @SMR,R1 ;CACS RESET ?
365 013170          TST8 R1
    013170 032701 000400   BIT @BIT8,R1
366 013174 001404          BEQ 8$      ;IF EQ, YES
367 013176          ERROR 16.        ;CACS NOT RESET WITH MC
    013176 104420   TRAP 16.        ;ERROR 16.
    013200 000000   .WORD 0         ;ERROR COUNT
368 013202 000137 012700          ;
369 013206 104000          8$: SCOPE    ;SCOPE LOOP MARKER
370 013210          SIC
    013210 052777 000100 010770   BIS @BIT6,@SMR ;SEND INTERFACE CLEAR
371 013216          SAVCIR          ;SAVE CIR CONTENT
    013216 017700 010762   MOV @CIR,R0 ;IS 'ILL. MESS' SET ?
372 013222          TST12 R0
    013222 032700 010000   BIT @BIT12,R0
373 013226 001004          BNE 9$      ;IF NE, YES
374 013230          ERROR 17.        ;NO ILL MESS TO SICS W/O SACS
    013230 104421   TRAP 17.        ;ERROR 17.
    013232 000000   .WORD 0         ;ERROR COUNT
375 013234 000137 012700          ;
376 013240          9$: LOOP -1,LOPT3  ;COUNT LOOPS
    013240 005327   DEC (PC).
    013242 177777   .WORD 1
    013244 005737 013242   TST 64$    ;IS 64$ ZERO ?
    013250 001401          BEQ .+4
    013252 000621          BR LOPT3
    013254 012737 177777 013242   MOV @-1,64$ ;SETUP NEW LOOP COUNT
377 013262          TSTEND 3
    013262 005737 011670   TST MSGFLG ;PRINTOUT OR NOT?
    013266 001004          BNE 66$
    013270 012746 022647   MOV @MTE3,(SP) ;SAVE @MTE3 ONTO STACK

```

G6

IEC11 A DIAGNOSTIC MACRO M1200 30 MAR 84 16:02 PAGE 37-2
TEST 4 TEST SACS SIC SIAS ILLMSG CACS

SEQ 71

013274 004737 005654
013300 000207

668: CALL \$..WRT
RETURN;

;INITIATE PRINTOUT
;RETURN TO CALLER

```

379          .SBTTL TEST 5 TEST INT ENB
380 013302 005737 011670      TEST5:  TST MSGFLG      ;PRINTOUT OR NOT?
381 013306 001004              BNE LOPT3A
382 013310              WRITE MT3AS      ;START MESSAGE TEST 5
      013310 012746 022663      MOV #MT3AS, -(SP)      ;SAVE #MT3AS ONTO STACK
      013314 004737 005654      CALL $..WRT      ;INITIATE PRINTOUT
383 013320              LOPT3A: MCLAR      ;MASTER CLEAR
      013320 052777 000040 010656  BIS #BIT5, @CIR      ;EXECUTE MASTER CLEAR
384 013326 013701 024204      MOV CIR, R1      ;GET CIR ADDRESS
385 013332 104000              SCOPE      ;SCOPE LOOP MARKER
386 013334              SET6 @R1      ;TRY TO SET INTERRUPT ENABLE
      013334 052711 000100      BIS #BIT6, @R1
387 013340              TST6 @R1      ;IS INT ENBL SET ?
      013340 032711 000100      BIT #BIT6, @R1
388 013344 001004              BNE 1$      ;IF NE, YES
389 013346 011100              MOV @R1, R0      ;SAVE CIR
390 013350              ERROR 33.      ;INTERRUPT ENBALE NOT SET
      013350 104441              TRAP 33.      ;ERROR 33.
      013352 000000              .WORD 0      ;ERROR COUNT
391 013354 000752              BR TEST5
392 013356 104000              1$: SCOPE      ;SCOPE LOOP MARKER
393 013360              CLR6 @R1      ;TRY TO RESET INTERRUPT ENABLE
      013360 042711 000100      BIC #BIT6, @R1
394 013364              TST6 @R1      ;IS INT ENABLE RESET ?
      013364 032711 000100      BIT #BIT6, @R1
395 013370 001404              BEQ 2$      ;IF EQ, YES
396 013372 011100              MOV @R1, R0      ;SAVE CIR
397 013374              ERROR 34.      ;INTERRUPT ENABLE NOT RESET
      013374 104442              TRAP 34.      ;ERROR 34.
      013376 000000              .WORD 0      ;ERROR COUNT
398 013400 000740              BR TEST5
399 013402              2$: LOOP -1, LOPT3A
      013402 005327              DEC (PC)+      ;COUNT LOOPS
      013404 177777              64$: .WORD -1
      013406 005737 013404      TST 64$      ;IS 64$ ZERO ?
      013412 001401              BEQ .+4
      013414 000741              BR LOPT3A
      013416 012737 177777 013404  MOV #-1, 64$      ;SETUP NEW LOOP COUNT
400 013424              TSTEND 3A      ;RETURN TO CALLER
      013424 005737 011670      TST MSGFLG      ;PRINTOUT OR NOT?
      013430 001004              BNE 66$
      013432 012746 022720      MOV #MTE3A, -(SP)      ;SAVE #MTE3A ONTO STACK
      013436 004737 005654      CALL $..WRT      ;INITIATE PRINTOUT
      013442 000207              66$: RETURN;      ;RETURN TO CALLER

```

```

402          .SBTTL TEST 6 TEST INTERRUPT WITH STATE CMGE
403 013444 005737 011670 TEST6: TST MSGFLG ;PRINTOUT OR NOT?
404 013450 001004 BNE 101
405 013452 012746 022734 WRITE MT45 ;TEST 6 START MESSAGE
      013452 004737 005654 MOV #MT45, (SP) ;SAVE #MT45 ONTO STACK
      013456 004737 005654 CALL $..WRT ;INITIATE PRINTOUT
406 013462 012746 024222 101: SETVEC #VECTOR, #IECINT, #0 ;SET VECTOR ; **V03E02**
      013462 012746 021334 MOV #VECTOR, (SP) ;SAVE #VECTOR ONTO STACK
      013472 012746 000000 MOV #IECINT, -(SP) ;SAVE #IECINT ONTO STACK
      013476 104001 EMT 1 ;SAVE #0 ONTO STACK
      013500 104000 LOPT4: SCOPE 1 ;SETUP #VECTOR VECTOR
      013502 052777 000040 010474 MCLEAR ;SCOPE LOOP MARKER
      013510 012705 010000 BIS #BIT5, #CIR ;EXECUTE MASTER CLEAR
      013514 012704 013574 MOV #BIT12, R5 ;SETUP INTERRUPT MASK
      013520 052777 000100 010456 MOV #31, R4 ;SETUP RETURN ADDRESS
      013526 105037 177776 BIS #BIT6, #CIR ;ENABLE INTERRUPT IN IEC-11A
      013532 052777 000100 010446 CLR #PS ;ENABLE INTERRUPTS
      013532 005037 013546 BIS #BIT6, #SMR ;SEND INTERFACE CLEAR
      013544 005227 11: INC 21 ;WAIT FOR INTERRUPT
      013546 000000 21: .WORD 0
      013550 001375 BNE 11
      013552 017700 010426 SAVIEC MOV #CIR, R0 ;SAVE CIR CONTENT
      013556 017701 010424 MOV #SMR, R1 ;SAVE SMR CONTENT
      013562 017702 010424 MOV #IOR, R2 ;SAVE IOR CONTENT
419 013566 104424 ERROR 20. ;NO INTERRUPT TO SIC
      013566 000000 TRAP 20. ;ERROR 20.
      013570 000724 .WORD 0 ;ERROR COUNT
420 013572 103003 BR TEST6
421 013574 104425 31: BCC 41 ;IF CC, INTERRUPT OK
      013576 000000 ERROR 21. ;ILLEGAL INTERRUPT
      013576 000000 TRAP 21. ;ERROR 21.
      013600 000720 .WORD 0 ;ERROR COUNT
423 013602 104000 BR TEST6
424 013604 052777 000040 010370 41: SCOPE ;SCOPE LOOP MARKER
      013606 012705 000400 MCLEAR BIS #BIT5, #CIR ;EXECUTE MASTER CLEAR
      013614 012704 013702 MOV #BIT8, R5 ;SETUP INTERRUPT MASK
      013624 052777 000100 010352 MOV #71, R4 ;SETUP INTERRUPT RETURN ADDRESS
      013632 052777 000001 010344 BIS #BIT6, #CIR ;ENABLE INTERRUPT IN IEC 11A
      013640 052777 000100 010340 SACS BIS #BIT0, #CIR ;SET SACS BIT IN CIR
      013646 005037 013654 SIC BIS #BIT6, #SMR ;SEND INTERFACE CLEAR
      013652 005227 51: CLR 61 ;WAIT FOR INTERRUPT
      013654 000000 61: .WORD 0
      013656 001375 BNE 51
435 013660 017700 010320 SAVIEC ;SAVE ALL IEC REGISTERS
      013664 017701 010316 MOV #CIR, R0 ;SAVE CIR CONTENT
      013670 017702 010316 MOV #SMR, R1 ;SAVE SMR CONTENT
      MOV #IOR, R2 ;SAVE IOR CONTENT

```

J6

SEQ 74

436	013674			ERROR	22.				
	013674	104426		TRAP	22.				;NO INTERRUPT TO SIC AND SACS
	013676	000000		.WORD	0				;ERROR 22.
437	013700	000661		BR	TEST6				;ERROR COUNT
438	013702	103003		BCC	8:				;IF CC. INTERRUPT OK
439	013704			ERROR	23.				;ILLEGAL INTERRUPT
	013704	104427		TRAP	23.				;ERROR 23.
	013706	000000		.WORD	0				;ERROR COUNT
440	013710	000655		BR	TEST6				
441	013712			LOOP	1,LOPT4				
	013712	005327		DEC	(PC).				;COUNT LOOPS
	013714	177777		.WORD	1				
	013716	005737	013714	TST	64:				;IS 64: ZERO ?
	013722	001401		BEQ	.+4				
	013724	000665		BR	LOPT4				
	013726	012737	177777 013714	MOV	# 1,64:				;SETUP NEW LOOP COUNT
442	013734			TSTEND	4				
	013734	005737	011670	TST	MSGFLG				;PRINTOUT OR NOT?
	013740	001004		BNE	66:				
	013742	012746	023013	MOV	#MTE4, (SP)				;SAVE #MTE4 ONTO STACK
	013746	004737	005654	CALL	\$.WRT				;INITIATE PRINTOUT
	013752	000207		66:	RETURN;				;RETURN TO CALLER

```

444 .SBTTL TEST 7 TEST STATE CHGE INTERRUPT WITH SRAS
445 013754 005737 011670 TEST7: TST MSGFLG ;PRINTOUT OR NOT?
446 013760 001004 BNE LOPT5
447 013762 WRITE MTSS ;START MESSAGE TEST 7
      013762 012746 023027 MOV #MTSS, (SP) ;SAVE #MTSS ONTO STACK
      013766 004737 005654 CALL $.WRT ;INITIATE PRINTOUT
448 013772 LOPT5: MCLEAR ;MASTER CLEAR
      013772 052777 000040 010204 BIS #BIT5,@CIR ;EXECUTE MASTER CLEAR
449 014000 SACS ;SET CONTROLLER ACTIVE STATE
      014000 052777 000001 010176 BIS #BIT0,@CIR ;SET SACS BIT IN CIR
450 014006 104000 SCOPE ;SCOPE LOOP MARKER
451 014010 SRE ;SEND REMOTE ENABLE
      014010 052777 000200 010170 BIS #BIT7,@SMR
452 014016 105777 010164 TSTB @SMR ;SRE SET ?
453 014022 100407 BMI 5$ ;IF MI, YES
454 014024 SAVCIR
      014024 017700 010154 MOV @CIR,R0 ;SAVE CIR CONTENT
455 014030 SAVSMR
      014030 017701 010152 MOV @SMR,R1 ;SAVE SMR CONTENT
456 014034 ERROR 24. ;SRE NOT SET
      014034 104430 TRAP 24. ;ERROR 24.
      014036 000000 .WORD 0 ;ERROR COUNT
457 014040 000745 BR TEST7 ;
458 014042 5$: CLR @CIR ;RESET STATE CHANGE
      014042 042777 000400 010134 BIC #BIT8,@CIR
459 014050 TST12 @CIR ;ILLEGAL MESSAGE ?
      014050 032777 010000 010126 BIT #BIT12,@CIR
460 014056 001407 BEQ 1$ ;IE EQ, NO
461 014060 SAVCIR
      014060 017700 010120 MOV @CIR,R0 ;SAVE CIR CONTENT
462 014064 SAVSMR
      014064 017701 010116 MOV @SMR,R1 ;SAVE SMR CONTENT
463 014070 ERROR 25. ;ILLEGAL MESSAGE
      014070 104431 TRAP 25. ;ERROR 25.
      014072 000000 .WORD 0 ;ERROR COUNT
464 014074 000727 BR TEST7 ;
465 014076 005037 014114 1$: CLR 9$ ;WAIT A SHORT TIME
466 014102 10$: TST14 @SMR ;SRAS SET ?
      014102 032777 040000 010076 BIT #BIT14,@SMR
467 014110 001014 BNE 2$ ;IF NE, YES
468 014112 005227 INC (PC).
469 014114 000000 9$: .WORD 0
470 014116 001371 BNE 10$
471 014120 SAVIEC ;SAVE ALL IEC REGISTERS
      014120 017700 010060 MOV @CIR,R0 ;SAVE CIR CONTENT
      014124 017701 010056 MOV @SMR,R1 ;SAVE SMR CONTENT
      014130 017702 010056 MOV @IOR,R2 ;SAVE IOR CONTENT
472 014134 ERROR 26. ;SRAS NOT SET
      014134 104432 TRAP 26. ;ERROR 26.
      014136 000000 .WORD 0 ;ERROR COUNT
473 014140 000705 BR TEST7 ;
474 014142 2$: TST @CIR ;STATE CHANGE ?
      014142 032777 000400 010034 BIT #BIT8,@CIR
475 014150 001003 BNE 3$ ;IF NE, YES
476 014152 ERROR 27. ;NO STATE CHANGE
      014152 104433 TRAP 27. ;ERROR 27.
      014154 000000 .WORD 0 ;ERROR COUNT

```

477	014156	000676			BR	TEST7		
478	014160			3\$:	CLR7	@SMR		; RESET SRE
	014160	042777	000200	010020	BIC	@BIT7,@SMR		
479	014166	000240			NOP			; WAIT SOME MICRO SECONDS
480	014170	000240			NOP			
481	014172	000240			NOP			
482	014174				SAVCIR			; SAVE CIR CONTENT
	014174	017700	010004		MOV	@CIR,R0		; SAVE CIR CONTENT
483	014200				SAVSMR			; SAVE SMR CONTENT
	014200	017701	010002		MOV	@SMR,R1		; SAVE SMR CONTENT
484	014204	105701			TSTB	R1		; SRE RESET ?
485	014206	100003			BPL	6\$		; IF PL, YES
486	014210				ERROR	30.		; SRE NOT RESET
	014210	104436			TRAP	30.		; ERROR 30.
	014212	000000			.WORD	0		; ERROR COUNT
487	014214	000657			BR	TEST7		
488	014216			6\$:	CLR8	@CIR		; RESET STATE CHANGE
	014216	042777	000400	007760	BIC	@BIT8,@CIR		
489	014224				TST14	R1		; SRAS RESET ?
	014224	032701	040000		BIT	@BIT14,R1		
490	014230	001403			BEQ	4\$		; IF EQ, YES
491	014232				ERROR	31.		; SRAS NOT RESET
	014232	104437			TRAP	31.		; ERROR 31.
	014234	000000			.WORD	0		; ERROR COUNT
492	014236	000646			BR	TEST7		
493	014240	000300		4\$:	SWAB	R0		
494	014242				TST8	R0		; STATE CHANGE SET ?
	014242	032700	000400		BIT	@BIT8,R0		
495	014246	001004			BNE	7\$		; IF NE, YES
496	014250	000300			SWAB	R0		; REBUILD ORIGINAL CONTENT
497	014252				ERROR	32.		; NO STATE CHANGE
	014252	104440			TRAP	32.		; ERROR 32.
	014254	000000			.WORD	0		; ERROR COUNT
498	014256	000636			BR	TEST7		
499	014260			7\$:	MCLEAR			; MASTER CLEAR
	014260	052777	000040	007716	BIS	@BIT5,@CIR		; EXECUTE MASTER CLEAR
500	014266	104000			SCOPE			; SCOPE LOOP MARKER
501	014270				SRE			; SEND REMOTE ENABLE
	014270	052777	000200	007710	BIS	@BIT7,@SMR		
502	014276	000240			NOP			; WAIT SOME MICRO SECONDS
503	014300	000240			NOP			
504	014302	000240			NOP			
505	014304				SAVCIR			; SAVE CIR CONTENT
	014304	017700	007674		MOV	@CIR,R0		; SAVE CIR CONTENT
506	014310				SAVSMR			; SAVE SMR CONTENT
	014310	017701	007672		MOV	@SMR,R1		; SAVE SMR CONTENT
507	014314				TST12	R0		; ILLEG. MESSAGE ?
	014314	032700	010000		BIT	@BIT12,R0		
508	014320	001003			BNE	8\$		; IF NE, YES
509	014322				ERROR	104.		; NO ILLEG. MESSAGE
	014322	104550			TRAP	104.		; ERROR 104.
	014324	000000			.WORD	0		; ERROR COUNT
510	014326	000612			BR	TEST7		
511	014330			8\$:	LOOP	1,LOPT5		
	014330	005327			DEC	(PC).		; COUNT LOOPS
	014332	177777		64\$:	.WORD	1		
	014334	005737	014332		TST	64\$		; IS 64\$ ZERO ?

IEC11 A DIAGNOSTIC MACRO M1200 30 MAR 84 16:02 PAGE 40 2
TEST 7 TEST STATE CHGE INTERRUPT WITH SRAS

SEQ 77

	014340	001401		BEQ	..4	
	014342	000613		BR	LOPT5	
	014344	012737	177777 014332	MOV	#-1,64\$	;SETUP NEW LOOP COUNT
512	014352			TSTEND	5	
	014352	005737	011670	IST	MSGFLG	;PRINTOUT OR NOT?
	014356	001004		BNE	66\$	
	014360	012746	023113	MOV	#MTES, -(SP)	;SAVE #MTES ONTO STACK
	014364	004737	005654	CALL	\$.WRT	;INITIATE PRINTOUT
	014370	000207	66\$:	RETURN;		;RETURN TO CALLER

				TEST 8	MSGFLG	TEST 8	IOR DATA TEST
514				.SBTTL			
515	014372	005737	011670	TEST8:	TST	MSGFLG	;PRINTOUT OR NOT?
516	014376	001004			BNE	108	
517	014400				WRITE	MTSAS	;START MESSAGE TEST 8
	014400	012746	023127		MOV	@MTSAS, -(SP)	;SAVE @MTSAS ONTO STACK
	014404	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
518	014410	104000		108:	SCOPE		
519	014412			LOPT5A:	MCLEAR		
	014412	052777	000040 007564		BIS	@B:75,BCIR	;EXECUTE MASTER CLEAR
520	014420	013704	024212		MOV	IOR,R4	;GET IOR ADDRESS
521	014424	005003			CLR	R3	;PREPARE DATA
522	014426	110314		18:	MOVB	R3,8R4	;WRITE PATTERN INTO IOR
523	014430	011402			MOV	8R4,R2	;READ BACK PATTERN FROM IOR
524	014432	020302			CMP	R3,R2	;PATTERN EQUAL ?
525	014434	001403			BEQ	28	;IF EQ, YES
526	014436				ERROR	47.	;DATA CHECK IN IOR
	014436	104457			TRAP	47.	;ERROR 47.
	014440	000000			.WORD	0	;ERROR COUNT
527	014442	000753			BR	TEST8	;RESTART TEST
528	014444	005203		28:	INC	R3	;BUILD NEXT PATTERN
529	014446	020327	000400		CMP	R3,#400	;ALL DONE ?
530	014452	001365			BNE	18	;IF NE, NO
531	014454				LOOP	1000.,LOPT5A	
	014454	005327			DEC	(PC).	;COUNT LOOPS
	014456	001750		648:	.WORD	1000.	
	014460	005737	014456		TST	(48	;IS 648 ZERO ?
	014464	001401			BEQ	.+4	
	014466	000751			BR	LOPT5A	
	014470	012737	001750 014456		MOV	@1000.,648	;SETUP NEW LOOP COUNT
532	014476				TSTEND	5A	;RETURN TO MAINPROGRAM
	014476	005737	011670		TST	MSGFLG	;PRINTOUT OR NOT?
	014502	001004			BNE	668	
	014504	012746	023165		MOV	@MTESA, -(SP)	;SAVE @MTESA ONTO STACK
	014510	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
	014514	000207		668:	RETURN;		;RETURN TO CALLER

```

534 .SBTTL TEST 9 TEST STATE CMGE INTERRUPT WITH GTS AND SIC
535 014516 TEST9: SETVEC #VECTOR,#IECINT,#0 ;SET VECTOR ;**V03E02**
014516 012746 024222 MOV #VECTOR,-(SP) ;SAVE #VECTOR ONTO STACK
014522 012746 021334 MOV #IECINT,-(SP) ;SAVE #IECINT ONTO STACK
014526 012746 000000 MOV #0,-(SP) ;SAVE #0 ONTO STACK
014532 104001 EMT 1 ;SETUP #VECTOR VECTOR
536 014534 005737 011670 TST MSGFLG ;PRINTOUT OR NOT?
537 014540 001004 BNE LOPT6
538 014542 WRITE MT6S ;START MESSAGE TEST 9
014542 012746 023201 MOV #MT6S,(SP) ;SAVE #MT6S ONTO STACK
014546 004737 005654 CALL B..WRT ;INITIATE PRINTOUT
539 014552 LOPT6: MCLEAR ;MASTER CLEAR
014552 052777 000040 007424 BIS #BIT5,#CIR ;EXECUTE MASTER CLEAR
540 014560 SACS ;SET CONTROLLER ACTIVE STATE
014560 052777 000001 007416 BIS #BIT0,#CIR ;SET SACS BIT IN CIR
541 014566 SIC ;SEND INTERFACE CLEAR
014566 052777 000100 007412 BIS #BIT6,#SMR ;SEND INTERFACE CLEAR
542 014574 013700 024204 MOV CIR,R0 ;GET CIR ADDRESS
543 014600 11: TSTB #R0 ;WAIT FOR STATE CHANGE
014600 032710 000400 BIT #BIT8,#R0
544 014604 001775 BEQ 11 ;PRODUCED BY SIAS
545 014606 104000 SCOPE ;SCOPE LOOP MARKER
546 014610 CLRB #R0 ;RESET STATE CHANGE
014610 042710 000400 BIC #BIT8,#R0
547 014614 TSTB #R0 ;WAIT FOR SIAS E... ;**V01E01**
014614 032710 000400 BIT #BIT8,#R0
548 014620 001775 BEQ .-4 ;
549 014622 CLRB #R0 ;RESET STATE CHANGE ;**V01E01**
014622 042710 000400 BIC #BIT8,#R0
550 014626 GTS ;GO TO STAND BY STATE
014626 052777 000004 007352 BIS #BIT2,#SMR ;GO TO STAND BY
551 014634 005037 014652 CLR 31 ;WAIT FOR CSBS
552 014640 21: TST9 #SMR ;
014640 032777 001000 007340 BIT #BIT9,#SMR ;
553 014646 001012 BNE 41 ;
554 014650 005227 INC (PC). ;
555 014652 000000 31: .WORD 0 ;
556 014654 001371 BNE 21 ;
557 014656 SAVCIR ;SAVE CIR CONTENT
014656 017700 007322 MOV #CIR,R0 ;SAVE CIR CONTENT
558 014662 SAVSMR ;SAVE SMR CONTENT
014662 017701 007320 MOV #SMR,R1 ;SAVE SMR CONTENT
559 014666 ERROR 35. ;NO CSBS
014666 104443 TRAP 35. ;ERROR 35.
014670 000000 .WORD 0 ;ERROR COUNT
560 014672 000711 BR TEST9 ;RESTART TEST
561 014674 41: TSTB #R0 ;DID CSBS BRING STATE CHANGE ?
014674 032710 000400 BIT #BIT8,#R0
562 014700 001007 BNE 51 ;IF NE, YES
563 014702 SAVCIR ;SAVE CIR CONTENT
014702 017700 007276 MOV #CIR,R0 ;SAVE CIR CONTENT
564 014706 SAVSMR ;SAVE SMR CONTENT
014706 017701 007274 MOV #SMR,R1 ;SAVE SMR CONTENT
565 014712 ERROR 36. ;NO STATE CHANGE TO CSBS
014712 104444 TRAP 36. ;ERROR 36.
014714 000000 .WORD 0 ;ERROR COUNT
566 014716 000677 BR TEST9 ;RESTART TEST

```

567	014720			58:	CLRB	8R0	RESET STATE CHANGE
	014720	042710	000400		BIC	8BIT8,8R0	
568	014724	104000			SCOPE		SCOPE LOOP MARKER
569	014726				GTS		FORCE ILLEGAL MESSAGE
	014726	052777	000004	007252	BIS	8BIT2,8SMR	GO TO STAND BY
570	014734	005037	014750		CLR	78	WAIT FOR ILLEGAL MESSAGE
571	014740			68:	TST12	8R0	
	014740	032710	010000		BIT	8BIT12,8R0	
572	014744	001012			BNE	88	
573	014746	005227			INC	(PC).	
574	014750	000000		78:	.WORD	0	
575	014752	001372			BNE	68	
576	014754				SAVCIR		SAVE CIR CONTENT
	014754	017700	007224		MOV	8CIR,R0	SAVE CIR CONTENT
577	014760				SAVSMR		SAVE SMR CONTENT
	014760	017701	007222		MOV	8SMR,R1	SAVE SMR CONTENT
578	014764				ERROR	37.	NO ILLEGAL MESSAGE
	014764	104445			TRAP	37.	ERROR 37.
	014766	000000			.WORD	0	ERROR COUNT
579	014770	000652			BR	TEST9	RESTART TEST9
580	014772	005200		88:	INC	R0	POINT TO CIR HIGH BYTE
581	014774	104000			SCOPE		SCOPE LOOP MARKER
582	014776	105010			CLRB	8R0	RESET INTERRUPT CONDITIONS
583	015000				TCA		TAKE CONTROL ASYNC
	015000	052777	000002	007200	BIS	8BIT1,8SMR	TAKE CONTROL ASYNC
584	015006	005037	015022		CLR	108	WAIT FOR CACS
585	015012	122710	000001	98:	CMPS	81,8R0	
586	015016	001413			BEQ	118	
587	015020	005227			INC	(PC).	
588	015022	000000		108:	.WORD	0	
589	015024	001372			BNE	98	
590	015026				SAVCIR		SAVE CIR CONTENT
	015026	017700	007152		MOV	8CIR,R0	SAVE CIR CONTENT
591	015032				SAVSMR		SAVE SMR CONTENT
	015032	017701	007150		MOV	8SMR,R1	SAVE SMR CONTENT
592	015036				ERROR	40.	TCA DOES NOT PRODUCE STATE CHANGE
	015036	104450			TRAP	40.	ERROR 40.
	015040	000000			.WORD	0	ERROR COUNT
593	015042	000137	014516		JMP TEST9		RESTART TEST9
594	015046	013701	024206	118:	MOV	8SMR,R1	GET ADDRESS OF SMR
595	015052	005201			INC	R1	ADDRESS HIGH BYTE OF SMR
596	015054	122711	000001		CMPS	81,8R1	CACS SET AGAIN ?
597	015060	001410			BEQ	128	IF EQ, YES
598	015062				SAVCIR		SAVE CIR CONTENT
	015062	017700	007116		MOV	8CIR,R0	SAVE CIR CONTENT
599	015066				SAVSMR		SAVE SMR CONTENT
	015066	017701	007114		MOV	8SMR,R1	SAVE SMR CONTENT
600	015072				ERROR	41.	TCA WITH CSBS DOES NOT PRODUCE CACS
	015072	104451			TRAP	41.	ERROR 41.
	015074	000000			.WORD	0	ERROR COUNT
601	015076	000137	014516		JMP	TEST9	RESTART TEST 9
602	015102	104000		128:	SCOPE		SCOPE LOOP MARKER
603	015104	105010			CLRB	8R0	RESET INTERRUPT CONDITIONS
604	015106	012705	100000		MOV	8BIT15,R5	SETUP INTERRUPT MASK
605	015112	012704	015200		MOV	8151,R4	SETUP INTERRUPT RETURN ADDRESS
606	015116				INTENB		ENABLE IEC INTERRUPT
	015116	052777	000100	007060	BIS	8BIT6,8CIR	ENABLE INTERRUPT IN IEC 11A

607	015124	105037	177776		CLRB	00PS		;ALLOW SYSTEM INTERRUPTS
608	015130				SAVCIR			;SAVE CIR CONTENT
	015130	017700	007050		MOV	@CIR,R0		;SAVE CIR CONTENT
609	015134	113777	024216	007050	MOVB	TALKER,@IOR		;ADDRESS TALKER !
610	015142	005037	015150		CLR	14:		;WAIT FOR INTERRUPT
611	015146	005227		13:	INC	(PC).		
612	015150	000000		14:	.WORD	0		
613	015152	001375			BNE	13:		
614	015154				SAVIOR			;SAVE IOR CONTENT
	015154	017700	007032		MOV	@IOR,R2		;SAVE IOR CONTENT
615	015160				SAVCIR			;SAVE CIR CONTENT
	015160	017700	007020		MOV	@CIR,R0		;SAVE CIR CONTENT
616	015164				SAVSMR			;SAVE SMR CONTENT
	015164	017701	007016		MOV	@SMR,R1		;SAVE SMR CONTENT
617	015170				ERROR	42.		;NO INTERRUPT TO TALKER ADDRESSING
	015170	104452			TRAP	42.		;ERROR 42.
	015172	000000			.WORD	0		;ERROR COUNT
618	015174	000137	014516		JMP	TEST9		;RESTART TEST9
619	015200	103004		15:	BCC	16:		;IF CC. INTERRUPT OK
620	015202				ERROR	43.		;ILLEGAL INTERRUPT
	015202	104453			TRAP	43.		;ERROR 43.
	015204	000000			.WORD	0		;ERROR COUNT
621	015206	000137	014516		JMP	TEST9		;RESTART TEST 9
622	015212			16:	GTS			;GOTO STANDBY STATE
	015212	052777	000004	006766	BIS	@BIT2,@SMR		;GO TO STAND BY
623	015220	005037	015234		CLR	18:		;WAIT FOR CSBS AND TACS
624	015224	122711	000012	17:	CMPB	@12,@R1		
625	015230	001415			BEQ	19:		
626	015232	005227			INC	(PC).		
627	015234	000000		18:	.WORD	0		
628	015236	001372			BNE	17:		
629	015240				SAVCIR			;SAVE CIR CONTENT
	015240	017700	006740		MOV	@CIR,R0		;SAVE CIR CONTENT
630	015244				SAVSMR			;SAVE SMR CONTENT
	015244	017701	006736		MOV	@SMR,R1		;SAVE SMR CONTENT
631	015250				SAVIOR			;SAVE IOR CONTENT
	015250	017702	006736		MOV	@IOR,R2		;SAVE IOR CONTENT
632	015254				ERROR	44.		;NO CSBS AND TACS
	015254	104454			TRAP	44.		;ERROR 44.
	015256	000000			.WORD	0		;ERROR COUNT
633	015260	000137	014516		JMP	TEST9		;RESTART TEST9
634	015264			19:	IST11	@CIR		;IS NO LAC SET ?
	015264	032777	004000	006712	BIT	@BIT11,@CIR		
635	015272	001012			BNE	20:		;IF NE, YES
636	015274				SAVCIR			;SAVE CIR CONTENT
	015274	017700	006704		MOV	@CIR,R0		;SAVE CIR CONTENT
637	015300				SAVSMR			;SAVE SMR CONTENT
	015300	017701	006702		MOV	@SMR,R1		;SAVE SMR CONTENT
638	015304				SAVIOR			;SAVE IOR CONTENT
	015304	017702	006702		MOV	@IOR,R2		;SAVE IOR CONTENT
639	015310				ERROR	45.		;NO LAC NOT SET
	015310	104455			TRAP	45.		;ERROR 45.
	015312	000000			.WORD	0		;ERROR COUNT
640	015314	000137	014516		JMP	TEST9		;RESTART SUBTEST
641	015320			20:	MCLEAR			;RESET IFC 11A
	015320	052777	000040	006656	BIS	@BIT5,@CIR		;EXECUTE MASTER CLEAR
642	015326				LOOP	1,LOPT6		

	015326	005327		DEC	(PC).	;COUNT LOOPS
	015330	177777	644:	.WORD	1	;
	015332	005737	015330	TST	644	; IS 644 ZERO ?
	015336	001402		BEQ	.+6	;
	015340	000137	014552	JMP	LOPT6	;
	015344	012737	177777 015330	MOV	#-1,644	;SETUP NEW LOOP COUNT
643	015352			REMVEC	#VECTOR	;REMOVE VECTOR
	015352	012746	024222	MOV	#VECTOR,-(SP)	;SAVE #VECTOR ONTO STACK
	015356	104002		EMT	2	;RESET #VECTOR VECTOR
644	015360			TSTEND	6	;END OF SUBTEST
	015360	005737	011670	TST	MSGFLG	;PRINTOUT OR NOT?
	015364	001004		BNE	664	
	015366	012746	023274	MOV	#MTE6,-(SP)	;SAVE #MTE6 ONTO STACK
	015372	004737	005654	CALL	\$.WRT	;INITIATE PRINTOUT
	015376	000207	664:	RETURN;		;RETURN TO CALLER

646					.SBTTL	TEST 10 TEST DATA TRANSFER WITH INTERRUPT	
647	015400				TEST10: SETVEC	VECTOR,IECINT,00	;SET VECTOR ; **V03E02**
	015400	012746	024222		MOV	VECTOR,-(SP)	;SAVE VECTOR ONTO STACK
	015404	012746	021334		MOV	IECINT,-(SP)	;SAVE IECINT ONTO STACK
	015410	012746	000000		MOV	00,-(SP)	;SAVE 00 ONTO STACK
	015414	104001			EMT	1	;SETUP VECTOR VECTOR
648	015416	005737	011670		TST	MSGFLG	;PRINTOUT OR NOT?
649	015422	001004			BNE	LOPT7	
650	015424				WRITE	MT7S	;START MESSAGE TEST 10
	015424	012746	023310		MOV	MT7S,-(SP)	;SAVE MT7S ONTO STACK
	015430	004737	005654		CALL	..WRT	;INITIATE PRINTOUT
651	015434				LOPT7: MCLEAR		;MASTER CLEAR
	015434	052777	000040	006542	BIS	BIT5,CIR	;EXECUTE MASTER CLEAR
652	015442				SACS		;SET CONTROLLER ACTIVE STATE
	015442	052777	000001	006534	BIS	BIT0,CIR	;SET SACS BIT IN CIR
653	015450				SIC		;SEND INTERFACE CLEAR
	015450	052777	000100	006530	BIS	BIT6,SMR	;SEND INTERFACE CLEAR
654	015456				13: TST8	CIR	;WAIT FOR STATE CHANGE
	015456	032777	000400	006520	BIT	BIT8,CIR	
655	015464	001774			BEQ	13	
656	015466				CLR8	CIR	;RESET INTERRUPT CONDITIONS
	015466	042777	000400	006510	BIC	BIT8,CIR	
657	015474				TST8	CIR	;WAIT FOR SIAS END ; **V01E01**
	015474	032777	000400	006502	BIT	BIT8,CIR	
658	015502	001774			BEQ	..6	
659	015504				CLR8	CIR	;RESET STATE CHANGE ; **V01E01**
	015504	042777	000400	006472	BIC	BIT8,CIR	; **V01E01**
660	015512				GTS		;GO TO STANDBY STATE
	015512	052777	000004	006466	BIS	BIT2,SMR	;GO TO STAND BY
661	015520	104000			SCOPE		;SET SCOPE LOOP MARKER
662	015522				1: TST9	SMR	;WAIT FOR CSBS
	015522	032777	001000	006456	BIT	BIT9,SMR	
663	015530	001774			BEQ	1	
664	015532				TCS		;TAKE CONTROL SYNCR.
	015532	052777	000001	006446	BIS	BIT0,SMR	
665	015540	005037	015556		CLR	3	
666	015544				2: TST12	CIR	;WAIT FOR ILLEGAL MESSAGE
	015544	032777	010000	006432	BIT	BIT12,CIR	
667	015552	001013			BNE	4	
668	015554	005227			INC	(PC).	
669	015556	000000			3: .WORD	0	
670	015560	001371			BNE	2	
671	015562				SAVCIR		;SAVE CIR CONTENT
	015562	017700	006416		MOV	CIR,R0	;SAVE CIR CONTENT
672	015566				SAVSMR		;SAVE SMR CONTENT
	015566	017701	006414		MOV	SMR,R1	;SAVE SMR CONTENT
673	015572				ERROR	46.	;NO ILLEGAL MESSAGE
	015572	104456			TRAP	46.	;ERROR 46.
	015574	000000			.WORD	0	;ERROR COUNT
674	015576	000137	015400		JMP	TEST10	;RESTART SUBTEST
675	015602	042777	177400	006374	4: BIC	177400,CIR	;RESET INTERRUPT CONDITIONS
676	015610				TCA		;TAKE CONTROL ASYNCR.
	015610	052777	000002	006370	BIS	BIT1,SMR	;TAKE CONTROL ASYNC
677	015616				5: TST8	CIR	;WAIT FOR STATE CHANGE
	015616	032777	000400	006360	BIT	BIT8,CIR	
678	015624	001774			BEQ	5	
679	015626				6: TST8	SMR	;CACS SET ?

680	015626	032777	000400	006352	BIT	#BIT8, @SMR	
681	015634	001774			BEQ	6:	; IF EQ, NO
682	015636	104000			SCOPE		; SET SCOPE LOOP MARKER
683	015640	042777	177400	006336	BIC	#177400, @CIR	; RESET INTERRUPT BIT
684	015646	012705	100000		MOV	#BIT15, R5	; SETUP INTERRUPT MASK
685	015652	012704	015740		MOV	#9:, R4	; SETUP INTERRUPT RETURN ADDRESS
686	015656				SAVCIR		; SAVE CIR CONTENT
687	015662	017700	006322		MOV	@CIR, R0	; SAVE CIR CONTENT
688	015662	052777	000100	006314	INTENB		; ENABLE IEC-INTERRUPT
689	015670	105037	177776		BIS	#BIT6, @CIR	; ENABLE INTERRUPT IN IEC 11A
690	015674	113777	024220	006310	CLRB	@PS	; ENABLE SYSTEM INTERRUPT
691	015702	005037	015710		MOVB	LISTNR, @IOR	; SEND LISTENER ADDRESS
692	015706	005227			CLR	8:	
693	015710	000000			INC	(PC),	; WAIT FOR INTERRUPT
694	015712	001375			.WORD	0	
695	015714				BNE	7:	
696	015714	017700	006264		SAVCIR		; SAVE CIR CONTENT
697	015720				MOV	@CIR, R0	; SAVE CIR CONTENT
698	015720	017701	006262		SAVSMR		; SAVE SMR CONTENT
699	015724				MOV	@SMR, R1	; SAVE SMR CONTENT
700	015724	017702	006262		SAVIOR		; SAVE IOR CONTENT
701	015730				MOV	@IOR, R2	; SAVE IOR CONTENT
702	015730	104462			ERROR	50.	; NO INTERRUPT TO LISTENER ADDRESSING
703	015732	000000			TRAP	50.	; ERROR 50.
704	015734	000137	015400		.WORD	0	; ERROR COUNT
705	015740	103004			JMP	TEST10	; RESTART SUBTEST
706	015742				BCC	10:	; IF CC, INTERRUPT OK
707	015742	104463			ERROR	51.	; ILLEGAL INTERRUPT
708	015744	000000			TRAP	51.	; ERROR 51.
709	015746	000137	015400		.WORD	0	; ERROR COUNT
710	015752	104000			JMP	TEST10	; RESTART SUBTEST
711	015754	042777	177400	006222	SCOPE		; SET SCOPE LOOP MARKER
712	015762	012705	100000		BIC	#177400, @CIR	; CLEAR INTERRUPT BITS
713	015766	012704	016054		MOV	#BIT15, R5	; SET INTERRUPT MASK
714	015772				MOV	#14:, R4	; SETUP INTERRUPT RETURN
715	015772	017700	006206		SAVCIR		; SAVE CIR CONTENT
716	015776				MOV	@CIR, R0	; SAVE CIR CONTENT
717	015776	052777	000100	006200	INTENB		; ENABLE IEC INTERRUPT
718	016004	105037	177776		BIS	#BIT6, @CIR	; ENABLE INTERRUPT IN IEC 11A
719	016010	113777	024216	006174	CLRB	@PS	; ENABLE SYSTEM INTERRUPTS
720	016016	005037	016024		MOVB	TALKER, @IOR	; ADDRESS TALKER AGAIN
721	016022	005227			CLR	12:	; WAIT FOR INTERRUPT
722	016024	000000			INC	(PC),	
723	016026	001375			.WORD	0	
724	016030				BNE	11:	
725	016030	017700	006150		SAVCIR		; SAVE CIR CONTENT
726	016034				MOV	@CIR, R0	; SAVE CIR CONTENT
727	016034	017701	006146		SAVSMR		; SAVE SMR CONTENT
728	016040				MOV	@SMR, R1	; SAVE SMR CONTENT
729	016040	017702	006146		SAVIOR		; SAVE IOR CONTENT
730	016044				MOV	@IOR, R2	; SAVE IOR CONTENT
731	016044	104465			ERROR	53.	; NO INTERRUPT TO TALKER ADDRESSING
732	016046	000000			TRAP	53.	; ERROR 53.
733	016050	000137	015400		.WORD	0	; ERROR COUNT
734	016054				JMP	TEST10	; RESTART SUBTEST
735	016054	052777	000004	006124	GTS		; GOTO STANDBY STATE
					BIS	#BIT2, @SMR	; GO TO STAND BY

719	016062				15:	TST9	BSMR		;WAIT FOR CSBS
	016062	032777	001000	006116		BIT	#BIT9,BSMR		
720	016070	001774				BEQ	15:		
721	016072					TST12	BSMR		;LACS SET ?
	016072	032777	010000	006106		BIT	#BIT12,BSMR		
722	016100	001012				BNE	16:		;IF EQ, NO
723	016102					SAVCIR			;SAVE CIR CONTENT
	016102	017700	006076			MOV	BCIR,R0		;SAVE CIR CONTENT
724	016106					SAVSMR			;SAVE SMR CONTENT
	016106	017701	006074			MOV	BSMR,R1		;SAVE SMR CONTENT
725	016112					SAVIOR			;SAVE IOR CONTENT
	016112	017702	006074			MOV	BIOR,R2		;SAVE IOR CONTENT
726	016116					ERROR	52.		;NO LACS
	016116	104464				TRAP	52.		;ERROR 52.
	016120	000000				.WORD	0		;ERROR COUNT
727	016122	000137	015400			JMP	TEST10		;RESTART SUBTEST
728	016126	112777	177777	006056	16:	MOVB	#-1,BIOR		;SEND A DATA BYTE
729	016134				17:	TST15	BCIR		;DATA ACCEPTED ?
	016134	032777	100000	006042		BIT	#BIT15,BCIR		
730	016142	001774				BEQ	17:		;IF EQ, WAIT
731	016144					CLR15	BCIR		;RESET DATA ACCEPTED
	016144	042777	100000	006032		BIC	#BIT15,BCIR		
732	016152					CLR8	BCIR		;RESET STATE CHANGE
	016152	042777	000400	006024		BIC	#BIT8,BCIR		
733	016160	105377	006026			DECB	BIOR		;SEND NEXT DATA BYTE ;**V01E02**
734	016164	005227			18:	INC	(PC).		;WAIT A SHORT TIME
735	016166	000000				.WORD	0		
736	016170	001375				BNE	18:		
737	016172					TST15	BCIR		;DATA ACCEPTED ?
	016172	032777	100000	006004		BIT	#BIT15,BCIR		
738	016200	001412				BEQ	19:		;IF EQ, NO
739	016202					SAVCIR			;SAVE CIR CONTENT
	016202	017700	005776			MOV	BCIR,R0		;SAVE CIR CONTENT
740	016206					SAVSMR			;SAVE SMR CONTENT
	016206	017701	005774			MOV	BSMR,R1		;SAVE SMR CONTENT
741	016212					SAVIOR			;SAVE IOR CONTENT
	016212	017702	005774			MOV	BIOR,R2		;SAVE IOR CONTENT
742	016216					ERROR	54.		;DATA ACCEPTED TO SECOND DATA BYTE
	016216	104466				TRAP	54.		;ERROR 54.
	016220	000000				.WORD	0		;ERROR COUNT
743	016222	000137	015400			JMP	TEST10		;RESTART SUBTEST
744	016226				19:	SAVIOR			;GET IOR CONTENT
	016226	017702	005760			MOV	BIOR,R2		;SAVE IOR CONTENT
745	016232	000302				SWAB	R2		
746	016234	120227	000377			CMQB	R2,#377		;DATA OK ?
747	016240	001411				BEQ	20:		;IF EQ, YES
748	016242	000302				SWAB	R2		;NO, REBUILD DATA PATTERN READ
749	016244					SAVSMR			;SAVE SMR CONTENT
	016244	017701	005736			MOV	BSMR,R1		;SAVE SMR CONTENT
750	016250					SAVCIR			;SAVE CIR CONTENT
	016250	017700	005730			MOV	BCIR,R0		;SAVE CIR CONTENT
751	016254					ERROR	55.		;DATA CHECK ON IEC TALKER OR LISTENER
	016254	104467				TRAP	55.		;ERROR 55.
	016256	000000				.WORD	0		;ERROR COUNT
752	016260	000137	015400			JMP	TEST10		;RESTART SUBTEST
753	016264				20:	TST15	BCIR		;SECOND "DATA ACCEPTED ?
	016264	032777	100000	005712		BIT	#BIT15,BCIR		

754	016272	001012		BNE	218		;IF NE, YES
755	016274			SAVCIR			;SAVE CIR CONTENT
	016274	017700	005704	MOV	BCIR,R0		;SAVE CIR CONTENT
756	016300			SAVSMR			;SAVE SMR CONTENT
	016300	017701	005702	MOV	BSMR,R1		;SAVE SMR CONTENT
757	016304			SAVIOR			;SAVE IOR CONTENT
	016304	017702	005702	MOV	BIOR,R2		;SAVE IOR CONTENT
758	016310			ERROR	56.		;NO SECOND DATA ACCEPTED
	016310	104470		TRAP	56.		;ERROR 56.
	016312	000000		.WORD	0		;ERROR COUNT
759	016314	000137	015400	JMP	TEST10		;RESTART SUBTEST
760	016320			SAVIOR			;GET SECOND DATA BYTE
	016320	017702	005666	MOV	BIOR,R2		;SAVE IOR CONTENT
761	016324	000302		SWAB	R2		
762	016326	122702	000376	CMQB	#376,R2		;IS SECOND DATA BYTE OK ? ;**V01E02**
763	016332	001411		BEQ	TST10A		;IF EQ, YES
764	016334	000302		SWAB	R2		;NO, REBUILD PATTERN READ
765	016336			SAVCIR			;SAVE CIR CONTENT
	016336	017700	005642	MOV	BCIR,R0		;SAVE CIR CONTENT
766	016342			SAVSMR			;SAVE SMR CONTENT
	016342	017701	005640	MOV	BSMR,R1		;SAVE SMR CONTENT
767	016346			ERROR	57.		;DATA CHECK ON SECOND BYTE
	016346	104471		TRAP	57.		;ERROR 57.
	016350	000000		.WORD	0		;ERROR COUNT
768	016352	000137	015400	JMP	TEST10		;RESTART SUBTEST

218:

```

770 .SBTTL TEST 10A RANDOM TALKER AND LISTENER DATA TEST
771 ;
772 ;
773 ; THIS TEST CNA NOT BE CALLED SEPARATELY, DUE TO THE NECESSARY
774 ; SETUP TO BRING THE BUS INTO THIS STATE.
775 ;
776 ;
777 ;
778 016356 005037 016604 TST10A: .ENABL LSB ;DISABLE RANDOM MODE
779 016362 005046 CLR RANMOD ;CLEAR DATA BUFFER
780 016364 005237 024204 INC -(SP) ;ADDRESS CIR HIGH BYTE
781 016370 105077 005610 CLR CIR ;RESET INTERRUPT BITS
782 016374 005337 024204 CLR RB SCIR ;REBUILD FULL CIR ADDRESS
783 016400 104000 DEC CIR ;SET SCOPE LOOP MARKER
784 016402 012705 100000 SCOPE ;SETUP INTERRUPT MASK
785 016406 012704 016474 LOPT7A: MOV #BIT15,R5 ;SETUP INTERRUPT RETURN ADDRESS
786 016412 042777 177400 005564 MOV #31,R4 ;RESET INTERRUPT BITS
787 016420 052777 000100 005556 BIC #177400,SCIR ;ENABLE IEC INTERRUPT
788 016426 105037 177776 BIS #BIT6,SCIR ;ENABLE INTERRUPT IN IEC 11A
789 016432 011677 005554 CLR RB #PS ;ENABLE SYSTEM INTERRUPT
790 016436 005037 016444 MOV (SP),R4 ;SEND A DATA BYTE
791 016442 005227 11: INC 21 ;WAIT FOR INTERRUPT
792 016444 000000 21: .WORD 0 ;
793 016446 001375 BNE 11 ;
794 016450 SAVIEC ;SAVE ALL IEC REGISTERS
795 016450 017700 005530 MOV SCIR,R0 ;SAVE CIR CONTENT
796 016454 017701 005526 MOV SMR,R1 ;SAVE SMR CONTENT
797 016460 017702 005526 MOV RIOR,R2 ;SAVE IOR CONTENT
798 016464 104474 ERROR 60. ;NO INTERRUPT TO DATA SENDING
799 016466 000000 TRAP 60. ;ERROR 60.
800 016470 000137 015400 .WORD 0 ;ERROR COUNT
801 016474 103004 JMP TEST10 ;RESTART TEST
802 016476 104475 31: BCC 41 ;IF CC, INTERRUPT OK
803 016500 000000 TRAP 61. ;ILLEGAL INTERRUPT TO DATA SENDING
804 016502 000137 015400 .WORD 0 ;ERROR COUNT
805 016506 017703 005500 JMP TEST10 ;RESTART TEST
806 016512 105003 MOV RIOR,R3 ;READ BACK DATA BYTE
807 016514 000303 CLR RB R3 ;PREPARE DATA
808 016516 020316 SWAB R3 ;
809 016520 001002 IF R3 EQ (SP) GOTO 51 ;COMPARE R3 AND (SP)
810 016522 000137 016554 CMP R3,(SP) ;
811 016526 017700 005452 BNE .+6 ;
812 016532 017701 005450 JMP 51 ;
813 016536 017702 005450 SAVIEC ;SAVE ALL IEC REGISTERS
814 016542 011604 MOV SCIR,R0 ;SAVE CIR CONTENT
815 016544 104476 MOV SMR,R1 ;SAVE SMR CONTENT
816 016546 000000 TRAP 62. ;SAVE IOR CONTENT
817 016550 000137 015400 MOV (SP),R4 ;SAVE 'SHOULD BE' VALUE
818 016554 005737 016604 ERROR 62. ;DATA CHECK ON IEC BUS
819 016560 001402 TRAP 62. ;ERROR 62.
820 .WORD 0 ;ERROR COUNT
821 016550 000137 015400 JMP TEST10 ;RESTART TEST
822 016554 005737 016604 51: IF RANMOD NE #0 GOTO 61 ;IS RANMOD ZERO ?
823 016560 001402 TST RANMOD ;
824 BEQ .+6 ;

```

016562	000137	016606		JMP	6:		
809 016566	005216			INC	(SP)		;BUILD NEXT DATA PATTERN
810 016570				IF (SP) NE #400	GOTO LOPT7A		
016570	021627	000400		CMP	(SP), #400		;COMPARE (SP) AND #400
016574	001401			BEQ	.+4		
016576	000701			BR	LOPT7A		
811 016600	012727	177777		MOV	#-1,(PC).		;SWITCH TO RANDOM MODE
812 016604	000000			.WORD	0		;RANDOM MODE FLAG
813 016606	004737	011672		CALL	RANDOM		;GET A RANDOM PATTERN IN R0
814 016612	010016			MOV	R0,(SP)		;MOVE PATTERN ONTO STACK
815 016614	042716	177400		BIC	#177400,(SP)		;BUILD A BYTE
816 016620	005227			INC	(PC).		;LOOP 65536 TIMES
817 016622	000000			.WORD	0		
818 016624	001266			BNE	LOPT7A		
819 016626	005037	016604		CLR	RANMOD		;SWITCH BACK TO NORMAL MODE
820 016632	012716	000227		MOV	#222,(SP)		;PREPARE LAST DATA PATTERN
821 016636				SET2	@CIR		;INDICATE "LAST BYTE"
016636	052777	000004	005340	BIS	#BIT2,@CIR		
822 016644	012677	005342		MOV	(SP)+,@IOR		;LOAD IT
823 016650	005037	016666		CLR	8:		;WAIT FOR "DATA ACCEPTED"
824 016654				TST15	@CIR		
016654	032777	100000	005322	BIT	#BIT15,@CIR		
825 016662	001015			BNE	9:		
826 016664	005227			INC	(PC).		
827 016666	000000			.WORD	0		
828 016670	001371			BNE	7:		
829 016672				SAVIEC			;SAVE ALL IEC REGISTERS
016672	017700	005306		MOV	@CIR,R0		;SAVE CIR CONTENT
016676	017701	005304		MOV	@SMR,R1		;SAVE SMR CONTENT
016702	017702	005304		MOV	@IOR,R2		;SAVE IOR CONTENT
830 016706				ERROR	63.		;NO DATA ACC WHEN SENDING LAST BYTE
016706	104477			TRAP	63.		;ERROR 63.
016710	000000			.WORD	0		;ERROR COUNT
831 016712	000137	015400		JMP	TEST10		;RESTART TEST
832 016716				TST13	@CIR		;IS "END" SET ?
016716	032777	020000	005260	BIT	#BIT13,@CIR		
833 016724	001012			BNE	10:		;IF NE, YES
834 016726				SAVIEC			;SAVE ALL IEC REGISTERS
016726	017700	005252		MOV	@CIR,R0		;SAVE CIR CONTENT
016732	017701	005250		MOV	@SMR,R1		;SAVE SMR CONTENT
016736	017702	005250		MOV	@IOR,R2		;SAVE IOR CONTENT
835 016742				ERROR	64.		;NO "END" TO "LAST BYTE"
016742	104500			TRAP	64.		;ERROR 64.
016744	000000			.WORD	0		;ERROR COUNT
836 016746	000137	015400		JMP	TEST10		;RESTART TEST
837 016752	017703	005234		MOV	@IOR,R3		;READ BACK DATA BYTE
838 016756	105003			CLRB	R3		
839 016760	000303			SWAB	R3		;PREPARE FOR CHECK
840 016762				IF R3 EQ #222	GOTO 11:		
016762	020327	000222		CMP	R3,#222		;COMPARE R3 AND #222
016766	001002			BNE	.+6		
016770	000137	017020		JMP	11:		
841 016774				SAVIEC			;SAVE ALL IEC REGISTERS
016774	017700	005204		MOV	@CIR,R0		;SAVE CIR CONTENT
017000	017701	005202		MOV	@SMR,R1		;SAVE SMR CONTENT
017004	017702	005202		MOV	@IOR,R2		;SAVE IOR CONTENT
842 017010				ERROR	65.		;DATA CHECK ON "LAST BYTE"

	017010	104501			TRAP	65.		;ERROR 65.
	017012	000000			.WORD	0		;ERROR COUNT
843	017014	000137	015400		JMP	TEST10		;RESTART SUBTEST
844	017020			11\$:	MCLEAR			;MASTER CLEAR
	017020	052777	000040	005156	BIS	#BIT5,8CIR		;EXECUTE MASTER CLEAR
845	017026				TSTEND	7		;END OF SUBTEST 7
	017026	005737	011670		TST	MSGFLG		;PRINTOUT OR NOT?
	017032	001004			BNE	68\$		
	017034	012746	023373		MOV	#MTE7, (SP)		;SAVE #MTE7 ONTO STACK
	017040	004737	005654		CALL	\$..WRT		;INITIATE PRINTOUT
	017044	000207		68\$:	RETURN;			;RETURN TO CALLER
846					.DSABL	LSB		
847								

```

849      .SBTTL TEST 11 -- TEST FUNCTION OF <LTN> AND <LUN>
850 017046 005737 011670 TEST11: TST MSGFLG ;PRINTOUT OR NOT?
851 017052 001004      BNE LOPT8
852 017054      WRITE MT8S ;START MESSAGE TEST 11
      017054 012746 023410      MOV #MT8S, (SP) ;SAVE #MT8S ONTO STACK
      017060 004737 005654      CALL $.WRT ;INITIATE PRINTOUT
853 017064 104000      LOPT8: SCOPE ;SET SCOPE LOOP MARKER
854 017066      MCLEAR ;MASTER CLEAR
      017066 052777 000040 005110      BIS #BIT5, @CIR ;EXECUTE MASTER CLEAR
855 017074      SACS ;SET CONTROLLER ACTIVE
      017074 052777 000001 005102      BIS #BIT0, @CIR ;SET SACS BIT IN CIR
856 017102      SIC ;SEND INTERFACE CLEAR
      017102 052777 000100 005076      BIS #BIT6, @SMR ;SEND INTERFACE CLEAR
857 017110      TST13 @SMR ;WAIT FOR SIAS OFF
      017110 032777 020000 005070      BIT #BIT13, @SMR
858 017116 001374      BNE 1$
859 017120 042777 177400 005056      BIC #177400, @CIR ;RESET INTERRUPT CONDITIONS
860 017126      LTN ;LISTEN !!
      017126 052777 000020 005052      BIS #BIT4, @SMR
861 017134 000240      NOP
862 017136 000240      NOP
863 017140 000240      NOP
864 017142      GTS ;GOTO STAND BY
      017142 052777 000004 005036      BIS #BIT2, @SMR ;GO TO STAND BY
865 017150      TST9 @SMR ;WAIT FOR CSBS
      017150 032777 001000 005030      BIT #BIT9, @SMR
866 017156 001774      BEQ 2$
867 017160 005037 017176      CLR 4$ ;WAIT FOR LACS
868 017164      TST12 @SMR
      017164 032777 010000 005014      BIT #BIT12, @SMR
869 017172 001015      BNE 5$
870 017174 005227      INC (PC)+
871 017176 000000      .WORD 0
872 017200 001371      BNE 3$
873 017202      SAVIEC ;SAVE ALL IEC REGISTERS
      017202 017700 004776      MOV @CIR, R0 ;SAVE CIR CONTENT
      017206 017701 004774      MOV @SMR, R1 ;SAVE SMR CONTENT
      017212 017702 004774      MOV @IOR, R2 ;SAVE IOR CONTENT
874 017216      ERROR 66. ;NO LACS TO LTN
      017216 104502      TRAP 66. ;ERROR 66.
      017220 000000      .WORD 0 ;ERROR COUNT
875 017222 000137 017046      JMP TEST11
876 017226 042777 177400 004750 5$: BIC #177400, @CIR ;RESET ALL INTERRUPT CONDITIONS
877 017234      TCA ;GO BACK TO CACS
      017234 052777 000002 004744      BIS #BIT1, @SMR ;TAKE CONTROL ASYNC
878 017242      TST8 @SMR ;WAIT FOR CACS AGAIN
      017242 032777 000400 004736      BIT #BIT8, @SMR
879 017250 001774      BEQ 6$
880 017252      GTS ;TRY AGAIN TO GET LACS
      017252 052777 000004 004726      BIS #BIT2, @SMR ;GO TO STAND BY
881 017260 005037 017276      CLR 8$
882 017264      TST12 @SMR
      017264 032777 010000 004714      BIT #BIT12, @SMR
883 017272 001015      BNE 9$
884 017274 005227      INC (PC)+
885 017276 000000      .WORD 0
886 017300 001371      BNE 7$

```

PC	PC+4	PC+8	PC+12	PC+16	PC+20	PC+24	PC+28	PC+32	PC+36	PC+40	PC+44	PC+48	PC+52	PC+56	PC+60	PC+64	PC+68	PC+72	PC+76	PC+80	PC+84	PC+88	PC+92	PC+96	PC+100	PC+104	PC+108	PC+112	PC+116	PC+120	PC+124	PC+128	PC+132	PC+136	PC+140	PC+144	PC+148	PC+152	PC+156	PC+160	PC+164	PC+168	PC+172	PC+176	PC+180	PC+184	PC+188	PC+192	PC+196	PC+200	PC+204	PC+208	PC+212	PC+216	PC+220	PC+224	PC+228	PC+232	PC+236	PC+240	PC+244	PC+248	PC+252	PC+256	PC+260	PC+264	PC+268	PC+272	PC+276	PC+280	PC+284	PC+288	PC+292	PC+296	PC+300	PC+304	PC+308	PC+312	PC+316	PC+320	PC+324	PC+328	PC+332	PC+336	PC+340	PC+344	PC+348	PC+352	PC+356	PC+360	PC+364	PC+368	PC+372	PC+376	PC+380	PC+384	PC+388	PC+392	PC+396	PC+400	PC+404	PC+408	PC+412	PC+416	PC+420	PC+424	PC+428	PC+432	PC+436	PC+440	PC+444	PC+448	PC+452	PC+456	PC+460	PC+464	PC+468	PC+472	PC+476	PC+480	PC+484	PC+488	PC+492	PC+496	PC+500	PC+504	PC+508	PC+512	PC+516	PC+520	PC+524	PC+528	PC+532	PC+536	PC+540	PC+544	PC+548	PC+552	PC+556	PC+560	PC+564	PC+568	PC+572	PC+576	PC+580	PC+584	PC+588	PC+592	PC+596	PC+600	PC+604	PC+608	PC+612	PC+616	PC+620	PC+624	PC+628	PC+632	PC+636	PC+640	PC+644	PC+648	PC+652	PC+656	PC+660	PC+664	PC+668	PC+672	PC+676	PC+680	PC+684	PC+688	PC+692	PC+696	PC+700	PC+704	PC+708	PC+712	PC+716	PC+720	PC+724	PC+728	PC+732	PC+736	PC+740	PC+744	PC+748	PC+752	PC+756	PC+760	PC+764	PC+768	PC+772	PC+776	PC+780	PC+784	PC+788	PC+792	PC+796	PC+800	PC+804	PC+808	PC+812	PC+816	PC+820	PC+824	PC+828	PC+832	PC+836	PC+840	PC+844	PC+848	PC+852	PC+856	PC+860	PC+864	PC+868	PC+872	PC+876	PC+880	PC+884	PC+888	PC+892	PC+896	PC+900	PC+904	PC+908	PC+912	PC+916	PC+920	PC+924	PC+928	PC+932	PC+936	PC+940	PC+944	PC+948	PC+952	PC+956	PC+960	PC+964	PC+968	PC+972	PC+976	PC+980	PC+984	PC+988	PC+992	PC+996	PC+1000	PC+1004	PC+1008	PC+1012	PC+1016	PC+1020	PC+1024	PC+1028	PC+1032	PC+1036	PC+1040	PC+1044	PC+1048	PC+1052	PC+1056	PC+1060	PC+1064	PC+1068	PC+1072	PC+1076	PC+1080	PC+1084	PC+1088	PC+1092	PC+1096	PC+1100	PC+1104	PC+1108	PC+1112	PC+1116	PC+1120	PC+1124	PC+1128	PC+1132	PC+1136	PC+1140	PC+1144	PC+1148	PC+1152	PC+1156	PC+1160	PC+1164	PC+1168	PC+1172	PC+1176	PC+1180	PC+1184	PC+1188	PC+1192	PC+1196	PC+1200	PC+1204	PC+1208	PC+1212	PC+1216	PC+1220	PC+1224	PC+1228	PC+1232	PC+1236	PC+1240	PC+1244	PC+1248	PC+1252	PC+1256	PC+1260	PC+1264	PC+1268	PC+1272	PC+1276	PC+1280	PC+1284	PC+1288	PC+1292	PC+1296	PC+1300	PC+1304	PC+1308	PC+1312	PC+1316	PC+1320	PC+1324	PC+1328	PC+1332	PC+1336	PC+1340	PC+1344	PC+1348	PC+1352	PC+1356	PC+1360	PC+1364	PC+1368	PC+1372	PC+1376	PC+1380	PC+1384	PC+1388	PC+1392	PC+1396	PC+1400	PC+1404	PC+1408	PC+1412	PC+1416	PC+1420	PC+1424	PC+1428	PC+1432	PC+1436	PC+1440	PC+1444	PC+1448	PC+1452
----	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------

					.SBTTL	TEST 12	TEST FUNCTION OF <BLOCK DAC>
910					TEST12:	MSGFLG	PRINTOUT OR NOT?
911	017510	005737	011670			LOPT9	
912	017514	001004				WRITE	START MESSAGE TEST 12
913	017516					MOV	SAVE #MT9S ONTO STACK
	017516	012746	023507			CALL	INITIATE PRINTOUT
	017522	004737	005654			1..WRT	
914	017526	104000			LOPT9:	SCOPE	SET SCOPE LOOP MARKER
915	017530					MCLEAR	MASTER CLEAR
	017530	052777	000040	004446		BIS	EXECUTE MASTER CLEAR
916	017536					SET3	TRY TO SET <BLOCK DAC>
	017536	052777	000010	004440		BIS	
917	017544					TST3	IS <BLOCK DAC> SET ?
	017544	032777	000010	004432		BIT	
918	017552	001011				BNE	IF NE, YES
919	017554					SAVIEC	SAVE ALL IEC REGISTERS
	017554	017700	004424			MOV	SAVE CIR CONTENT
	017560	017701	004422			MOV	SAVE SMR CONTENT
	017564	017702	004422			MOV	SAVE IOR CONTENT
920	017570					ERROR	<BLOCK DAC> NOT SET
	017570	104507				TRAP	ERROR 71.
	017572	000000				.WORD	ERROR COUNT
921	017574	000745				BR	RESTART SUBTEST
922	017576				18:	CLR3	TRY TO RESET <BLOCK DAC>
	017576	042777	000010	004400		BIC	
923	017604					TST3	IS <BLOCK DAC> RESET
	017604	032777	000010	004372		BIT	
924	017612	001411				BEQ	IF EQ, YES
925	017614					SAVIEC	SAVE ALL IEC REGISTERS
	017614	017700	004364			MOV	SAVE CIR CONTENT
	017620	017701	004362			MOV	SAVE SMR CONTENT
	017624	017702	004362			MOV	SAVE IOR CONTENT
926	017630					ERROR	<BLOCK DAC> NOT RESET
	017630	104510				TRAP	ERROR 72.
	017632	000000				.WORD	ERROR COUNT
927	017634	000725				BR	RESTART SUBTEST
928	017636	104000			28:	SCOPE	SET SCOPE LOOP MARKER
929	017640					MCLEAR	MASTER CLEAR
	017640	052777	000040	004336		BIS	EXECUTE MASTER CLEAR
930	017646					SACS	SET CONTROLLER ACTIVE
	017646	052777	000001	004330		BIS	SET SACS BIT IN CIR
931	017654					SIC	SEND INTERFACE CLEAR
	017654	052777	000100	004324		BIS	SEND INTERFACE CLEAR
932	017662				38:	TST13	WAIT FOR SIAS OFF
	017662	032777	020000	004316		BIT	
933	017670	001374				BNE	
934	017672					SET3	SET <BLOCK DAC>
	017672	052777	000010	004304		BIS	
935	017700	013777	024220	004304		MOV	SEND LISTENER ADDRESS
936	017706				48:	TST15	WAIT FOR <DATA ACCEPTED>
	017706	032777	100000	004270		BIT	
937	017714	001774				BEQ	
938	017716	042777	177400	004260		BIC	RESET ALL INTERRUPT CONDITIONS
939	017724	013777	024216	004260		MOV	SEND TALKER ADDRESS
940	017732				58:	TST15	WAIT FOR <DATA ACCEPTED>
	017732	032777	100000	004244		BIT	
941	017740	001774				BEQ	
942	017742	042777	177400	004234		BIC	RESET ALL INTERRUPT CONDITIONS

943	017750				GTS		;GOTO <STAND BY> STATE
	017750	052777	000004	004230	BIS	#BIT2,BSMR	;GO TO STAND BY
944	017756				TST12	BSMR	;WAIT FOR <LACS>
	017756	032777	010000	004222	BIT	#BIT12,BSMR	
945	017764	001774			BEQ	61	
946	017766	042777	177400	004210	BIC	#177400,BCIR	;RESET ALL INTERRUPT CONDITIONS
947	017774	005005			CLR	R5	;SETUP INTERRUPT MASK
948	017776	012704	020032		MOV	#81,R4	;SETUP INTERRUPT RETURN ADDRESS
949	020002				INTENB		;ENABLE IEC INTERRUPT
	020002	052777	000100	004174	BIS	#BIT6,BCIR	;ENABLE INTERRUPT IN IEC-11A
950	020010	005037	177776		CLR	#PS	;ENABLE SYSTEM INTERRUPTS
951	020014	012777	000333	004170	MOV	#333,BIOR	;SEND A DATA PATTERN
952	020022	005227			INC	(PC).	;WAIT A SHORT TIME
953	020024	000000			.WORD	0	
954	020026	001375			BNE	71	
955	020030	000412			BR	91	
956	020032				SAVIEC		;OK, NO INTERRUPT OCCURED
	020032	017700	004146		MOV	BCIR,R0	;SAVE ALL IEC REGISTERS
	020036	017701	004144		MOV	BSMR,R1	;SAVE CIR CONTENT
	020042	017702	004144		MOV	BIOR,R2	;SAVE SMR CONTENT
957	020046				ERROR	73.	;SAVE IOR CONTENT
	020046	104511			TRAP	73.	;INTERUPT WITH <BLOCK DAC>
	020050	000000			.WORD	0	;ERROR 73.
958	020052	000137	017510		JMP	TEST12	;ERROR COUNT
959	020056	017703	004130		MOV	BIOR,R3	;RESTART SUBTEST
960	020062	105003			CLRB	R3	;GET INPUT BUFFER CONTENT
961	020064	000303			SWAB	R3	;PREPARE FOR CHECK
962	020066				IF R3 EQ #333 GOTO 101		
	020066	020327	000333		CMP	R3,#333	;COMPARE R3 AND #333
	020072	001002			BNE	.+6	
	020074	000137	020124		JMP	101	
963	020100				SAVIEC		;SAVE ALL IEC REGISTERS
	020100	017700	004100		MOV	BCIR,R0	;SAVE CIR CONTENT
	020104	017701	004076		MOV	BSMR,R1	;SAVE SMR CONTENT
	020110	017702	004076		MOV	BIOR,R2	;SAVE IOR CONTENT
964	020114				ERROR	74.	;IEC DATA CHECK WITH <BLOCK DAC>
	020114	104512			TRAP	74.	;ERROR 74.
	020116	000000			.WORD	0	;ERROR COUNT
965	020120	000137	017510		JMP	TEST12	;RESTART SUBTEST
966	020124				MCLEAR		;MASTER CLEAR
	020124	052777	000040	004052	BIS	#BIT5,BCIR	;EXECUTE MASTER CLEAR
967	020132				LOOP	20.,LOPT9	
	020132	005327			DEC	(PC).	;COUNT LOOPS
	020134	000024			.WORD	20.	
	020136	005737	020134		TST	651	
	020142	001402			BEQ	.+6	;IS 651 ZERO ?
	020144	000137	017526		JMP	LOPT9	
	020150	012737	000024	020134	MOV	#20.,651	;SETUP NEW LOOP COUNT
968	020156				TSTEND	9	;END OF TEST 12
	020156	005737	011670		TST	MSGFLG	;PRINTOUT OR NOT?
	020162	001004			BNE	671	
	020164	012746	023565		MOV	#MTE9,(SP)	;SAVE #MTE9 ONTO STACK
	020170	004737	005654		CALL	1..WRT	;INITIATE PRINTOUT
	020174	000207			RETURN;		;RETURN TO CALLER

970					.SBTTL	TEST 13 -- TEST FUNCTION OF <RSV>, <SRQ> AND <SPAS>
971	020176	005737	011670		TEST13: TST	MSGFLG
972	020202	001004			BNE	LOPT10
973	020204				WRITE	MT105
	020204	012746	023602		MOV	#MT105, -(SP)
	020210	004737	005654		CALL	\$.WRT
974	020214	104000			LOPT10: SCOPE	
975	020216				MCLEAR	
	020216	052777	000040	003760	BIS	#BIT5, @CIR
976	020224				SACS	
	020224	052777	000001	003752	BIS	#BIT0, @CIR
977	020232				SIC	
	020232	052777	000100	003746	BIS	#BIT6, @SMR
978	020240				TST13	@SMR
	020240	032777	020000	003740	BIT	#BIT13, @SMR
979	020246	001374			BNE	1\$
980	020250	042777	177400	003726	BIC	#177400, @CIR
981	020256				SET4	@CIR
	020256	052777	000020	003720	BIS	#BIT4, @CIR
982	020264	005037	020302		CLR	3\$
983	020270				TST14	@CIR
	020270	032777	040000	003706	BIT	#BIT14, @CIR
984	020276	001015			BNE	4\$
985	020300	005227			INC	(PC).
986	020302	000000			.WORD	0
987	020304	001371			BNE	2\$
988	020306				SAVIEC	
	020306	017700	003672		MOV	@CIR, R0
	020312	017701	003670		MOV	@SMR, R1
	020316	017702	003670		MOV	@IOR, R2
989	020322				ERROR	75.
	020322	104513			TRAP	75.
	020324	000000			.WORD	0
990	020326	000137	020176		JMP	TEST13
991	020332				TST7	@CIR
	020332	032777	000200	003644	BIT	#BIT7, @CIR
992	020340	001004			BNE	41\$
993	020342				ERROR	111.
	020342	104557			TRAP	111.
	020344	000000			.WORD	0
994	020346	000137	020176		JMP	TEST13
995	020352				INHSRQ	
	020352	052777	002000	003624	BIS	#BIT10, @CIR
996	020360				TST7	@CIR
	020360	032777	000200	003616	BIT	#BIT7, @CIR
997	020366	001404			BEQ	42\$
998	020370				ERROR	112.
	020370	104560			TRAP	112.
	020372	000000			.WORD	0
999	020374	000137	020176		JMP	TEST13
1000	020400				CLR10	@CIR
	020400	042777	002000	003576	BIC	#BIT10, @CIR
1001	020406				TST7	@CIR
	020406	032777	000200	003570	BIT	#BIT7, @CIR
1002	020414	001004			BNE	43\$
1003	020416				ERROR	113.
	020416	104561			TRAP	113.

	020420	000000				.WORD	0		ERROR COUNT
1004	020422	000137	020176			JMP	TEST13		; **V02E00**
1005	020426	013777	024220	003556	43:	MOV	LISTNR, @IOR		; SEND LISTENER ADDRESS
1006	020434				5:	TST15	@CIR		; WAIT FOR <DATA ACCEPTED>
	020434	032777	100000	003542		BIT	@BIT15, @CIR		
1007	020442	001774				BEQ	5:		
1008	020444	042777	177400	003532		BIC	@177400, @CIR		; RESET ALL INTERRUPT CONDITIONS
1009	020452	013777	024216	003532		MOV	TALKER, @IOR		; SEND TALKER ADDRESS
1010	020460				6:	TST15	@CIR		; WAIT FOR <DATA ACCEPTED>
	020460	032777	100000	003516		BIT	@BIT15, @CIR		
1011	020466	001774				BEQ	6:		
1012	020470	042777	177400	003506		BIC	@177400, @CIR		; RESET ALL INTERRUPT CONDITIONS
1013	020476	012777	000030	003506		MOV	@30, @IOR		; SEND <SERIAL POLL ENABLE>
1014	020504				7:	TST15	@CIR		; WAIT FOR <DATA ACCEPTED>
	020504	032777	100000	003472		BIT	@BIT15, @CIR		
1015	020512	001774				BEQ	7:		
1016	020514	042777	177400	003462		BIC	@177400, @CIR		; RESET ALL INTERRUPT CONDITIONS
1017	020522					GTS			; GOTO <STAND BY> STATE
	020522	052777	000004	003456		BIS	@BIT2, @SMR		; GO TO STAND BY
1018	020530	005037	020546			CLR	9:		
1019	020534				8:	TST15	@SMR		; WAIT FOR <SPAS>
	020534	032777	100000	003444		BIT	@BIT15, @SMR		
1020	020542	001015				BNE	10:		
1021	020544	005227				INC	(PC),		
1022	020546	000000			9:	.WORD	0		
1023	020550	001371				BNE	8:		
1024	020552					SAVIEC			; SAVE ALL IEC REGISTERS
	020552	017700	003426			MOV	@CIR, R0		; SAVE CIR CONTENT
	020556	017701	003424			MOV	@SMR, R1		; SAVE SMR CONTENT
	020562	017702	003424			MOV	@IOR, R2		; SAVE IOR CONTENT
1025	020566					ERROR	76.		; NO <SPAS> TO <SPE>
	020566	104514				TRAP	76.		; ERROR 76.
	020570	000^70				.WORD	0		; ERROR COUNT
1026	020572	000137	020176			JMP	TEST13		; RESTART SUBTEST
1027	020576	042777	177400	003400	10:	BIC	@177400, @CIR		; RESET ALL INTERRUPT CONDITIONS
1028	020604	012777	000100	003400		MOV	@BIT6, @IOR		; SEND STATUS BYTE
1029	020612				11:	TST15	@CIR		; WAIT FOR <DATA ACCEPTED>
	020612	032777	100000	003364		BIT	@BIT15, @CIR		
1030	020620	001774				BEQ	11:		
1031	020622					SAVIER			; SAVE IOR CONTENT
	020622	017702	003364			MOV	@IOR, R2		; SAVE IOR CONTENT
1032	020626	105002				CLRB	R2		; PREPARE FOR DATA CHECK
1033	020630	000302				SWAB	R2		
1034	020632					IF R2 EQ @BIT6 GOTO 12:			
	020632	020227	000100			CMP	R2, @BIT6		; COMPARE R2 AND @BIT6
	020636	001002				BNE	..6		
	020640	000137	020670			JMP	12:		
1035	020644					SAVIEC			; SAVE ALL IEC REGISTERS
	020644	017700	003334			MOV	@CIR, R0		; SAVE CIR CONTENT
	020650	017701	003332			MOV	@SMR, R1		; SAVE SMR CONTENT
	020654	017702	003332			MOV	@IOR, R2		; SAVE IOR CONTENT
1036	020660					ERROR	77.		; IEC DATA CHECK ON STATUS BYTE SENDING
	020660	104515				TRAP	77.		; ERROR 77.
	020662	000000				.WORD	0		; ERROR COUNT
1037	020664	000137	020176			JMP	TEST13		; RESTART TEST
1038	020670				12:	TST14	@CIR		; IS <SRQ> OFF ?
	020670	032777	040000	003306		BIT	@BIT14, @CIR		

1039	020676	001412				BEQ	13:		;IF EQ, YES
1040	020700					SAVIEC			;SAVE ALL IEC REGSITERS
	020700	017700	003300			MOV		@CIR,R0	;SAVE CIR CONTENT
	020704	017701	003276			MOV		@SMR,R1	;SAVE SMR CONTENT
	020710	017702	003276			MOV		@IOR,R2	;SAVE IOR CONTENT
1041	020714					ERROR		100.	; <SRQ> NOT OFF AFTER STATUS BYTE SENDING
	020714	104544				TRAP		100.	;ERROR 100.
	020716	000000				.WORD		0	;ERROR COUNT
1042	020720	000137	020176			JMP		TEST13	;RESTART TEST
1043	020724	042777	177400	003252	13:	BIC		@177400,@CIR	;RESET ALL INTERRUPT CONDITIONS
1044	020732					TCA			;ENTER <CACS>
	020732	052777	000002	003246		BIS		@BIT1,@SMR	;TAKE CONTROL ASYNC
1045	020740				14:	TSTB		@SMR	;WAIT FOR <CACS>
	020740	032777	000400	003240		BIT		@BITB,@SMR	
1046	020746	001774				BEQ		14:	
1047	020750	012777	000031	003234		MOV		@31,@IOR	;SEND <SERIAL POLL DISABLE>
1048	020756				15:	TST15		@CIR	;WAIT FOR <DATA ACCEPTED>
	020756	032777	100000	003220		BIT		@BIT15,@CIR	
1049	020764	001774				BEQ		15:	
1050	020766					GTS			;TRY TO GET <TACS> WHEN ENTERING <CSBS>
	020766	052777	000004	003212		BIS		@BIT2,@SMR	;GO TO STAND BY
1051	020774	005037	021012			CLR		17:	
1052	021000				16:	TST11		@SMR	;WAIT FOR <TACS> TO BECOME TRUE
	021000	032777	004000	003200		BIT		@BIT11,@SMR	
1053	021006	001015				BNE		18:	
1054	021010	005227				INC		(PC).	
1055	021012	000000			17:	.WORD		0	
1056	021014	001371				BNE		16:	
1057	021016					SAVIEC			;SAVE ALL IEC REGISTERS
	021016	017700	003162			MOV		@CIR,R0	;SAVE CIR CONTENT
	021022	017701	003160			MOV		@SMR,R1	;SAVE SMR CONTENT
	021026	017702	003160			MOV		@IOR,R2	;SAVE IOR CONTENT
1058	021032					ERROR		101.	;NO <TACS> AFTER <SPD>
	021032	104545				TRAP		101.	;ERROR 101.
	021034	000000				.WORD		0	;ERROR COUNT
1059	021036	000137	020176			JMP		TEST13	;RESTART SUBTEST
1060	021042				18:	MCLEAR			;MASTER CLEAR
	021042	052777	000040	003134		BIS		@BIT5,@CIR	;EXECUTE MASTER CLEAR
1061	021050					LOOP		-1,LOPT10	
	021050	005327				DEC		(PC).	;COUNT LOOPS
	021052	177777			65:	.WORD		-1	
	021054	005737	021052			TST		65:	;IS 65: ZERO ?
	021060	001402				BEQ		.0	
	021062	000137	020214			JMP		LOPT10	
	021066	012737	177777	021052		MOV		@-1,65:	;SETUP NEW LOOP COUNT
1062	021074					TSTEND		10	;END OF TEST 13
	021074	005737	011670			TST		MSGFLG	;PRINTOUT OR NOT?
	021100	001004				BNE		67:	
	021102	012746	023674			MOV		@MTE10,(SP)	;SAVE @MTE10 ONTO STACK
	021106	004737	005654			CALL		\$.WRT	;INITIATE PRINTOUT
	021112	000207			67:	RETURN;			;RETURN TO CALLER

1064					.SBTTL	TEST 14	- TEST RPP BIT R/W
1065	021114	005737	011670		TEST14: TST	MSGFLG	;PRINTOUT OR NOT?
1066	021120	001004			BNE	LOPT11	
1067	021122				WRITE	MT115	;START MESSAGE TEST 14
	021122	012746	023711		MOV	#MT115, -(SP)	;SAVE #MT115 ONTO STACK
	021126	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
1068	021132	104000			LOPT11: SCOPE		;SCOPE LOOP MARKER
1069	021134				MCLEAR		;MASTER CLEAR
	021134	052777	000040	003042	BIS	#BIT5, #CIR	;EXECUTE MASTER CLEAR
1070	021142				SACS		;SET <CONTROLLER ACTIVE> STATE
	021142	052777	000001	003034	BIS	#BIT0, #CIR	;SET SACS BIT IN CIR
1071	021150				SIC		;SEND <INTERFACE CLEAR>
	021150	052777	000100	003030	BIS	#BIT6, #SMR	;SEND INTERFACE CLEAR
1072	021156				36: TST13	#SMR	;WAIT FOR <SIAS TO GO OFF
	021156	032777	020000	003022	BIT	#BIT13, #SMR	
1073	021164	001374			BNE	36	
1074	021166				SET3	#SMR	;TRY TO SET <RPP>
	021166	052777	000010	003012	BIS	#BIT3, #SMR	
1075	021174				TST3	#SMR	;IS <RPP> SET ?
	021174	032777	000010	003004	BIT	#BIT3, #SMR	
1076	021202	001012			BNE	18	;IF NE, YES
1077	021204				SAVIEC		;SAVE ALL IEC-REGISTERS
	021204	017700	002774		MOV	#CIR, R0	;SAVE CIR CONTENT
	021210	017701	002772		MOV	#SMR, R1	;SAVE SMR CONTENT
	021214	017702	002772		MOV	#IOR, R2	;SAVE IOR CONTENT
1078	021220				ERROR	102.	; <RPP> NOT SET
	021220	104546			TRAP	102.	;ERROR 102.
	021222	000000			.WORD	0	;ERROR COUNT
1079	021224	000137	021114		JMP	TEST14	;RESTART TEST
1080	021230				18: CLR3	#SMR	;TRY TO RESET <RPP>
	021230	042777	000010	002750	BIC	#BIT3, #SMR	
1081	021236				TST3	#SMR	;IS BIT RESET ?
	021236	032777	000010	002742	BIT	#BIT3, #SMR	
1082	021244	001412			BEQ	28	;IF EQ, YES
1083	021246				SAVIEC		;SAVE ALL IEC-REGISTERS
	021246	017700	002732		MOV	#CIR, R0	;SAVE CIR CONTENT
	021252	017701	002730		MOV	#SMR, R1	;SAVE SMR CONTENT
	021256	017702	002730		MOV	#IOR, R2	;SAVE IOR CONTENT
1084	021262				ERROR	103.	; <RPP> NOT RESET
	021262	104547			TRAP	103.	;ERROR 103.
	021264	000000			.WORD	0	;ERROR COUNT
1085	021266	000137	021114		JMP	TEST14	;RESTART TEST
1086	021272				28: LOOP	1, LOPT11	
	021272	005327			DEC	(PC).	;COUNT LOOPS
	021274	177777			648: .WORD	1	
	021276	005737	021274		TST	648	;IS 648 ZERO ?
	021302	001401			BEQ	..4	
	021304	000712			BR	LOPT11	
	021306	012737	177777	021274	MOV	# 1, 648	;SETUP NEW LOOP COUNT
1087	021314				TSTEND	11	
	021314	005737	011670		TST	MSGFLG	;PRINTOUT OR NOT?
	021320	001004			BNE	668	
	021322	012746	023752		MOV	#MTE11, (SP)	;SAVE #MTE11 ONTO STACK
	021326	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
	021332	000207			668: RETURN;		;RETURN TO CALLER

```

1089
1090 021334      IECINT: .SBTTL IEC INTERRUPT HANDLER
      021334 010046      PUSH R0,R1,R2
      021336 010146      MOV  R0,-(SP)      ;SAVE R0 ONTO STACK
      021340 010246      MOV  R1,-(SP)      ;SAVE R1 ONTO STACK
      021342      MOV  R2,-(SP)      ;SAVE R2 ONTO STACK
1091 021342      SAVIEC
      021342 017700 002636      MOV  @CIR,R0      ;SAVE CIR CONTENT
      021346 017701 002634      MOV  @SMR,R1      ;SAVE SMR CONTENT
      021352 017702 002634      MOV  @IOR,R2      ;SAVE IOR CONTENT
1092 021356 000300      SWAB  R0      ;BRING INTERRUPT STATUS INTO LOW BYTE
1093 021360 042700 000006      BIC  @6,R0      ;MASK OUT BITS 9 AND 10 OF CIR
1094 021364 000305      SWAB  R5      ;BRING INTERRUPT MASK INTO LOW BYTE
1095 021366      CLRO  10(SP)      ;ASSUME ALL IS OK
      021366 042766 000001 000010      BIC  @BIT0,10(SP)
1096 021374 120500      CMPB  R5,R0      ;SUSPECTED INTERRUPT ?
1097 021376 001411      BEQ   1$      ;IF EQ, YES
1098 021400 000300      SWAB  R0      ;REBUILD SAVED CIR CONTENT
1099 021402 000305      SWAB  R5      ;REBUILD 'SHOULD BE' CONTENT
1100 021404 062706 000012      ADD  @12,SP      ;REBUILD STACK POINTER
1101 021410      POP   R0,R1,R2      ;REBUILD R0,R1 AND R2 ;**V03E01**
      021410 012600      MOV  (SP)+,R0      ;RESTORE R0 FROM STACK
      021412 012601      MOV  (SP)+,R1      ;RESTORE R1 FROM STACK
      021414 012602      MOV  (SP)+,R2      ;RESTORE R2 FROM STACK
1102 021416 000261      SEC
1103 021420 000114      JMP   (R4)      ;SET ERROR INDICATOR ;**V03E01**
1104 021422 010466 000006      1$: MOV  R4,6(SP)      ;RETURN
1105 021426 013704 024204      MOV  CIR,R4      ;BUILD RETURN ADDRESS
1106 021432 005204      INC   R4      ;PREPARE TO CLEAR INTERRUPT BITS
1107 021434 105014      CLRB  (R4)      ;
1108 021436      POP   R2,R1,R0      ;CLEAR INTERRUPT BITS
      021436 012602      MOV  (SP)+,R2      ;RESTORE R2 FROM STACK
      021440 012601      MOV  (SP)+,R1      ;RESTORE R1 FROM STACK
      021442 012600      MOV  (SP)+,R0      ;RESTORE R0 FROM STACK
1109 021444 000002      RTI
1110 021446 062706 000004      TOUT: ADD  @4,SP      ;RETURN FROM INTERRUPT
1111 021452      WRITE MNOAV      ;INCREASE SP BY 4
      021452 012746 023767      MOV  @MNOAV,-(SP)      ;
      021456 004737 005654      CALL $.WRT      ;SAVE @MNOAV ONTO STACK
1112 021462      DUMP  OCT,R0      ;INITIATE PRINTOUT
      021462 010046      MOV  R0,(SP)      ;
      021464 005046      CLR   (SP)      ;SAVE R0 ONTO STACK
      021466 004737 006400      CALL $.DMP      ;ENTER DUMP ROUTINE
1113 021472 005737 007152      1$: TST  OUTFLG      ;
1114 021476 001375      BNE   1$      ;
1115 021500 012737 000006 0C0004      MOV  @6,@4      ;
1116 021506 000111      JMP   (R1)      ;

```

IEC11 A DIAGNOSTIC
SUBTEST ADDRESS TABLE

MACRO M1200 30 MAR 84 16:02 PAGE 50

SEQ 99

1118

.SBTTL SUBTEST ADDRESS TABLE

```

1120 .SBTTL MESSAGES
1121 .MLIST BEX
1122 021510 133 105 116 MES1: .ASCII /[[ENTER FIRST REGISTER ADDRESS OF IEC11 A/
1123 021560 133 050 104 .ASCII /[[DEFAULT IS 160010]]/
1124 .EVEN
1125 021606 133 111 105 M1: .ASCII /[[IEC11 A BUS CONTROLLER BASIC TESTS/
1126 021651 133 101 103 .ASCII /[[AC-T883A MC]]/
1127 .EVEN
1128 021666 133 105 116 MES2: .ASCII /[[ENTER VECTOR ADDRESS OF IEC11 A/
1129 021726 133 050 104 .ASCII /[[DEFAULT IS 270]]/
1130 .EVEN
1131 021750 133 105 116 MES5 .ASCII /[[ENTER IEC BUS ADDRESS OF IEC11 A/
1132 022012 133 050 104 .ASCII /[[DEFAULT IS 35]]/
1133 .EVEN
1134 022034 133 126 105 MES3: .ASCII /[[VECTOR INPUT IS NOT EQUAL TO CONTENTS OF VSR]]/
1135 .EVEN
1136 022112 133 133 123 SLON: .ASCII /[[SET LON SWITCH ON IEC11-A TO POS.1 (ACTIVE)]/
1137 022167 133 124 131 .ASCII /[[TYPE <CR> WHEN READY]]/
1138 022216 133 133 123 CLON: .ASCII /[[SET LON SWITCH ON IEC11-A TO POS.2 (INACTIVE)]/
1139 022275 133 124 131 .ASCII /[[TYPE <CR> WHEN READY]]/
1140 022324 133 133 123 MT1S: .ASCII /[[START TEST 1: EXECUTE MASTER CLEAR]]/
1141 022371 133 105 116 MTE1: .ASCII /[[END TEST 1]]/
1142 022405 133 133 123 MT2S: .ASCII /[[START TEST 2: TEST SACS BIT IN CIR]]/
1143 022452 133 105 116 MTE2: .ASCII /[[END TEST 2]]/
1144 022466 133 133 123 MT2AS: .ASCII /[[START TEST 3: TEST CIR BIT 10 INHIBIT SRQ INTERRUPT]]/
1145 022554 133 105 116 MTE2A: .ASCII /[[END TEST 3]]/
1146 022570 133 133 123 MT3S: .ASCII /[[START TEST 4: TEST SACS SIC SIAS ILLMSG CACS]]/
1147 022647 133 105 116 MTE3: .ASCII /[[END TEST 4]]/
1148 022663 133 133 123 MT3AS: .ASCII /[[START TEST 5: TEST INT ENB]]/
1149 022720 133 105 116 MTE3A: .ASCII /[[END TEST 5]]/
1150 022734 133 133 123 MT4S: .ASCII /[[START TEST 6: TEST INTERRUPT WITH STATE CHGE]]/
1151 023013 133 105 116 MTE4: .ASCII /[[END TEST 6]]/
1152 023027 133 133 123 MT5S: .ASCII /[[START TEST 7: TEST STATE CHGE INTERRUPT WITH SRAS]]/
1153 023113 133 105 116 MTE5: .ASCII /[[END TEST 7]]/
1154 023127 133 133 123 MT5AS: .ASCII /[[START TEST 8: IOR DATA TEST]]/
1155 023165 133 105 116 MTE5A: .ASCII /[[END TEST 8]]/
1156 023201 133 133 123 MT6S: .ASCII /[[START TEST 9: TEST STATE CHGE INTERRUPT WITH GTS AND SIC]]/
1157 023274 133 105 116 MTE6: .ASCII /[[END TEST 9]]/
1158 023310 133 133 123 MT7S: .ASCII /[[START TEST 10: TEST DATA TRANSFER WITH INTERRUPT]]/
1159 023373 133 105 116 MTE7: .ASCII /[[END TEST 10]]/
1160 023410 133 133 123 MT8S: .ASCII /[[START TEST 11: TEST FUNCTION OF <LTN> AND <LUN>]]/
1161 023472 133 105 116 MTE8: .ASCII /[[END TEST 11]]/
1162 023507 133 133 123 MT9S: .ASCII /[[START TEST 12: TEST FUNCTION OF <BLOCK DAC>]]/
1163 023565 133 105 116 MTE9: .ASCII /[[END TEST 12]]/
1164 023602 133 133 123 MT10S: .ASCII /[[START TEST 13: TEST FUNCTION OF <RSV>, <SRQ>, AND <SPAS>]]/
1165 023674 133 105 116 MTE10: .ASCII /[[END TEST 13]]/
1166 023711 133 133 123 MT11S: .ASCII /[[START TEST 14: TEST RPP BIT RW]]/
1167 023752 133 105 116 MTE11: .ASCII /[[END TEST 14]]/
1168 023767 133 133 124 MNOAV: .ASCII /[[THIS ADDRESS IS NOT AVAILABLE ON UNIBUS: ]]/
1169 024044 133 133 116 MESNEX: .ASCII /[[NEXT TEST TO RUN?]]/
1170 .EVEN
1171 024070 133 133 104 TPMSG: .ASCII /[[DO YOU WANT A PRINTOUT OF START,END TEST MESSAGES?(Y OR N)]]/
1172 .EVEN
1173 024166 133 133 105 MESEND: .ASCII /[[END OF TEST]]/
1174 .EVEN
1175 .LIST BEX
1176 .EVEN

```

1178					
1179	024204	160010	CIR: .WORD	160010	; ADDRESS OF IEC CONTROL AND INTERRUPT REGISTER
1180	024206	160012	SMR: .WORD	160012	; ADDRESS OF IEC STATUS AND MESSAGE REGISTER
1181	024210	160015	IORMB: .WORD	160015	; IOR HIGH-BYTE ADDRESS DEFAULT VALUE
1182	024212	160014	IOR: .WORD	160014	; ADDRESS OF IEC INPUT/OUTPUT REGISTER
1183	024214	160016	VSR: .WORD	160016	; ADDRESS OF IEC VECTOR SWITCH REGISTER
1184	024216	000100	TALKER: .WORD	100	; ADDRESS OF TALKER
1185	024220	000040	LISTNR: .WORD	40	; ADDRESS OF LISTENER
1186	024222	000270	VECTOR: .WORD	270	; ADDRESS OF IEC-VECTOR
1187	024224	000270	VECA: .WORD	270	; DEFAULT OF IEC11-A VECTOR ADDRESS
1188	024226	000035	IECAAD: .WORD	35	; IEC11 A BUS ADDRESS DEFAULT VALUE
1189		001204	.END	START	

ADDRIN	011170	LAB11	011432	MTE5	023113	SFLAG	011426	VECA	024224
BIT0	= 000001	LISTNR	024220	MTE5A	023165	SINGLE	011506	VECAIN	011070
BIT1	= 000002	LOPT1	012124	MTE6	023274	SISDR0	= 172200	VECTOR	024222
BIT10	= 002000	LOPT10	020214	MTE7	023373	SLON	022112	VSR	024214
BIT11	= 004000	LOPT11	021132	MTE8	023472	SMR	024206	WRT..E	006060
BIT12	= 010000	LOPT2	012266	MTE9	023565	SPEFLG	007170	X	= 000152
BIT13	= 020000	LOPT2A	012474	MT1S	022324	SR0	= 177572	XX	= 000004
BIT14	= 040000	LOPT3	012716	MT10S	023602	SR3	= 172516	Z##Z	= 000004
BIT15	= 100000	LOPT3A	013320	MT11S	023711	START	001204	Z###Z	= 000000
BIT2	= 000004	LOPT4	013500	MT2AS	022466	STATUS	007164	\$MNOAV	010351
BIT3	= 000010	LOPT5	013772	MT2S	022405	TALKER	024216	\$POINT	006614
BIT4	= 000020	LOPT5A	014412	MT3AS	022663	TESTAL	011536	\$SHLD	007176
BIT5	= 000040	LOPT6	014552	MT3S	022570	TESTL1	011612	\$TABLE	006616
BIT6	= 000100	LOPT7	015434	MT4S	022734	TEST0	011740	\$WAS	007200
BIT7	= 000200	LOPT7A	016402	MT5AS	023127	TEST1	012106	##MTYP	007020
BIT8	= 000400	LOPT8	017064	MT5S	023027	TEST10	015400	##PTYP	007016
BIT9	= 001000	LOPT9	017526	MT6S	023201	TEST11	017046	##SPE	003520
CDM..E	010614	MAPER	010472	MT7S	023310	TEST12	017510	\$.ASF	007174
CIR	024204	MEMDEF	010426	MT8S	023410	TEST13	020176	\$.ADP	006706
CLON	022216	MEMEND	007022	MT9S	023507	TEST14	021114	\$.ASC	005464
CMODE	= 140000	MERCOV	007737	MVERSN	010156	TEST2	012250	\$.BIN	005504
DMP..E	006674	MERHLT	010002	M..DET	001662	TEST3	012456	\$.BUF	005444
DUMMES	007210	MERROR	007652	M..HIM	001766	TEST4	012700	\$.DBN	006630
D..TYP	006420	MERRP2	007662	M..MMH	002006	TEST5	013302	\$.DDC	006520
EMTTBL	002416	MER1	010304	M..RET	001726	TEST6	013444	\$.DEC	005270
EMTTND	002424	MER2	010333	M..SAV	001424	TEST7	013754	\$.DMP	006400
EMT0	002424	MESEND	024166	M..TRP	001776	TEST8	014372	\$.DOC	006426
EMT1	002432	MESNEX	024044	M..TSE	001374	TEST9	014516	\$.EMT	002326
EMT2	002470	MES1	021510	M..TST	001516	TKB	= 177562	\$.ER	007166
ERRFLG	007154	MES2	021666	M1	021606	TKS	= 177560	\$.IAA	004120
IAD..E	004302	MES5	021750	N	= 000003	TOUT	021446	\$.IAD	004210
IAE..E	005016	MES6	022034	NOR.ER	004376	TPB	= 177566	\$.IAE	004472
IAS..E	004466	MILADR	007445	NOR..E	004372	TPMSG	024070	\$.IAS	004402
IAY..E	004116	MILBIN	007360	NO6	010612	TPS	= 177564	\$.IAY	004010
IECAAD	024226	MILDEC	007316	OFFCNT	011664	TSTREP	011666	\$.INP	003574
IECINT	021334	MILEMT	007421	OFFSET	011504	TST10A	016356	\$.IOT	003056
INFLAG	007150	MILOCT	007256	OUTFLG	007152	TT	011626	\$.KBI	003424
INPREQ	007156	MILSCP	007524	PMODE	= 030000	TTE	011666	\$.KBO	006122
INP..E	003776	MILTR1	010104	PROMES	007202	TYP..E	001252	\$.MAP	002116
IOR	024212	MILTR2	010144	PS	= 177776	UBMPR	= 170200	\$.MSP	005700
IORHB	024210	MKBOVF	007213	PSDSWR	007160	UDSAR0	= 177660	\$.NOR	004304
I..DMP	= 000001	MNOAV	023767	RA	011734	UDSDR0	= 177620	\$.OCT	005112
KBBEND	007146	MNOSUB	010032	RANDOM	011672	UISAR0	= 177640	\$.PRO	006136
KBBPNT	007146	MPWRFL	007704	RANMOD	016604	UISAR4	= 177650	\$.PWR	003126
KBBUFF	007026	MSGFLG	011670	RB	011736	UISAR5	= 177652	\$.RED	005020
KDSAR0	= 172360	MSYERR	007503	RED..E	005626	UISAR6	= 177654	\$.RRS	003400
KSDSR0	= 172320	MTE1	022371	RGAIN	010624	UISAR7	= 177656	\$.RSV	003360
KISAR0	= 172340	MTE10	023674	RUBFLG	007162	UISDR0	= 177600	\$.SPC	011342
KISAR5	= 172352	MTE11	023752	R..TYP	= 000001	UISDR4	= 177610	\$.STX	006262
KISAR6	= 172354	MTE2	022452	SAV	011164	UISDR5	= 177612	\$.TOT	006754
KISAR7	= 172356	MTE2A	022554	SCOPAD	007172	UISDR6	= 177614	\$.TRP	002524
KISDR0	= 172300	MTE3	022647	SCOPE	= 104000	UISDR7	= 177616	\$.TYP	001242
KISDR6	= 172314	MTE3A	022720	SEQ	011552	U..DET	001760	\$.WRT	005654
KISDR7	= 172316	MTE4	023013	SEQPNT	011550				

. ABS.	024230	000
	000000	001

IEC11 A DIAGNOSTIC MACRO M1200 30 MAR 84 16:02 PAGE 52-2
SYMBOL TABLE

SEQ 103

CSSMON 000000 002
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 14133 WORDS (56 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:02:12
ZIEBA0,ZIEBA0/-SP=NEWMAC/ML,CDM,ZIEBA0
.