

IDENTIFICATION
.....

PRODUCT CODE: AC-T883A-MC
PRODUCT TITLE: CZIEBAO 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 TRANSFERED.
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

E 1

SEQ 4

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 3ETC.

.

.

.

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 = SMR
; ERROR 11 SIAS BIT 0 TO 1 CHANGE DOES NOT PRODUCE A
; 'STATE CHANGE' .
; RO = CIR R1 = SMR
; ERROR 12 STAS 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
; RO = 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.
; RO = 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
; RO = ACTUAL CONTENT OF CIR

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

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

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

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

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

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

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

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

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

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

; ERROR 36 <CSBS> SETTING DOES NOT PRODUCE <STATE CHANGE>.
; RO = 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 INTEHRUPT 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> C'N 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 *
* UPDATE TO V03E03 BY G. SZENTES *
* UPDATE 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.

C) USING 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 PSFUDO-SWITCH REGISTER.
 THE FOLLOWING OPTIONS ARE AVAILABLE:

/HY	=	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.

B) 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	\$.PRO
23-	1285	\$.STX
24-	1317	\$.DMP
26-	1418	\$.ADP
27-	1437	\$.TOT
28-	1463	\$.BUF
30-	62	TEST SELECT ROUTINE
32-	107	LOCAL MACRO DEFINITIONS
33-	209	TEST LON SWITCH FEATURE
34-	234	TEST 1 EXECUTE MASTER CLEAR
35-	255	TEST 2 TEST SACS BIT IN CIR
36-	284	TEST 3 TEST CIR BIT 10 INHIBIT SRQ INTERRUPT
37-	314	TEST 4 TEST SACS SIC SIAS ILLMSG CACS
38-	379	TEST 5 TEST INT ENB
39-	402	TEST 6 TEST INTERRUPT WITH STATE CHGE
40-	444	TEST 7 TEST STATE CHGE INTERRUPT WITH SRAS
41-	514	TEST 8 - IOR DATA TEST
42-	534	TEST 9 TEST STATE CHGE INTERRUPT WITH GTS AND SIC
43-	646	TEST 10 TEST DATA TRANSFER WITH INTERRUPT
44-	770	TEST 10A -- RANDOM TALKER AND LISTENER DATA TEST
45-	849	TEST 11 -- TEST FUNCTION OF <LTN> AND <LUN>
46-	910	TEST 12 -- TEST FUNCTION OF <BLOCK DAC>
47-	970	TEST 13 -- TEST FUNCTION OF <RSV>, <SRQ> AND <SPAS>
48-	1064	TEST 14 --- TEST RPP BIT R/W
49-	1089	IEC-INTERRUPT HANDLER
50-	1118	SUBTEST ADDRESS TABLE
51-	1120	MESSAGES
52-	1178	CONSTANTS AND PARAMETERS

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.  
;  
;
```



```

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      $KBBUFF,RO      ;RESET ALL FLAGS
69 001306 005020      1$: CLR      (RO),      ;
70 001310 020027 007202      CMP      RO,$PROMES      ;
71 001314 001374      BNE      1$      ;
72 001316 012737 001000 007172      MOV      $1000,$SCOPAD      ;IF NO SCOPE LABEL IS SET IN THE PROGRAM
73      ; THEN GOTO START AFTER ERR=/$C
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,RO      ;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      -(RO)      ;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      RO,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      RO      ;ROTATE ADDR16 INTO MSB OF RO      ;BB001
91 001440 006201      ASR      R1      ;PUT ADDR17 INTO CARRY      ;BB001
92 001442 006000      ROR      RO      ;ROTATE ADDR17 INTO MSB OF RO      ;BB001
93 001444 000241      CLC      ;RESET CARRY      ;BB001
94 001446 006000      ROR      RO      ;SHIFT OUT THE LS-OCT-DIGIT OF PHYS ADDR ;BB001
95 001450      DUMP      OCT,RO      ;DUMP ADDRESS
96 001460      WRITE     NO6      ;PRINT LS-OCT-DIGIT ALWAYS AS "6"      ;BB001
97 001470 052737 000100 177560      RIS      $BIT6,$@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:          ; REFERENCE LABEL
123          001516 012737 001776 000250  MOV    #M..TRP,#0250  ; SETUP SEGMENTATION TRAP CATCHER
124          001524 005037 000252          CLR    #0252          ; ...
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..MM        ; 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,MMTYP      ; 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$:  TST    #R0        ; SEC IF ADDRESS PRESENT
133          001576 103031          BCC   M..DET        ; CC IF ADDRESS FOUND
134          001600 162737 000200 172354  SUB    #200,KISAR6   ; MAP TO NEXT LOWER 4K WORD BANK
135          001606 002401          BLT   #30$         ; THIS CASE SHOULD NEVER HAPPEN
136          001610 000771          BR    #20$         ; CHECK NEXT LOWER 4-K WORDS
137          001612          30$:          ; HERE ONLY IF MAIN PROCESSOR HARDWARE ERROR
138          001612          PSET   #0           ; SET PRIORITY TO 0
139          001636          WRITE  MEMDEF        ; MEMORY MANAGEMENT DEFECT
140          001646 012716 001374          MOV    #M..TSE,(SP)  ; MODIFY RETURN ADDRESS TO UNMAPPED SYSTEM
141          001652 042737 000001 177572  BIC   #BIT0,SRO     ; DISABLE MM
142          001660 000422          BR    M..RET        ; AND GO AHEAD LIKE UNMAPPED SYSTEM
143          001662          M..DET:          ; HIGHEST MEMORY ADDRESS FOUND
144          001662          PUSH   KISAR6        ; GET MAP REGISTER CONTENTS
145          000005          .REPT  5          ;
146          ASL    (SP)          ; PUT ADDRESS BIT 17 INTO PSW CARRY
147          .ENDR
148          001700 103402          BCS   #40$         ; CS IF ADDRESS BIT 17 SET
149          001702 005001          CLR   R1           ; HERE IF BIT17 NOT SET
150          001704 000402          BR    #45$         ; ...
151          001706 012701 000001          40$:  MOV    #BIT0,R1    ; FLAG BIT 17 FOUND
152          001712 006316          45$:  ASL    (SP)      ; GET ADDRESS BIT 16
153          001714 006101          ROL   R1           ; INSERT BIT IN MEMORY EXTENSION MASK
154          001716 042700 160000          BIC   #160000,R0    ; MASK OUT APR SELECTION BITS
155          001722 050016          BIS   R0,(SP)      ; BUILD 16 BIT BASE ADDRESS
156          001724 012600          MOV   (SP)+,R0     ; -->R0 FOR OUTPUT TO CALLER
157          001726 012737 000252 000250  M..RET: MOV    #252,#0250 ; RESET SEGMENTATION TRAP CATCHER
158          001734 012737 000004 000252  MOV    #4,#0252     ; ...
159          001742 012737 000006 000004  MOV    #6,#04      ; RESET TRAP-4 CATCHER

```

```

160 001750 012737 000004 000006      MOV    #4,0#6      ;...
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..ISE,(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
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195 002006                    M..MMM:  ;
196 002006                    ;
197 002014 012700 077406                    ;
198 002020 012701 000010                    ;
199 002024 012702 172300                    ;
200 002030 010022                    ;
201 002032 005301                    ;
202 002034 001375                    ;
203 002036 012702 172340                    ;
204 002042 005022                    ;
205 002044 012722 000200                    ;
206 002050 012722 007400                    ;
207 002054 012722 000600                    ;
208 002060 012722 001000                    ;
209 002064 012722 001200                    ;
210 002070 012722 001400                    ;
211 002074 012712 007600                    ;
212 002100 052737 000001 177572    BIS    #BIT0,SRO  ;ENABLE MEMORY MANAGEMENT
213 002106                    POP    R2,R1,RO    ;RESTORE REGS.
214 002114 000207                    RETURN          ;BACK TO CALLER
215

```

SUBROUTINE M..MMM

THIS ROUTINE IS CALLED TO SET UP THE APR AND PDR REGISTER WITH DEFAULT VALUES AND TO ENABLE MEMORY MANAGEMENT.
INPUT: NONE
OUTPUT: ALL PDRS ARE PRESET WITH 77406
APRO IS PRESET WITH 0
APR1 " 200
APR2 " 400
.
APR7 " 7600
MM IS ENABLED

```

104:
PUSH    R0,R1,R2      ;SAVE
MOV     #77406,R0     ;GET PAGE DESCRIPTOR REGISTER SETUP
MOV     #8,R1         ;SETUP FOR 8 PDR'S
MOV     #KISDRO,R2    ;GET KISDRO ADDRESS
MOV     R0,(R2)+      ;SETUP NEXT PDR
DEC     R1            ;COUNT SETUPS
BNE     104           ;IF MORE TO DO
MOV     #KISAR0,R2    ;GET KISAR BASE ADDRESS
CLR     (R2)+         ;SETUP KISAR 0
MOV     #200,(R2)+    ;SETUP KISAR 1
MOV     #400,(R2)+    ;SETUP KISAR 2
MOV     #600,(R2)+    ;SETUP KISAR 3
MOV     #1000,(R2)+   ;SETUP KISAR 4
MOV     #1200,(R2)+   ;SETUP KISAR 5
MOV     #1400,(R2)+   ;SETUP KISAR 6
MOV     #7600,R2      ;SETUP KISAR 7 TO I/O PAGE
BIS     #BIT0,SRO     ;ENABLE MEMORY MANAGEMENT
POP     R2,R1,RO      ;RESTORE REGS.
RETURN          ;BACK TO CALLER
.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
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
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 CUM (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$:  CALL      M.###        ;ENABLE MM,SETUP DEFAULT MAPPING
      MOV       MEMEND,R0    ;SAVE HIGHEST MEMCRY LOCATION
      .REPT     6
      ROR      RO           ;COMPUTE 100 OCTAL BYTE BLOCKS
      .ENDR
      BIC      #176000,R0    ;"
      MOV      MEMEND+2,R1   ;SAVE MEMORY EXTENSION
      .REPT    10.
      ASL     R1            ;COMPUTE ADDITIONAL BLOCKS
      .ENDR
      AC     RO,R1          ;TOTAL BLOCKS OF 32.WORDS MEMORY
      BIC     #BIT0,#PS     ;CLEAR CARRY
      CMP     R1,6(SP)      ;IS MEMORY AVAILABLE?
      BGE     1$           ;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
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284 002326 012746 177776
285 002332 066616 000002
286 002336 017616 000000
287 002342 042716 177100
288 002346 006316
289 002350 021627 000004
290 002354 101413
291 002356
292 002366
293 002400 000137 001000
294 002404 062716 002416
295 002410 017616 000000
296 002414 000136
297 002416 002424
298 002420 002432
299 002422 002470
300 002424
301
302
303
304 002424 011637 007172
305 002430 000002
306
307
308
309 002432 017646 000010
310 002436 016676 000010 000000
311 002444 062716 000002
312 002450 016636 000006
313 002454 012666 000004
314 002460 012666 000004
315 002464 005726
316 002466 000002
317
318
319
320 002470 017646 000004
321 002474 011646
322 002476 062716 000002
323 002502 012636

```

```

.SBTTL $.EMT
;
; SUBROUTINE: $.EMT
; *****
;
; THIS ROUTINE HANDLES ALL EMT-TRAPS THAT MIGHT OCCUR
; IN A DIAGNOSTIC PROGRAM, AND DIRECTS THE PROGRAM TO
; CONTINUE AT THE APPROPRIATE FUNCTION SUBROUTINES.
;
; INPUT:      2(SP) = PS
;             (SP) = PC
; OUTPUT:     NONE
;
; AUTHOR:     BERT HUBER CSS/DP MUNICH 14-JULI-76
;
;
; .ENABL LSB
$.EMT: MOV  0-2,-(SP)      ;REBUILD EMT-ADDR.
      ADD  2(SP),(SP)    ;
      MOV  0(SP),(SP)    ;GET EMT-CODE
      BIC  0177400,(SP)  ;EXTRACT EMT-NR.
      ASL  (SP)          ;CONVERT TO TABLE-OFFSET
      CMP  (SP),0EMTTND-EMTTBL-2;VALID EMT-NR.?
      BLOS 1$           ;IF LOS, YES
      WRITE MILEMT      ;ILLEGAL EMT
      DUMP OCT,2(SP)    ;
      JMP  001000       ;RESTART PROGRAM
1$:   ADD  0EMTTBL,(SP)  ;GET FUCTION CODE ADDR.
      MOV  0(SP),(SP)   ;
      JMP  0(SP)+       ;ENTER FUCTION SUBROUTINE
EMTTBL: EMT0
        EMT1
        EMT2
EMTTND:
;
; SUBROUTINE TO SAVE SCOPE-LOOP ADDR.
;
EMT0:  MOV  (SP),SCOPAD  ;SAVE ADDR. OF SCOPE-LOOP
      RTI                ;RETURN TO CALLER
;
; SUBROUTINE TO SETUP A VECTOR
;
EMT1:  MOV  010(SP),-(SP) ;GET POINTER TO VECTOR ADDR.
      MOV  10(SP),0(SP)  ;SETUP VECTOR
      ADD  02,(SP)       ;POINT TO NEW PS
      MOV  6(SP),0(SP)+  ;SETUP NEW PS
      MOV  (SP)+,4(SP)   ;CLEAN UP STACK
      MOV  (SP)+,4(SP)   ;
      TST  (SP)+        ;
      RTI                ;RETURN TO CALLER
;
; SUBROUTINE TO RESET A VECTOR
;
EMT2:  MOV  04(SP),-(SP)  ;GET POINTER TO VECTOR A
      MOV  (SP),-(SP)    ;DUPLICATE IT
      ADD  02,(SP)       ;BUILD ADDRESS OF VECTOR
      MOV  (SP)+,0(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	4,0-4(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:  $.TRP
332          ; *****
333
334          ; THIS ROUTINE HANDLES ALL TRAPS TO LOC 34 THAT MIGHT
335          ; OCCUR IN A CSS-DIGNOSTIC PROGRAM.
336          ; AS THE TRAP INSTRUCTION IN THIS DIAGNOSTICS IS ONLY USED TO
337          ; INDICATE AN ERROR CONDITIC:, THIS TRAP-HANDLER IS A (PSEUDO-)
338          ; SWITCH-REGISTER CONTROLLED ERROR HANDLER. FOR THE DEFINITION
339          ; OF ALL AVAILABLE 'SWITCH'-OPTIONS, REFER TO SUBROUTINE "$.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          ;
350          ; .PSECT  CSSMON
351          ; .ENABL  LSB
352          $.TRP:  MOV    0-2,-(SP)      ;REBUILD TRAP ADDRESS
353                ADD    2(SP),(SP)      ;
354                MOV    0(SP),(SP)     ;GET TRAP CODE
355                BIC    0177400,(SP)   ;EXTRACT ERROR NUMBER
356                TST    $.ER           ;FIRST ERROR OCCURENCE ?
357                BEQ    2$              ;IF EQ, FIRST PASS
358                TST    PSDSWR         ;SCOPE LOOP ?
359                BPL    3$              ;
360                CMP    (SP),$.ER      ;SAME SCOPE LOOP ?
361                BEQ    3$              ;IF EQ, YES
362                WRITE  MILSCP          ;ILLEGAL SCOPE LOOP
363                DUMP   OCT,2(SP)       ;DUMP ADDRESS OF ERROR +2
364                GOTO   START           ;RESTART PROGRAM
365                MOV    (SP),$.ER      ;STORE ERROR NUMBER
366                BIT    0BIT14,PSDSWR  ;INHIBIT PRINTOUT ?
367                BNE   4$              ;IF NE, YES
368                WRITE  MERROR          ;PRINT ERROR MESSAGE PART 1
369                DUMP   DEC,(SP)        ;DUMP ERROR NUMBER
370                WRITE  MERRP2         ;PRINT ERROR MESSAGE PART 2
371                DUMP   OCT,2(SP)       ;DUMP ERROR PC +2
372                BIT    0BIT13,PSDSWR  ;COUNT ERRORS ?
373                BEQ    5$              ;IF EQ, NO
374                INC    02(SP)         ;COUNT !
375                JNE   5$              ;IF EQ, OVERFLOW
376                WRITE  MERCOV         ;ERROR COUNTER OVERFLOW
377                DUMP   DEC,(SP)        ;DUMP ERROR NUMBER
378                BLT   0BIT14,PSDSWR   ;PRINT ? ?
379                BNE   6$              ;IF NE, NO
380                BIT    0BIT15,PSDSWR  ;SCOPE LOOP ?
381                BNE   6$              ;IF SCOPE, NO PRINT
382                BIT    0BIT12,PSDSWR  ;HALT AFTER ERROR SELECTED
383                BNE   6$              ;IF NE, YES
384                WRITE  MERHLT         ;STOP MESSAGE
385                BIT    0BIT12,PSDSWR  ;HALT ?
386                BNE   7$              ;IF EQ, YES

```

N2

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
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428

003056
003066 162716 000002
003072
003102
003112
003124 000777

```

.SBTTL $.IOT
SUBROUTINE: $.IOT
*****
THIS SUBROUTINE HANDLES ILLEGAL TRAPS OR INTERRUPTS.
IF SUCH A TRAP OR INTERRUPT OCCURS, A MESSAGE IS
PRINTED, SHOWING THE VECTOR TRAPPED TO, AND THE PROGRAMS PC
WHERE IT WAS INTERRUPTED.
THE PROGRAM HALTS THEN, AND MUST BE RESTARTED.
INPUT:      6(SP)  =      PS OF PROGRAM
            4(SP)  =      PC OF PROGRAM
            2(SP)  =      PS OF VECTOR
            (SP)  =      PC OF VECTOR +2
OUTPUT:     MESSAGE TO THE OPERATOR
AUTHOR:     BERT HUBER, CSS/DP MUNICH  22-JUL-76
.PSECT CSSMON
.ENABL  LSB
$.IOT:  WRITE  MILTR1           ;SEND FIRST MESSAGE PART
        SUB    02,(SP)         ;COMPUTE VECTOR
        DUMP  OCT,(SP)        ;DUMP VECTOR ADDRESS
        WRITE  MILTR2           ;SEND SECOND MESSAGE PART
        DUMP  OCT,4(SP)       ;DUMP PROGRAM PC
        BR
        .DSABL  LSB

```

```

430          .SBTTL  $.PWR
431
432          ; SUBROUTINE:  $.PWR
433          ; *****
434
435          ; THIS ROUTINE IS DIRECTLY ENTERED AFTER EACH POWER-FAIL TO
436          ; HANDLE THE NECESSARY REGISTER SAVE OPERATIONS, AND IS USED
437          ; AFTER THE POWER-UP INTERRUPT TO RESTORE THE SAVED REGISTERS
438          ; AFTER A SUCCESSFUL POWER-FAIL RESTART, A SHORT MESSAGEW IS PRINTED.
439
440          ; AUTHOR:      BERT HUBER, CSS/DP MUNICH 27-JULY-76
441          ; MZ001      SAVE THE OUTFLAG IN 3$ AND RESTORE IT
442          ; *****    AFTER PRINTOUT OF THE POWERFAIL MESSAGE
443
444          ;
445          ; .PSECT  CSSMON
446          ; .ENABL  LSB
447 003126  $.PWR: PUSH  R0,R1,R2,R3,R4,R5
448 003142  010627  MOV   SP,(PC)+      ;SAVE STACK-POINTER
449 003144  000000  1$:   .WORD  0          ;STACKPOINTER-SAVE-AREA
450 003146  012737  003172  000024  MOV   02$,0024      ;SETUP VECTOR FOR RESTART
451 003154  013727  007152  MOV   OUTFLG,(PC)+  ;SAVE PRINT FLAG      ;MZ001
452 003160  000000  3$:   .WORD  0          ;SAVE FOR FLAG      ;MZ001
453 003162  013727  007150  MOV   INFLAG,(PC)+ ;SAVE INPUT FLAG
454 003170  000777  5$:   .WORD  0          ;SAVE WORD FOR INPUT FLAG
455 003172  005002  BR    .              ;WAIT FOR POWER DOWN
456 003174  005202  2$:   CLR   R2            ;DELAI
457 003176  020227  177777  INC   R2            ;THE
458 003202  001374  CMP   R2,017777    ;POWER UP START
459 003204  013706  003144  BNE   -.6          ;FOR 1SECOND
460 003210  005037  007152  MOV   1$,SP        ;REBUILD STACKPOINTER
461 003214  005037  007150  CLR   OUTFLG       ;ALLOW PRINTOUT OF PWRFL MES. ;MZ001
462 003220  012737  003126  000024  MOV   0$,$.PWR,0024 ;REBUJLD VECTOR SETUP
463 003226  POP   R5,R4,R3,R2,R1,R0
464 003242  013727  005700  MOV   $$.MSP,(PC)+ ;SAVE OLD MESSAGE POINTER
465 003246  000000  8$:   .WORD  0          ;SAVE WORD FOR OLD MESSAGE POINTER
466 003250  WRITE  MPWRFL    ;SEND POWER-FAIL-MESSAGE      ;MZ001
467 003260  005737  007152  4$:   TST   OUTFLG      ;WAIT FOR OUTPUT COMPLETE    ;MZ001
468 003264  001375  BNE   4$           ;          ;MZ001
469 003266  SET6  00TKS       ;ENABLE FURTHER TTY-INTERRUPTS
470 003274  013737  003160  007152  MOV   3$,OUTFLG    ;RESTORE FLAG IF IT WAS SET  ;MZ001
471 003302  005737  007152  TST   OUTFLG      ;IF AN OUTPUT BEFORE PWRFL WAS RUNNING
472 003306  001411  BEQ   7$          ;
473 003310  013737  003246  005700  MOV   8$,$.MSP    ;THEN RESTORE OLD MESSAGE POINTER
474 003316  SET6  00TPS       ;AND ENABLE INT IN TPS
475 003324  112737  000012  177566  MOVB  012,00TPB   ;AND START INTERRUPT
476 003332  005737  003166  7$:   TST   5$          ;IF AN INP FLG BEFORE PWRFL WAS NOT SET
477 003336  001407  BEQ   6$         ;GOTO 6$
478 003340  WRITE  PROMES    ;ELSE WRITE >
479 003350  013737  003166  007150  MOV   5$,INFLAG   ;RESTORE INPUT FLAG IF IT WAS SET
480 003356  000002  6$:   RTI          ;CONTINUE WITH PROGRAM
481          .DSABL  LSB
482          .SBTTL  $.RSV
483
484          ; SUBROUTINE:  $.RSV
485          ; *****
486

```

D3

IEC11-A DIAGNOSTIC
\$.RSV

MACRO M1200 30-MAR-84 16:02 PAGE 10-1

SEQ 29

```
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
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534

003400 005726
003402
003416 005726
003420 000176 177760

```

      .SBTTL  $.RRS
;
; SUBROUTINE:  $.RRS
; *****
;
; THIS SUBROUTINE RESTORES R5-R0 FROM THE STACK
;
; INPUT:      (SP) = CALLER PC
;             2(SP) = SAVED R5
;             4(SP) = SAVED R4
;             6(SP) = SAVED R3
;             10(SP) = SAVED R2
;             12(SP) = SAVED R1
;             14(SP) = SAVED R0
;             16(SP) = OLD CALLER PC
;
; OUTPUT:     REGISTERS ARE RESTORED AND STACK IS CLEANED
;
; AUTHOR:     BERT HUBER, CSS/DP MUNICH 12-APR-76
;
; .PSECT CSSMON
; .ENABL LSB
$.RRS: TST (SP), ;POINT TO SAVED R5
      POP R5,R4,R3,R2,R1,R0
      TST (SP), ;SKIP OLD CALLER PC
      JMP @-20(SP) ;RETURN TO CALLER
      .DSABL LSB

```

```

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 ;
552 ; .PSECT CSSMON
553 ; .ENABL LSB
554 $.KBI: TST OUTFLG ;;;OUTPUT PUNNING ?
555 BEQ 4$ ;;;IF NE, IGNORE INTERRUPT
556 PUSH R0 ;;;SAVE
557 CLR 00TKS ;;;INHIBIT INTERRUPT
558 MOV 00TKB,R0 ;;;GET CHARACTER
559 MOV 0100,00TKS ;;;RE-ENABLE
560 JIC 0200,R0 ;;;MASK OUT ASCII
561 CMP 03,R0 ;;; IS IT <CNTRL C> ?
562 BNE 13$ ;;;IF NOT, RETURN
563 MOV 07777,SPEFLG ;;;SET A SPECIAL FLAG
564 ;;;THIS FLAG IS USED TO FINISH
565 ;;;A RUNNING PRINTOUT AND THEN TO
566 ;;;REMEMBER THAT <CNTRL C> WAS HIT
567 13$: POP R0 ;;;REBUILD
568 BR 3$ ;;; RETURN TO CALLER
569 4$: CALL $.RSV ;;;SAVE ALL REGISTERS
570 MOV 00TKB,R0 ;;;GET INPUT BYTE
571 BIC 0200,R0 ;;;BUILD REAL ASCII
572 CMPB 03,R0 ;;;<CNTRL C> FUNCTION ?
573 BNE 1$ ;;;IF NE, NO
574 $$$SPE:
575 CALL $.PRO ;;;SEND PROMPT STRING
576 BR 2$ ;;;RETURN
577 1$: TST INFLAG ;;;INPUT FLAG SET ?
578 BEQ 2$ ;;;IF EQ, NO
579 5$: CALL $.INP ;;;PROCESS INPUT
580 CMP R0,015 ;;;WAS IT A <CR> ?
581 BNE 2$ ;;;IF NE, NO
582 CLR INFLAG ;;;RESET INPUT MODE FLAG
583 MOV 0KBBUFF,R1 ;;;REBUILD BUFFER POINTER
584 MOV R1,KBBPNT ;;; " "
585 CALL $.IAI ;;;ANALYZE INPUT
586 2$: CALL $.RRS ;;;RESTORE REGISTERS
587 3$: 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 ASCI-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  $..INP: MOV  KBBPNT,R1          ;;GET CURRENT BUFFER-POINTER
615 003600 012702 006122          MOV  #1,KBO,R2          ;;POINT TO KEYBOARD-ECHO
616 003604 022701 007146          CMP  #KBBEND,R1        ;;INPUT-BUFFER FULL?
617 003610 001015          BNE  1$              ;;IF NE, NO
618 003612 022700 000025          CMP  #25,R0          ;;<CNTRL U>?
619 003616 001415          BEQ  2$              ;;IF EQ, YES.
620 003620 122700 000177          CMPB #177,R0         ;;<DELETE>?
621 003624 001407          BEQ  1$              ;;IF EQ, YES.
622 003626 005037 007150          CLR  INFLAG          ;;RESET INPUTFLAG
623 003632          WRITE MKROVF          ;;PRINT OVERFLOW-MESS.
624 003642 000451          BR   7$              ;;PROMPT FOR INPUT
625 003644 122700 000025  1$:  CMPB  #25,R0          ;;<CNTRL U>?
626 003650 001436          BEQ  6$              ;;IF NE, NO
627 003652 122700 000177  2$:  CMPB  #177,R0         ;;<DELETE>?
628 003656 001016          BNE  4$              ;;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  #'/,RO          ;;PRINT "/"
633 003676 004712          CALL  @R2             ;;
634 003700 020127 007026  3$:  CMP  R1,#KBBUFF        ;;FULL LINE DELETED?
635 003704 001434          BEQ  INP,.E           ;;IF EQ, YES
636 003706 114100          MOVB -(R1),RO        ;;NO, REMOVE CHARACTER
637 003710 004712          CALL  @R2             ;;AND ECHO IT
638 003712 000431          BR   INP,.E         ;;WAIT FOR NEXT INPUT
639 003714 005737 007162  4$:  TST  RUBFLG          ;;RUBOUT MODE?
640 003720 001407          BEQ  5$              ;;IF EQ, NO
641 003722          PUSH  RO          ;;YES, SAVE CURRENT INPUT
642 003724 012700 000057          MOV  #'/,RO          ;;END OF RUBOUT
643 003730 004712          CALL  @R2             ;;ECHO "/"
644 003732 005037 007162          CLR  RUBFLG          ;;RESET RUBOUT MODE FLAG
645 003736          POP   RO          ;;GET CURRENT INPUT AGAIN
646 003740 004712  5$:  CALL  @R2             ;;ECHO IT

```

```

647 003742 110021          MOVB   RO,(R1)+      ;;;SAVE IN IPUT BUFFER
648 003744 000414          BR     INP..E        ;;;WAIT FOR NEXT INPUT
649 003746 012700 000136  6$:   MOV    @136,RO      ;;;ECHO "↑"
650 003752 004712          CALL   @R2           ;;;
651 003754 012700 000125  MOV    @'U,RO        ;;;ECHO "U"
652 003760 004712          CALL   @R2           ;;;
653 003762 005037 007162  CLR    RUBFLG       ;;;LEAVE RUBOUT MODE
654 003766 012701 007026  7$:   MOV    @KBBUF,R1   ;;;RESET INPUT BUFFER POINTER
655 003772 004737 006136  CALL   $ .PRO       ;;;PROMPT FOR INPUT
656 003776 010137 007146  INP..E: MOV   R1,KBBPNT ;;;SAVE POINTER
657 004002 116100 177777  MOVB   -1(R1),RO    ;;;SAVE LAST CHARACTER
658 004006 000207          RETURN                    ;;;RETURN TO CALLER
659          .DSABL   LSB

```

```

661          .SBTTL $..IAY
662
663          ;*
664          ;SUBROUTINE: $..IAY
665          ;*****
666          ;
667          ;LEGAL INPUT TYPES ARE:
668          ;
669          ;      ERR=          FOR ERROR-HANDLING
670          ;      DMP=          FOR CORE-DUMP OPTIONS
671          ;      SET=          FOR DEBUGGING OPTIONS
672          ;      NONFORMATTED INPUT FOR SPECIAL REQ.
673          ;
674          ;INPUT: (SP)=      RETURN PC
675          ;OUTPUT:          NONE
676          ;
677          ;AUTHOR:          BERT HUBER, CSS/DP MUNICH      28-JUN 76
678          ;
679          ;.PSECT CSSMON
680          .ENABL  LSB
681          $..IAY: CMPB  @15,(R1)          ;;; <CR>?
682          BEQ    IAY..E          ;;; IF EQ., SKIP ANALIZATION
683          3$:   TST    $,ASF          ;;; DONT ANALYZE IF ASCII INPUT RUNNING
684          BNE    IAY..E          ;;; IF NE, YES.
685          CMPB  @'D,(R1)          ;;; 'DMP' REQUESTED?
686          BNE    1$              ;;; IF NE, NO
687          CALL  $..IAD          ;;; ANALYZE DUMP REQUEST
688          BR    IAY..E          ;;; RETURN TO CALLER
689          1$:   CMPB  @'S,@R1          ;;; 'SET' REQUESTED?
690          BNE    2$              ;;; IF NE, NO
691          CALL  $..IAS          ;;; ANALYZE SET REQUEST
692          BR    IAY..E          ;;; RETURN TO CALLER
693          2$:   CMPB  @'E,@R1          ;;; 'ERR'-REQUEST?
694          BNE    4$              ;;; IF NE, NO
695          CALL  $..IAE          ;;; ANALYZE ERROR REQUEST
696          BR    IAY..E          ;;; RETURN TO CALLER
697          4$:   CMPB  @'A,@R1          ;;; 'ABO' REQUESTED ?
698          BNE    5$              ;;; IF NE, NO
699          CALL  $..IAA          ;;; ANALYZE ABO-REQUEST
700          BR    IAY..E          ;;; RETURN TO CALLER
701          5$:   TST    INPREQ          ;;; NO SYNTAX ERROR IF SPECIAL INPUT RUNNING
702          BNE    IAY..E          ;;; "
703          CALL  $..STX          ;;; SYNTAX-ERROR
704          IAY..E: RETURN          ;;; RETURN TO CALLER
          .DSABL  LSB

```


754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805

```

; .PSECT CSSMON
;
;
; .SBTTL $.IAD
;
; SUBROUTINE : $.IAD
; ***** : *****
;
; MZ001
; *****
; MODIFICATION TO ALLOW THE <CR> AFTER INPUT
; TO COMPLETE.
;
; THIS SUBROUTINE IS ENTERED WHEN A DUMP-REQUEST IS ANALYZED.
; DEPENDING ON HOW THE REQUEST IS TERMINATED, THERE ARE TWO
; WAYS FOR RETURN:
;
; A) WHEN TERMINATED BY <CR> THE CONTENT OF A LOCATION IS
; PRINTED AND THE RETURN IS PERFORMED
; B) WHEN INPUT IS TERMINATED BY A "/" THE MONITOR WAITS
; FOR INPUT AND LOADS THIS INPUT INTO THE EXAMINED LOCATION.
;
; INPUT : (SP) *RETURN ADDRESS
; OUTPUT : MODIFIED LOCATION IF REQUESTED
;
; AUTHOR: M. ZILLER, SS/DP MUNICH OCTOBER 1976
;
; -
; .PSECT CSSMON
; ENABL LSB
$.IAD: INC R1 ;POINT TO "M"
CMPB (R1),0'M ;IS IT A "M"
BNE 77$ ;NO IF NE
CMPB (R1),0'P ;IS IT A "P"
BNE 77$ ;NO IF NE
CMPB (R1),0' / ;IS IT A "/"
BNE 77$ ;NO IF NE
CALL $.NOR ;RETRIEFE INPUT
BCS 77$ ;IS CS, INPUT FORMAT ERROR
11$: TSTB @0TPS ;IS THE <CR> FINISHED ? ;MZ001
BPL 11$ ; ;MZ001
MOV @12,@0TPB ;PRINT A <LF
12$: TSTB @0TPS ;DONE ?
BPL 12$ ;WAIT FOR COMPLETION
DUMP OCT,(R0) ;AND PRINT
BR IAD..E ;GO TO END
77$: CALL $.STX ;WRITE SYNTAX ERROR MESSAGE
IAD..E: RETURN ;RETURN TO MAIN
;

```

005201		
122127	000115	
001027		
122127	000120	
001024		
122127	000057	
001021		
004737	004304	
103416		
105737	177564	
100375		
012737	000012	177566
105737	177564	
100375		
000402		
004737	006262	
000207		

```

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 ?
      BHI     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
NOR,.E: CLC              ;INDICATE SUCCESSFUL OPERATION
      RETURN
NOR,ER: SEC              ;INDICATE FAILURE
      RETURN
;
;
;+
; SUBROUTINE:  $.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)+,@'T    ;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 MACRO M1200 30-MAR-84 16:02 PAGE 17-1

SEQ 38

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

006262

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
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926

004472 010602
004474 012703 007160
004500 005004
004502 062701 000003
004506 122127 000075
004512 001137
004514 122127 000057
004520 001403
004522 005702
004524 001132
004526 000526
004530 005002
004532 005204
004534 121127 000116
004540 001002
004542 005201
004544 005004
004546 121127 000123
004552 001426
004554 121127 000103
004560 001437
004562 121127 000110
004566 001437
004570 121127 000120
004574 001106

```

:
:
:          .SBTTL  $.IAE
:
: SUBROUTINE:  $.IAE
:-----
:
: THIS ROUTINE CHECKS THE CORRECT SYNTAX OF
: AN "ERR"- INPUT STRING AND PROCESSES THE
: REQUESTED 'SWITCHREGISTER-OPTIONS'.
:
: LEGAL SWITCH OPTIONS ARE:
:
:          /HT      =      HALT AFTER ERROR (DEFAULT)
:          /NH      =      NO HALT AFTER ERROR
:          /PR      =      PRINT ERROR MESSAGE (DEFAULT)
:          /NP      =      INHIBIT ERROR PRINTOUT
:          /CT      =      COUNT ERROR
:          /NC      =      NO ERROR COUNTING (DEFAULT)
:          /SC      =      SCOPE LOOP ON THIS ERROR
:          /NS      =      NO SCOPE LOOP (DEFAULT)
:          /CONT    =      IMMEDIATE CONTINUE SWITCH
:
: INPUT:      R1      *START ADDR. OF TEXT STRING
: OUTPUT:     BIT-SETUP IN SOFTWARE-SWITCH-REGISTER
:
: AUTHOR:     BERT HUBER, CSS/DP MUNICH 28-JULY-76
:
:
:          .PSECT CSSMON
:          .ENABL  LSB
$.IAE: MOV      SP,R2          ;; FLAG FOR VERY FIRST CHARACTER
      MOV      @PDSWR,R3     ;; GET 'SWR' ADDRESS
      CLR      R4            ;; USE AS 'CHARACTER 1'
      ADD      @?,R1         ;; SKIP IDENTIFIER
      CMPB     (R1),@' '     ;; CORRECT SYNTAX?
      BNE      77$          ;; BRANCH IF NOT
1$:   CMPB     (R1),@'/'     ;; IS NEXT CHAR. A '/'
      BEQ      2$          ;; IF EQ, YES
      TST     R2            ;; VERY FIRST?
      BNE      77$          ;; IF YES: SYNTAX ERROR
      BR      11$          ;; ASSUME END OF STRING
2$:   CLR      R2            ;; NO LONGER VERY FIRST
      INC     R4            ;; SET FOR 'C1' FLAG
      CMPB     (R1),@'N     ;; IS IT THE '/N' OPTION?
      BNE      3$          ;; BRANCH IF NOT
      INC     R1            ;; ADVANCE TO NEXT CHARACTER
      CLR     R4            ;; RESET 'C1' FLAG
3$:   CMPB     (R1),@'S     ;; IS IT A 'S'?
      BEQ      5$          ;; BRANCH IF YES
      CMPB     (R1),@'C     ;; IS IT A 'C'?
      BEQ      7$          ;; BRANCH IF YES
      CMPB     (R1),@'H     ;; IS IT A 'H'?
      BEQ      9$          ;; BRANCH IF YES
      CMPB     (R1),@'P     ;; IS IT A 'P'?
      BNE      77$          ;; IF NO: SYNTAX ERROR

```

927	004576	005704				TST	R4	IS IT A 'P' ?
928	004600	001004				BNE	4#	BRANCH IF YES
929	004602	052713	040000			BIS	#BIT14,(R3)	SET SR-BIT FOR /NP
930	004606	005201		30#:		INC	R1	ADVANCE TO NEXT CHARACTER
931	004610	000741				BR	1#	CHECK FOR NEXT SWITCH
932	004612	005201			4#:	INC	R1	NEXT CHARACTER
933	004614	122127	000122			CMPB	(R1),#R	IS IT A '/PR' ?
934	004620	001074				BNE	7#	IF NO; SYNTAX ERROR
935	004622	042713	040000			BIC	#BIT14,(R3)	SET '/PR'-OPTION
936	004626	000732				BR	1#	CHECK NEXT SWITCH
937	004630	005704			5#:	TST	R4	IS IT 'S/' ?
938	004632	001003				BNE	6#	BRANCH IF YES
939	004634	042713	100000			BIC	#BIT15,(R3)	SETUP FOR '/NS' OPTION
940	004640	000762				BR	30#	TRY NEXT
941	004642	005201			6#:	INC	R1	NEXT CHARACTER
942	004644	122127	000103			CMPB	(R1),#C	IS IT A '/SC' ?
943	004650	001060				BNE	7#	IF NO; SYNTAX ERROR
944	004652	052713	100000			BIS	#BIT15,(R3)	SETUP FOR '/SC'-OPTION
945	004656	000716				BR	1#	CHECK NEXT SWITCH
946	004660	126127	000001	000117	7#:	CMPB	1(R1),#D	CHECK FOR '/CONT'
947	004666	001016				BNE	70#	IF NE, OTHER
948	004670	126127	000002	000116		CMPB	2(R1),#N	CHECK FOR '/CONT'
949	004676	001045				BNE	7#	IF NO; SYNTAX ERROR
950	004700	126127	000003	000124		CMPB	3(R1),#T	CHECK FOR '/CONT'
951	004706	001041				BNE	7#	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	11#	END THIS SCAN
955	004724	005704			70#:	TST	R4	IS IT A '/CT' ?
956	004726	001003				BNE	8#	BRANCH IF YES
957	004730	042713	020000			BIC	#BIT13,(R3)	SETUP FOR '/NC'-OPTION
958	004734	000724				BR	30#	TRY NEXT SWITCH
959	004736	005201			8#:	INC	R1	ADVANCE TO NEXT CHARACTER
960	004740	122127	000124			CMPB	(R1),#T	IS IT A '/CT'-OPTION ?
961	004744	001022				BNE	7#	IF NO; SYNTAX ERROR
962	004746	052713	020000			BIS	#BIT13,(R3)	SETUP FOR '/CT' OPTION
963	004752	000660				BR	1#	CHECK NEXT SWITCH
964	004754	005704			9#:	TST	R4	IS IT '/HT' ?
965	004756	001003				BNE	10#	BRANCH IF YES
966	004760	052713	010000			BIS	#BIT12,(R3)	SETUP FOR '/NP'-OPTION
967	004764	000710				BR	30#	TRY NEXT SWITCH
968	004766	005201			10#:	INC	R1	ADVANCE TO NEXT CHARACTER
969	004770	122127	000124			CMPB	(R1),#T	IS IT A '/HT' ?
970	004774	001006				BNE	7#	IF NO; SYNTAX ERROR
971	004776	042713	010000			BIC	#BIT12,(R3)	SETUP FOR '/HT'-OPTION
972	005002	000644				BR	1#	TRY NEXT SWITCH
973	005004	124127	000015		11#:	CMPB	(R1),#15	WAS LAST BYTE A <CR>?
974	005010	001402				BEQ	IAE,IE	BRANCH IF YES
975	005012	004737	006262		77#:	CALL	\$.SIX	SYNTAX ERROR
976	005016	000207			IAE,IE:	RETURN		RETURN TO CALLER
977						.DSABL	LSB	

979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035

005020 004737 006136

```

      .SBTTL  $.RED
;
; SUBROUTINE  $.RED
;-----
;
;*** ABOO
;   THE CARRY BIT IS SET IF A ZERO OCTAL INPUT (ONLY CR,NO DATA)
;   HAPPENS
;
; THIS SUBROUTINE HANDLES ALL TYPES OF SPECIAL INPUT REQUESTS
; SETUP BY THE PROGRAMMER THROUGH A ' READ ' MACRO.
; LEGAL INPUT TYPES THAT CAN BE HANDLED ARE:
;
; BINARY FORMAT (EX: 1001101101010001)
;-----
; INPUT CAN BE UP TO 16-DIGITS OF THE FORM 0 OR 1.
; LESS THAN 16 DIGIT INPUT IS FILLED WITH LEADING ZEROES
; INPUT IS RETURNED IN RO.
; THE PREVIOUS CONTENT OF RO WILL BE OVERWRITTEN.
;
; OCTAL FORMAT (EX:176523)
;-----
; INPUT CAN BE UP TO 6 DIGITS OF THE FORM 0-7.
; IN CASE OF A 6-DIGIT-WIDE INPUT,THE LEFTMOST(HIGH ORDER)
; DIGIT IS TRUNCTATED TO THE RIGHTMOST (LOW ORDER)BIT.
; IN CASE OF LESS THAN 6 DIGITS INPUT, LEADING ZERO DIGITS
; ARE INSERTED.
; THE RESULT IS RETURNED IN RO,HEREBY OVERWRITING
; THE PREVIOUS CONTENT OF RO.
;
; DECIMAL FORMAT (EX: 96003)
;-----
; INPUT CAN BE UP TO 5 DIGITS OF THE FORM 0-9.
; LESS THEN FIVE DIGITS INPUT ARE FILLED UP WITH
; LEADING ZEROE DIGITS.
; MAXIMUM NUMBER ALLOWED IS 32656.
; INPUT IS RETURNED IN RO, HEREBY OVERWRITING THE
; PREVIOUS CONTENT OF RO.
;
; ASCII FORMAT (EX: B6T#&L)
;-----
; INPUT CAN BE ANY LEGAL ASCII CHARACTER.
; IF INPUT INTO A BUFFER IS REQUESTED, UP TO 80
; CHARACTERS CAN BE INPUT.
; IF INPUT RETURN IN RO IS REQUESTED,ONLY THE FIRST
; CHARACTER WILL BE RETURNED IN THE LOW ORDER BYTE
; OF RO.
; OLD CONTENT OF RO IS OVERWRITTEN IN THIS CASE.
; IF THE ONLY INPUT IS <CR>, ZERO WILL BE RETURNED IN RO
; AS WELL AS IN THE FIRST BYTE OF THE SPECIFIED BUFFER.
;
; AUTHOR:          BERT HUBER, CSS/DP MUNICH          29 JUN-76
;
; $.RED: CALL  $.PRO          ;PRINT PROMPT STRING

```

```

$.RED
1036 005024 010637 007156      MOV     SP,INPREQ      ;SET SPECIAL REQUEST FLAG
1037 005030 026627 000002 000004  CMP     2(SP),#4      ;DETECT SPECIAL ASCII INPUT RUNNING
1038 005036 003402                BLE     100$         ;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     #KBBUFF,R1   ;GET INPUT POINTER
1045 005070 016602 000014      MOV     14(SP),R2    ;GET INPUT MODE INDICATOR
1046 005074 000172 005100      JMP     @2$(R2)      ;SELECT REQ. INPUT
1047 005100 005112                2$:      $.OCT
1048 005102 005270                $.DEC
1049 005104 005504                $.BIN
1050 005106 005444                $.BUF
1051 005110 005464                $.ASC
1052 005112 005000                3$:      $.OCT: CLR     R0          ;CLEAR INPUT POINTER
1053 005114 005037 005244      CLR     6$          ;RESET COUNTER
1054 005120 121127 000015      CMPB   @R1,#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   @R1,#60      ;LEGAL OCTAL DIGIT ?
1068 005200 103403                BLO     4$          ;IF LO, NO
1069 005202 121127 000067      CMPB   @R1,#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,@R1     ;BUILD BINARY
1075 005240 152103                BISB   (R1)+,R0     ;INSERT INTO BUFFER
1076 005242 005227                INC     (PC)+        ;COUNT INPUTS HANDLED
1077 005244 000000                6$:      .WORD  0          ;DIGIT COUNTER
1078 005246 121127 000015      CMPB   @R1,#15      ;END OF INPUT DETECTED?
1079 005252 001341                BNE     3$          ;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$:
1084 005266 000557                BR     RED..E       ;END OF INPUT
1085 005270 005000                8$:      $.DEC: CLR     R0          ;CLEAR TRANSFER BUFFER
1086 005272 005004                CLR     R4          ;RESET COUNTER
1087 005274 122721 000015      9$:      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$          ;CEP COUNTING
1091 005306 012701 007026      2$:      MOV     #KBBUFF,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          3$: WRITE MILDEC ;ILLEGAL DECIMAL INPUT
1095 005330          POP R5,R4,R3,R2,R1 ;REST. REGS
1096 005342 000626      BR $..RED ;RESTART INPUT
1097 005344 005704      4$: TST 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 005077 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$:   CMPB  @R1,@15 ;END OF INPUT ?
1153 005620 001343          BNE   5$          ;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 000072   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
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218

005654 005737 007152
005660 001375
005662 005737 007150
005666 001372
005670 010637 007152
005674 016627 000002
005700 000000
005702 012737 005736 000064
005710 012737 000340 000066
005716 012616
005720 052737 000100 177564
005726 005737 007152
005732 001375
005734 000207
005736 122737 000023 177562 1\$:
005744 001401
005746 000404
005750 122737 000021 177562 3\$:
005756 001367
005760
005760 127727 177714 000155 20\$:
005766 001434
005770 127727 177704 000155
005776 001022
006000 042737 000100 177564
006006 012737 000015 177566
006014 105737 177564
006020 100375
006022 005237 005700
006026 012737 000012 177566
006034 052737 000100 177564
006042 000002
006044 117737 177630 177566 30\$:
006052 005237 005700
006056 000002

```
.SBTTL $.WRT
;
; SUBROUTINE: $.WRT
; *****
;
; THIS ROUTINE IS CALLED TO SETUP A MESSAGE PRINTOUT ON CONSOLE
; ALL PRINTING IS CONTROLLED BY INTERRUPT EXCEPT <CR>, <LF>.
;
; INPUT:          2(SP)  = ADDRESS OF MESSAGE TEXT
;                (SP)  = CALLER PC
;
; OUTPUT:         MESSAGE ON TERMINAL
;
; AUTHOR:         BERT HUBER CSS/DP MUNICH      28-JUN-76
;
; MODIFIED BY:   JOHN LEVESQUE  CSS ISG  23-JAN-83
;
; MODIFICATION REQUIRED TO HANDLE XON AND XOFF
;
;
; .PSECT COMMON
; ENABL  LSB
$.WRT:  TST  OUTFLG      ;OUTPUT RUNNING ?
        BNE  $.WRT     ;IF NE, YES - WAIT
        TST  INFLAG    ;INPUT RUNNING ?
        BNE  $.WRT     ;IF NE, YES - WAIT
        MOV  SP,OUTFLG ;SET OUTPUTFLAG
        MOV  2(SP),(PC) ;GET MESSAGE POINTER
$.MSP:  .WORD  0        ;MESSAGE POINTER
        MOV  01$,0064  ;SETUP VECTOR
        MOV  0340,0066 ;"
        MOV  (SP)+,(SP) ;UPDATE STACK
        BIS  0100,00TPS ;ENABLE INTERRUPTS
        TST  OUTFLG    ;OUTPUT RUNNING?      ;**GS0001**
        BNE  -4        ;IF NE, YES->WAIT      ;**GS0001**
        RETURN        ;RETURN TO CALLER
        CMPB 023,00TKB ;: CHECK FOR XOFF
        BEQ  3$       ;: XOFF CONDITION WAIT TO CLEAR
        BR   20$      ;: NO XOFF THEN CONTINUE
        CMPB 021,00TKB ;: CHECK FOR XON
        BNE  1$       ;: NO XON GO WAIT FOR IT
20$:    CMPB 0$.MSP,0' ] ;END OF MESSAGE ?
        BEQ  WRT,.E    ;: IF EQ, YES
        CMPB 0$.MSP,0' [ ;: <CR>,<LF>?
        BNE  30$      ;: IF NE, NO
        BIC  0100,00TPS ;: DISABLE INTERRUPTS
        MOV  015,00TPB ;: SEND <CR>
        TSTB 00TPS    ;: WAIT FOR DONE
        BPL  -4        ;:
        INC  $.MSP     ;: POINT TO NEXT CHARACTER
        MOV  012,00TPB ;: SEND <LF>
        BIS  0100,00TPS ;: RE-ENABLE INTERRUPT
        RTI                ;: RETURN TO MAIN PROGRAM
30$:    MOVE  0$.MSP,00TPB ;: SEND NEXT CHARACTER
        INC  $.MSP     ;: POINT TO NEXT
        RTI                ;: RETURN TO CALLER
```

1219	006060	042737	000100	177584	WRT.,E:	BIC	#100,0#TPS	;;;DISABLE OUTPUT INTERRUPT
1220	006066	005037	007152			CLR	OUTFLG	;;;RESET OUTPUT FLAG
1221	006072	005737	007170			TST	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	;;;NECESARY 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
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250

006122 105737 177564
006126 100375
006130 010037 177566
006134 000207

```

.SBTTL $.KBO
;+
;SUBROUTINE: $.KBO
;*****
;
;KEYBOARD ECHO ROUTINE FOR TEMINAL INPUT
;
;INPUT: (SP) = CALLER PC
;        RU = PATTERN TO ECHO
;
;OUTPUT: PRINT PATTERN TO TERMINAL
;
;AUTHOR: BERT HUBER CSS/DP MUNICH 28-JUN-76
;-
; .PSECT CSSMON
; .ENABL LSB
$.KBO: TSTB @TPS ;ECHO RUNNING ?
; BPL $.KBO ;IF PL, WAIT
; MOV R0,@TPB ;ECHO CHARACTER
; RETURN ;RETURN TO CALLER
; .DSABL LSB

```

1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274 006136 005037 007150
1275 006142
1276 006166
1277 006212
1278 006222 005737 007152
1279 006226 001375
1280 006230
1281 006254 010637 007150
1282 006260 000207
1283

```

.SBTTL $.PRO
;
; SUBROUTINE: $.PRO
; *****
;
; THIS ROUTINE IS CALLED TO PRINT THE DEFAULT KEYBOARD
; PROMPT STRING ON THE TERMINAL, AND TO SET THE INPUT FLAG.
;
; INPUT:      (SP)      CALLER PC
;
; OUTPUT:     PROMPT STING ON TERMINAL
;
; AUTHOR:     BERT HUBER CSS/DP MUNICH      28-JUN-76
;
; MZ001
;
; THE USE OF THIS MACRO'S ALLOWS TO USE ONE VERSION OF CDM
; ON LSI AND PDP.
;
;
; .IDENT /V1.0/
; .PSECT CSSMON
; ENABL  LSB
$.PRO: CLR  INFLAG      ;;; RESET INPUT FLAG
      PREAD STATUS    ;SAVE STATUS   ;MZ001
      PSET  40        ;CLEAR PS     ;MZ001
      WRITE PROMES    ;;; WRITE PROMPT STRING
1$:   TST  OUTFLG     ;WAIT FOR OUTPUT COMPLETE
      BNE  1$
      PSET STATUS     ;RELOAD STATUS ;MZ001
      MOV  SP,INFLAG  ;;; SET INPUT FLAG
      RETURN          ;RETURN TO CALLING PROGRAM
      .DSABL  LSB

```

1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315

006262 000240
006264
006310
006334
006344 005737 007152
006350 001375
006376 000207

```

.SBTTL $..STX
;+
;SUBROUTINE: $..STX
;*****
;
;THIS ROUTINE PRINTS THE 'SYNTAX ERROR' MESSAGE
;
;INPUTS: (SP)= CALLER PC
;
;OUTPUT: MESSAGE ON TERMINAL
;
;AUTHOR: BERT HUBER CSS/DP MUNICH 28-JUN-76
;
;
; MZ001
; *****
; THE USE OF THIS MACRO'S ALLOWS TO RUN ONE VERSION OF CDM
; ON LSI AND PDP.
;
;
;.PSECT CSSMON
ENABL LSB
$..STX: NOP ;;;DUMMY
PREAD STATUS ;SAVE STATUS ;MZ001
PSET 40 ;CLEAR PS ;MZ001
WRITE MSYERR ;WRITE SYNTAX ERROR MESSAGE
1$: TST OUTFLG ;WAIT FOR OUTPUT COMPLETE
BNE 1$ ;
PSET STATUS ;RELOAD PS ;MZ001
RETURN ;;;RETURN TO CALLER
.DSABL LSB

```

1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329

```
.SBTTL $.DMP
;+
;SUBROUTINE: $.DMP
;*****
;
;THIS ROUTINE PRINTS ON THE TERMINAL PRINTER NUMBERS
;IN BINARY, DECIMAL AND OCTAL FORMAT, DEPENDING ON THE
;TYPE-ARGUMENT OF THE CALLING DUMP MACRO.
; THIS MONITOR WAS MODIFIED IN JANUARY 1983 TO INCLUDE XON AND XOFF
; TERMINAL FEATURES. ALSO INCLUDED SOME MACRO DIRECTIVES TO ALLOW
; ASSEMBLY ON VAX.
;
```

```

1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344 006400
1345 006402 016600 000004
1346 006406 005737 007152
1347 006412 001375
1348 006414 000170 006420
1349 006420 006426
1350 006422 006520
1351 006424 006630
1352 006426 012746 000006
1353 006432 016646 000010
1354 006436 012700 000060
1355 006442 006316
1356 006444 103001
1357 006446 005200
1358 006450 004737 006122
1359 006454 005366 000002
1360 006460 001415
1361 006462 012700 000060
1362 006466 006316
1363 006470 103002
1364 006472 062700 000004
1365 006476 006316
1366 006500 103001
1367 006502 122020
1368 006504 006316
1369 006506 103360
1370 006510 005200
1371 006512 000756
1372 006514 022626
1373 006516 000466
1374
1375
1376 006520 012746 000005
1377 006524 016646 000010
1378 006530 012700 000060
1379 006534 004737 006122
1380 006540 012700 000060
1381 006544 027716 000044
1382 006550 101004
1383 006552 005200
1384 006554 167716 000034
1385 006560 000771
1386 006562 004737 006122
1387 006566 062737 000002 006614

```

```

;
; INPUT:      4(SP)  = VALUE TO DUMP
;             2(SP)  =TYPE OF DUMP ; 0 = OCTAL
;                                     2 = DECIMAL
;                                     4 = BINARY
;
;             (SP)  =RETURN PC
;
; OUTPUT:     DUMP ON TERMINAL
;
; AUTHOR:     BERT HUBER CSS/DP MUNICH      28-JUN-76
;
; .PSECT CSSMON
; .ENABL LSB
$.DMP: PUSH RO ;SAVE RO
MOV 4(SP),RO ;GET TYPE ARGUMENT
TST OUTFLG ;OUTPUT RUNNING ?
BNE -4 ;IF NE, WAIT
JMP @D..TYP(RO) ;SELECT FUNCTION ROUTINE
D..TYP: $.DOC
$.DDC
$.DBN
$.DOC: MOV @6,-(SP) ;SETUP DIGIT COUNT
MOV 10(SP),-(SP) ;GET VALUE TO DUMP
MOV @'0,RO ;SETUP BASIC ASCII VALUE
ASL (SP) ;BUILD FIRST DIGIT
BCC 1$ ;IF CC, DIGIT IS A '0'
INC RO ;OTHERWISE DIGIT IS A ?
1$: CALL $.KBO ;DUMP DIGIT
DEC 2(SP) ;COUNT DIGITS DUMPED
BEQ 4$ ;IF EQ, ALL DONE
MOV @'0,RO ;SETUP NEXT DIGIT BASE
ASL (SP) ;BUILD NEXT DIGIT
BCC 2$ ;IF CC, DIGIT INCLUDES ?????
ADD @4,RO ;DIGIT = BASE+4
2$: ASL (SP) ;ANALYZE NEXT BIT
BCC 3$ ;IF CC, DIGIT INCLUDES ????
CMPB (RO)+,(RO)+ ;DIGIT = DIGIT + 2
3$: ASL (SP) ;ANALYZE LAST BIT OF DIGIT
BCC 1$ ;IF CC, DIGIT MUST BE EVEN
INC RO ;OTHERWISE DIGIT IS ODD
BR 1$ ;GO AND DUMP
4$: CMP (SP)+,(SP)+ ;READJUST STACK
BR DMP..E ;END OF DUMP
.DSABL LSB
.ENABL LSB
$.DDC: MOV @5,-(SP) ;SETUP A DIGIT COUNT
MOV 10(SP),-(SP) ;GET VALUE TO DUMP
MOV @'0,RO ;BUILD LEADING ZERO
CALL $.KBO ;AND PRINT
1$: MOV @'0,RO ;SETUP BASIC ASCII VALUE
2$: CMP @POINT,(SP) ;CMP VALUE WITH ACT. DEC. DIGIT
BHI 3$ ;VALUE IS SMALLER
INC RO ;COUNT NUMBER OF DIVISIONS
SUB @POINT,(SP) ;DIVIDE BY ACT. DEC. DIGIT
BR 2$ ;DO NEXT
3$: CALL $.KBO ;PRINT THIS DEC. DIGIT
ADD @2,$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..E    ;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,RO   ;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$: MOV      @61,RO ;SETUP A " 1 " CHARACTER
1411 006672 000767          BR       2$        ;DUMP IT
1412          .DSABL  LSB
1413 006674          DMP..E: POP      RO      ;RESTORE RO
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

```

.SBTTL $..ADP
;
; THIS SUBROUTINE IS USED TO PRINT OUT ADDITIONAL INFORMATION
; ABOUT AN ERROR. MOSTTIMES 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
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461

006754 062706 000004
006760
006770
007000 005737 007152
007004 001375
007006 012737 000006 000004
007014 000136

```

.SBTTL $..TOT
;
; THIS ROUTINE IS ENTERED AFTER THE MACRO "ADRTST" HAS BEEN EVOCED
; AND A BUS TIMEOUT HAS HAPPENED
;
; INPUT: 4(SP) = PC FOR PROGRAM CONTINUATION
;        RO   = REGISTER ADDRESS (E.G. 164000) WHICH SHOULD BE TESTED
;
; OUTPUT:      THE ERROR MESSAGE:
;              THIS ADDRESS IS NOT AVAILABLE ON UNIBUS; 164000
; AUTHOR:      ALBERT BREM,CSS MUC    FEB 1979
;
;
;
$..TOT: ADD    #4,SP      ;CORRECT SP
        WRITE #MNOAV    ;SAY ADDRESS NOT AVAILABLE ON UNIBUS
        DUMP  OCT,RO    ;DUMP REGISTER ADDRESS
        TST  OUTFLG    ;SYNCHRONIZE PRINTOUT
        BNE  .-4        ;
        MOV  #6,#4     ;REBUILD TRAP CATCHER
        JMP  @6(SP)+   ;JUMP TO CONTINUATION LABEL
;
;
;

```

```

1463          .SBTTL $.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          KDSAR0=172360
1474          KDSDR0=172320
1475          KISAR0=172340
1476          KISAR5=172352
1477          KISAR6=172354
1478          KISAR7=172356
1479          KISDR0=172300
1480          KISDR6=172314
1481          KISDR7=172316
1482          SISDR0=172200
1483          UDSAR0=177660
1484          UDSDR0=177620
1485          UISAR0=177640
1486          UISAR4=177650
1487          UISAR5=177652
1488          UISAR6=177654
1489          UISAR7=177656
1490          UISDR0=177600
1491          UISDR4=177610
1492          UISDR5=177612
1493          UISDR6=177614
1494          UISDR7=177616
1495          UDMPR=170200
1496          CMODE=140000
1497          PMODE=30000
1498          SRO=177572
1499          SR3=172516
1500          ;PTYP: .WORD 0          ; IF 0:=-LSI; IF -1:=-PDP
1501          ;MTYP: .WORD 0          ; IF 0:NO MM; IF 1:WITH MM
1502          MEMEND: .WORD 0,0      ; TOP OF CORE AND CORE EXTENSION
1503          ; STARTING FROM HERE TILL PROMES
1504          ; THE CONTENTS WILL BE CLEARED
1505          ; WITH "ABO"
1506          KBBUFF: .BLKB 80.      ; TERMINAL INPUT BUFFER
1507          KBBENL:          ; REFERENCE LABEL
1508          KBBPNT: .WORD KBBUFF  ; INPUT BUFFER POINTER
1509          INFLAG: .WORD 0        ; INPUT 'RUNNING' FLAG
1510          OUTFLG: .WORD 0        ; OUTPUT 'RUNNING' FLAG
1511          ERRFLG: .WORD 0        ; 'ERROR HANDLING REQ.' FLAG
1512          INPREQ: .WORD 0        ; 'WAIT FOR SPECIAL INPUT' FLAG
1513          PSDSWR: .WORD 0        ; SOFTWARE SWITCHREGISTER
1514          RUBFLG: .WORD 0        ; RUBOUTMODE-INDICATOR FLAG
1515          STATUS: .WORD 0        ; BUFFER PS-CONTENT IF SAVED
1516          $.ER: .WORD 0         ; HOLDS CURRENT ERROR NUMBER
1517          SPEFLG: .WORD 0        ; USED IN $.KBI AND $.WRT
1518          SCOPAD: 1000          ; SAVE AREA FOR SCOPE LOOP ADDRESS
1519          ;$.ASF: .WORD 0        ; SPECIAL ASCII INPUT FLAG

```

```

1520 007176 000000          $SHLD: .WORD 0          ;USED IN $.ADP
1521 007200 000000          $WAS:  .WORD 0          ;USED FOR $.ADP
1522                          .NLIST BEX
1523 007202      133      103      104 PROMES: .ASCII /[[CDM>]]/
1524 007210      133      040      135 DUMMES: .ASCII /[[ ]] /
1525 007213      133      113      105 MKBOVF: .ASCII /[[KEYBOARD BUFFER OVERFLOW! REENTER]]/
1526 007256      133      111      114 MILOCT: .ASCII /[[ILLEGAL OCTAL INPUT !! REENTER]]/
1527 007316      133      111      114 MILDEC: .ASCII /[[ILLEGAL DECIMAL INPUT !! REENTER]]/
1528 007360      133      111      114 MILBIN: .ASCII /[[ILLEGAL BINARY INPUT !! REENTER]]/
1529 007421      133      111      114 MILEMT: .ASCII /[[ILLEGAL EMT-NUMBER]]/
1530 007445      133      102      131 MILADR: .ASCII /[[BYTE ADDRESS IS NOT ALLOWED ]]/
1531 007503      133      123      131 MSYERR: .ASCII /[[SYNTAX ERROR !]]/
1532 007524      133      123      103 MILSCP: .ASCII /[[SCOPE LOOP SKIPPED TO A SUBSEQUENT ERROR !!!]]/
1533 007601      133      123      125      .ASCII /[[SUSPECT AN INTERMITTEND HARDWARE ERROR]]/
1534 007652      133      105      122 MERROR: .ASCII /[[ERROR ]]/
1535 007662      040      040      040 MERRP2: .ASCII / AT LOCATION : ]/
1536 007704      133      120      122 MPWRFL: .ASCII /[[PROGRAM MADE POWER FAIL []]]/
1537 007737      133      105      122 MERCOV: .ASCII /[[ERROR COUNTER OVERFLOW ON ERROR: ]]/
1538 010002      133      056      056 MERHLT: .ASCII /[[...HALT AFTER ERROR...]]/
1539 010032      133      117      120 MNOSUB: .ASCII /[[OPTION NOT SUPPORTED IN THIS CDM VERSION]]/
1540 010104      133      111      114 MILTR1: .ASCII /[[ILLEGAL INTERRUPT OR TRAP TO : ]]/
1541 010144      040      040      040 MILTR2: .ASCII / FROM: ]/
1542 010156      133      133      133 MVERSN: .ASCII /[[[[[ CDM V3.5 ---CSS DIAGNOSTIC MONITOR---[[[[[
1543 010252      110      111      107      .ASCII /[[HIGHEST MEMORY ADDRESS : ]]/
1544 010304      133      040      103 MER1: .ASCII /[[CONTENT SHOULD BE : ]]/
1545 010333      040      040      040 MER2: .ASCII / BUT WAS : ]/
1546 010351      133      133      124 $MNOAV: .ASCII /[[THIS ADDRESS IS NOT AVAILABLE ON UNIBUS: ]]/
1547 010426      133      115      105 MEMDEF: .ASCII /[[MEMORY MANAGEMENT UNIT IS DEFECT!!!]]/
1548 010472      133      111      114 MAPER: .ASCII /[[ILLEGAL MAPPING ADDRESS SPECIFIED!!!!]]/
1549 010542      133      127      105      .ASCII /[[WE WILL TRY WITHOUT MEMORY MANAGEMENT]]/
1550 010612      066      135      NO6: .ASCII /6]/
1551                          .EVEN
1552                          .LIST BEX
1553                          .DSABL LSB
1554 010614          CDM..E:

```

```

1          .TITLE  IEC11-A DIAGNOSTIC
2          .NLIST  CND
3          .LIST  MEB
4
5          WRITE  M1          ;IDENTIFY TEST
010614     012746 021606     MOV    #M1,-(SP)          ;SAVE #M1 ONTO STACK
010620     004737 005654     CALL   $.WRT              ;INITIATE PRINTOUT
6
7          RGAIN: WRITE  MES1          ;ENTER FIRST REG OF IEC11-A
010624     012746 021510     MOV    #MES1,-(SP)        ;SAVE #MES1 ONTO STACK
010630     004737 005654     CALL   $.WRT              ;INITIATE PRINTOUT
8          READ  OCT,RO          ;INPUT REQUEST
010634     005046          CLR    -(SP)              ;INDICATE OCTAL INPUT REQUIRED
010636     004737 005020     CALL   $.RED              ;ACCEPT INPUT
9          IF RO NE #0 GOTO 1$
010642     005700          TST   RO                  ;IS RO ZERO ?
010644     001402          BEQ   .+6                  ;
010646     000137 010656     JMP   1$                  ;
10         MOV   #160010,RO        ;SAVE DEFAULT VALUE
11         BIT   #7,RO             ;CHECK THREE LSB OF INPUT
12         BNE  RGAIN              ;IF NOT CLEAR,REQUEST AGAIN
13         IF RO LO #160010 GOTO RGAIN ;
010664     020027 160010     CMP   RO,#160010         ;COMPARE RO AND #160010
010670     103001          BHIS  .+4                  ;
010672     000754          BR   RGAIN                ;
14         IF RO HI #177776 GOTO RGAIN ;
010674     020027 177776     CMP   RO,#177776         ;COMPARE RO AND #177776
010700     101401          BLOS  .+4                  ;
010702     000750          BR   RGAIN                ;
15         ADRTST RGAIN
010704     012737 006754 000004 MOV   #$.TOT,#04         ;PROVIDE TIMEOUT ENTRY
010712     012746 010624     MOV   #RGAIN,-(SP)       ;SAVE RETURN ADDRESS ON STACK
010716     005710          TST   (RO)                ;IS ADDRESS AVAILABLE ?
010720     012737 000006 000004 MOV   #6,#04             ;RESTORE TIMEOUT VECTOR
010726     005726          TST   (SP)+                ;CORRECT STACK
16         MOV   RO,CIR            ;SAVE CIR ADDRESS
17         ADD  #2,RO              ;BUILD NEXT ADDRESS
18         ADRTST RGAIN
010740     012737 006754 000004 MOV   #$.TOT,#04         ;PROVIDE TIMEOUT ENTRY
010746     012746 010624     MOV   #RGAIN,-(SP)       ;SAVE RETURN ADDRESS ON STACK
010752     005710          TST   (RO)                ;IS ADDRESS AVAILABLE ?
010754     012737 000006 000004 MOV   #6,#04             ;RESTORE TIMEOUT VECTOR
010762     005726          TST   (SP)+                ;CORRECT STACK
19         MOV   RO,SMR            ;SAVE SMR REG ADDRESS
20         ADD  #2,RO              ;BUILD NEXT ADDRESS
21         ADRTST RGAIN
010774     012737 006754 000004 MOV   #$.TOT,#04         ;PROVIDE TIMEOUT ENTRY
011002     012746 010624     MOV   #RGAIN,-(SP)       ;SAVE RETURN ADDRESS ON STACK
011006     005710          TST   (RO)                ;IS ADDRESS AVAILABLE ?
011010     012737 000006 000004 MOV   #6,#04             ;RESTORE TIMEOUT VECTOR
011016     005726          TST   (SP)+                ;CORRECT STACK
22         MOV   RO,IOR            ;SAVE IOR REG ADDRESS
23         ADD  #1,RO              ;BUILD HIGH-BYTE ADDRESS
24         MOV   RO,IORHB          ;OF IOR
25         ADD  #1,RO              ;BUILD NEXT ADDRESS
26         ADRTST RGAIN
011040     012737 006754 000004 MOV   #$.TOT,#04         ;PROVIDE TIMEOUT ENTRY

```

```

011046 012746 010624      MOV      #RGAIN,-(SP)      ;SAVE RETURN ADDRESS ON STACK
011052 005710              TST      (RO)              ;IS ADDRESS AVAILABLE ?
011054 012737 000006 000004  MOV      #6,#0#4          ;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              2$: IF RO LO #40 GOTO VECAIN ;
011122 020027 000040      CMP     RO,#40            ;COMPARE RO AND #40
011126 103001              BHS    .+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,VECA        ;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              3$: IF RO LO #1 GOTO ADDRIN ;
011222 020027 000001      CMP     RO,#1            ;COMPARE RO AND #1
011226 103001              BHS    .+4                 ;
011230 000757              BR     ADDRIN             ;
44 011232              IF RO HI #36. GOTO ADDRIN ;
011232 020027 000044      CMP     RO,#36.          ;COMPARE RO AND #36.
011236 101401              BLOS   .+4                 ;
011240 000753              BR     ADDRIN             ;
45 011242 010037 024226      MOV     RO,IECAAD        ;SAVE IEC11-A BUS ADDRESS
46 011246 012737 000100 024216 2$: 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,TESTQ         ;TEST LON SWITCH FEATURE; **VQ2E00**
51 011276 017700 012712      8$:  MOV     #VSR,RO      ;CHECK VECTOR ADDRESS
    
```

\$.BOF

```

52 011302 020027 000774      CMP      RO,#774      ;LEGAL VECTOR ?
53 011306 101003              BHI      30$;        ;IF HI NO      ;**V01E02**
54 011310 020027 000040      CMP      RO,#40      ;LEGAL VECTOR  ;**V01E02**
55 011314 103005              BHIS     4$          ;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    $.WRT       ;INITIATE PRINTOUT
57 011326 000763              BR       8$;        ;CHECK AGAIN
58 011330 020027 000040      4$:  CMP      RO,#40; ;CHECK RANGE
59 011334 103770              BLO     30$;        ;IF LO, INVALID
60 011336 010037 024222      MOV      RO,VECTOR  ;GET VECTOR ADDRESS

```

62					.SBT1L	TEST SELECT ROUTINE	
63	011342				\$. .SPC:	REMVEC	SAVE #VECA ONTO STACK
	011342	012746	024224			MOV	RESET #VECA VECTOR
	011346	104002				EM1	
64	011350	005737	007152			TST	WAIT FOR ANY OUTPUT COMPLETE
65	011354	001375				BNE	
66	011356	000005				RESET	CLEAR THE WORLD
67	011360					SET6	RE-ENABLE TTY
	011360	052737	000100	177560		BIS	
68	011366					WRITE	ASK FOR PRINTING START MESSAGES
	011366	012746	024070			MOV	SAVE #TPMSG ONTO STACK
	011372	004737	005654			CALL	INITIATE PRINTOUT
69	011376					READ	Y OR N?
	011376	012746	000010			MOV	INDICATE ASCII INPUT IN RO REQUIRED
	011402	004737	005020			CALL	ACCEPT INPUT
70	011406					IF RO NE	SET FLAG
	011406	020027	000131			CMP	COMPARE RO AND #Y
	011412	001402				BEQ	
	011414	000137	011426			JMP	
71	011420	005037	011670			CLR	PRINT OUT START/END MESSAGES
72	011424	000402				BR	
73	011426	005237	011670		SFLAG:	INC	NO PRINT OUT
74	011432				LAB11:	WRITE	REQUEST NEXT TEST TO RUN
	011432	012746	024044			MOV	SAVE #MESNEX ONTO STACK
	011436	004737	005654			CALL	INITIATE PRINTOUT
75	011442					READ	WAIT FOR INPUT
	011442	012746	000002			MOV	INDICATE DECIMAL INPUT REQUIRED
	011446	004737	005020			CALL	ACCEPT INPUT
76	011452					IF RO HI	
	011452	020027	000016			CMP	COMPARE RO AND #14.
	011456	101401				BLOS	
	011460	000764				BR	
77	011462					IF RO EQ	
	011462	005700				TST	IS RO ZERO?
	011464	001002				BNE	
	011466	000137	011536			JMP	
78	011472	012737	177777	011666		MOV	LOOP ON SINGLE TESTS
79	011500	006300				ASL	BUILD WORD OFFSET INTO TABLE
80	011502	010027				MOV	AND SAVE
81	011504	000000			OFFSET:	.WORD	HOLDS CURRENT TEST NUMBER
82	011506	013737	011504	011664	SINGLE:	MOV	
83	011514	062737	011626	011664		ADD	
84	011522	017737	000136	011664		MOV	
85	011530	004777	000130			JSR	GOTO SELECTED TEST
86	011534	000764				BR	REPEAT TEST
87	011536	012737	000000	011666	TESTAL:	MOV	NO LOOP ON SINGLE TESTS
88	011544	012727	011630			MOV	GET ADDRESS OF FIRST TEST
89	011550	000000			SEQPNT:	.WORD	POINTER FOR TEST IN PROGRESS
90	011552	013737	011550	011664	SEQ:	MOV	LOAD ADDRESS OF NEXT TEST
91	011560	017737	000100	011664		MOV	
92	011566	004777	000072			JSR	GOTO EXECUTE TEST
93	011572	023727	011550	011662		CMP	ALL DONE?
94	011600	001404				BEQ	IF EQ, YES
95	011602	062737	000002	011550		ADD	POINT TO NEXT TEST
96	011610	000760				BR	AND DO NEXT
97	011612				TESTL1:	WRITE	END OF TEST
	011612	012746	024166			MOV	SAVE #MESEND ONTO STACK

J5

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
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163

```

.SBTTL LOCAL MACRO DEFINITIONS
;+
; THIS SECTION CONTAINS ONLY MACRO DEFINITIONS
; USED IN THE IEC-11A DIAGNOSTIC.
;-
.MACRO MCLEAR
BIS #BIT5,@CIR ;EXECUTE MASTER CLEAR
.ENDM MCLEAR

.MACRO SACS
BIS #BIT0,@CIR ;SET SACS BIT IN CIR
.ENDM SACS

.MACRO INHSRQ
BIS #BIT10,@CIR ;SET INHIBIT SRQ INTERRUPT ; **VO2E00**
.ENDM INHSRQ ; **VO2E00**

.MACRO INTENB
BIS #BIT6,@CIR ;ENABLE INTERRUPT IN IEC-11A
.ENDM INTENB

.MACRO TSTEND A,?L
TST MSGFLG ;PRINTOUT OR NOT?
BNE L ;TEST 'A' END MESSAGE
WRITE MTE'A; ;RETURN TO CALLER
RETURN;
.ENDM TSTEND

.MACRO SIC
BIS #BIT16,@SMR ;SEND INTERFACE CLEAR
.ENDM SIC

.MACRO SAVCIR
MOV @CIR,R0 ;SAVE CIR CONTENT
.ENDM SAVCIR

.MACRO SAVSMR
MOV @SMR,R1 ;SAVE SMR CONTENT
.ENDM SAVSMR

.MACRO SAVIOR
MOV @IOR,R2 ;SAVE IOR CONTENT
.ENDM SAVIOR

.MACRO GTS
BIS #BIT2,@SMR ;GO TO STAND BY
.ENDM GTS

.MACRO LTN
SET4 @SMR
.ENDM LTN

.MACRO LUN
SET5 @SMR
.ENDM LUN

```

```

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 C: .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,RO
194 011702 006316 ASL @SP
195 011704 005500 ADC RO
196 011706 006200 ASR RO
197 011710 005516 ADC @SP
198 011712 061600 ADD @SP,RO
199 011714 005616 SBC @SP
200 011716 060016 ADD RO,@SP
201 011720 005600 SBC RO
202 011722 012637 011734 MOV (SP),RA
203 011726 010037 011736 MOV RO,RB
204 011732 000207 RETURN
205 011734 135753 RA: .WORD 135753
206 011736 024642 RB: .WORD 024642
207 ;

```

TEST LON SWITCH FEATURE				TESTO:	SBTTL	TEST LON SWITCH FEATURE	
209					SCOPE		
210	011740	104000			WRITE		
211	011742				MOV	SLON	; SET LON SW MESSAGE ; **V02E00**
	011742	012746	022112		CALL	SLON, (SP)	; SAVE SLON ONTO STACK ; **V02E00**
	011746	004737	005654		CALL	\$.WRT	; INITIATE PRINTOUT
212	011752				READ	OCT,RO	
	011752	005046			CLR	-(SP)	; INDICATE OCTAL INPUT REQUIRED
	011754	004737	005020		CALL	\$.RED	; ACCEPT INPUT
213	011760				MCLEAR		; PRODUCE CLEAN STATUS ; **V02E00**
	011760	052777	000040	012216	BIS	BIT5,BCIR	; EXECUTE MASTER CLEAR
214	011766	027727	012212	000602	CMP	BCIR,#602	; STATE CHGE,INT,LON ; **V02E00**
215	011774	001404			BEQ	1\$; BRANCH IF CORRECT ; **V02E00**
216	011776				ERROR	114.	; CIR CONT NOT #602 ; **V02E00**
	011776	104562			TRAP	114.	; ERROR 114.
	012000	000000			.WORD	0	; ERROR COUNT
217	012002	000137	011740		JMP	TESTO	; **V02E00**
218	012006	027727	012174	010000	1\$:	CMP	SMR,#BIT12 ; LACS SET EXCLUSIVELY? ; **V02E00**
219	012014	001404			BEQ	2\$; BRANCH IF O.K. ; **V02E00**
220	012016				ERROR	115.	; LACS NOT OR NOT ONLY ; **V02E00**
	012016	104563			TRAP	115.	; ERROR 115.
	012020	000000			.WORD	0	; ERROR COUNT
221	012022	000137	011740		JMP	TESTO	; **V02E00**
222	012026				2\$:	WRITE	CLON ; RESET LON SW MSG ; **V02E00**
	012026	012746	022216		MOV	CLON, -(SP)	; SAVE CLON ONTO STACK ; **V02E00**
	012032	004737	005654		CALL	\$.WRT	; INITIATE PRINTOUT
223	012036				READ	OCT,RO	
	012036	005046			CLR	-(SP)	; INDICATE OCTAL INPUT REQUIRED
	012040	004737	005020		CALL	\$.RED	; ACCEPT INPUT
224	012044	022777	000600	012132	CMP	600,BCIR	; STATE CHGE,INT ; **V02E00**
225	012052	001404			BEQ	3\$; BRANCH IF O.K. ; **V02E00**
226	012054				ERROR	116.	; CIR CONTENT NOT #600 ; **V02E00**
	012054	104564			TRAP	116.	; ERROR 116.
	012056	000000			.WORD	0	; ERROR COUNT
227	012060	000137	011740		JMP	TESTO	; **V02E00**
228	012064	027727	012116	010000	3\$:	CMP	SMR,#BIT12 ; LACS STILL SET? ; **V02E00**
229	012072	001404			BEQ	4\$; BRANCH IF YES ; **V02E00**
230	012074				ERROR	117.	; ERROR IF NO ; **V02E00**
	012074	104565			TRAP	117.	; ERROR 117.
	012076	000000			.WORD	0	; ERROR COUNT
231	012100	000137	011740		JMP	TESTO	; **V02E00**
232	012104	000207			4\$:	RETURN	; RETURN FROM SUBTEST ; **V02E00**

234					.SBTTL	TEST 1 EXECUTE MASTER CLEAR	
235	012106	005737	011670	TEST1:	TST	MSGFLG	;PRINTOUT OR NOT?
236	012112	001004			BNE	LOPT1	
237	012114				WRITE	MT1S	;TEST1 START MESSAGE
	012114	012746	022324		MOV	#MT1S,-(SP)	;SAVE #MT1S ONTO STACK
	012120	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
238	012124	104000		LOPT1:	SCOPE		;SCOPE LOOP MARKER
239	012126				MCLEAR		
	012126	052777	000040	012050	BIS	#BITS,@CIR	;EXECUTE MASTER CLEAR
240	012134	017700	012046		MOV	@SMR,RO	;SAVE SMR CONTENT
241	012140	001404			BEQ	1\$;IF EQ, YES
242	012142				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			1\$:	SAVCIR		
	012152	017700	012026		MOV	@CIR,RO	;SAVE CIR CONTENT
245	012156	001404			BEQ	2\$;IF EQ, NO BIT SET
246	012160				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				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			3\$:	LOOP	-1,LOPT1	
	012206	005327			DEC	(PC)+	;COUNT LOOPS
	012210	177777		64\$:	.WORD	-1	
	012212	005737	012210		TST	64\$;IS 64\$ ZERO ?
	012216	001401			BEQ	.+4	
	012220	000741			BR	LOPT1	
	012222	012737	177777	012210	MOV	#-1,64\$;SETUP NEW LOOP COUNT
253	012230				TSTEND	1	
	012230	005737	011670		TST	MSGFLG	;PRINTOUT OR NOT?
	012234	001004			BNE	66\$	
	012236	012746	022371		MOV	#MTE1,-(SP)	;SAVE #MTE1 ONTO STACK
	012242	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
	012246	000207		66\$:	RETURN;		;RETURN TO CALLER

```

255 .SBTTL TEST 2 TEST SACS BIT IN CIR
256 012250 005737 011670 TEST2: TST MSGFLG ;PRINTOUT OR NOT?
257 012254 001004 BNE LOPT2
258 012256 012746 022405 WRITE MT2S ;TEST2 START MESSAGE
012262 004737 005654 MOV #MT2S,-(SP) ;SAVE #MT2S ONTO STACK
259 012266 052777 000040 011710 LOPT2: CALL $,.WRT ;INITIATE PRINTOUT
012266 052777 000040 011710 MCLER ;PRODUCE CLEAR CONTROLLER STATE
260 012274 104000 BIS #BIT5,@CIR ;EXECUTE MASTER CLEAR
261 012276 052777 000001 011700 SACS ;SCOPE LOOP MARKER
012304 SAVCIR ;SET SACS BIT IN CIR
262 012304 017700 011674 MOV @CIR,RO ;SAVE CIR CONTENT
263 012310 TSTO RO ;
012310 032700 000001 BIT #BIT0,RO
264 012314 001004 BNE 1# ;IF NE, SET
265 012316 104404 ERROR 4. ;SACS BIT NOT SET
012320 000000 TRAP 4. ;ERROR 4.
012322 000137 012250 .WORD 0 ;ERROR COUNT
266 012322 000137 012250 JMP TEST2 ;
267 012326 104000 1#: SCOPE ;SCOPE LOOP MARKER
268 012330 042777 000001 011646 CLRO @CIR ;TRY TO RESET SACS BIT
012330 042777 000001 011646 BIC #BIT0,@CIR
269 012336 017700 011642 SAVCIR
012336 017700 011642 MOV @CIR,RO ;SAVE CIR CONTENT
270 012342 032700 000001 TSTO RO ;SACS RESET ?
012342 032700 000001 BIT #BIT0,RO
271 012346 001404 BEQ 2# ;IF EQ, YES
272 012350 104405 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 2#: SCOPE ;SCOPE LOOP MARKER
275 012362 052777 000001 011614 SACS ;SET SACS BIT IN CIR
012362 052777 000001 011614 BIS #BIT0,@CIR
276 012370 052777 000040 011606 MCLER ;EXECUTE MASTER CLEAR
012370 052777 000040 011606 BIS #BIT5,@CIR ;IS SACS CLEAR AFTER MC ?
277 012376 017700 011602 SAVCIR ;SAVE CIR CONTENT
012376 017700 011602 MOV @CIR,RO
278 012402 001404 BEQ 3# ;IF EQ, YES
279 012404 104406 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 005327 3#: LOOP ;COUNT LOOPS
012414 005327 DEC (PC)+
012416 177777 .WORD 1
012420 005737 012416 TST 64# ;IS 64# ZERO ?
012424 001401 BEQ .+4 ;
012426 000717 BR LOPT2 ;
012430 012737 177777 012416 MOV #1,64# ;SETUP NEW LOOP COUNT
282 012436 005737 011670 TSTEND 2 ;PRINTOUT OR NOT?
012436 005737 011670 TST MSGFLG
012442 001004 BNE 66# ;SAVE #MTE2 ONTO STACK
012444 012746 022452 MOV #MTE2,-(SP) ;INITIATE PRINTOUT
012450 004737 005654 CALL $,.WRT
012454 000207 66#: RETURN ;RETURN TO CALLER
  
```

```

284          SBTTL TEST 3 TEST CIR BIT 10 INHIBIT SRQ INTERRUPT
285 012456 005737 011670 TEST3: TST MSGFLG ;PRINTOUT OR NOT?
286 012462 001004          BNE LOPT2A
287 012464          WRITE MT2AS ;GIVE TEST3 START MSG ; **V02E00**
      012464 012746 022466 MOV #MT2AS,-(SP) ;SAVE #MT2AS ONTO STACK
      012470 004737 005654 CALL $..WRT ;INITIATE PRINTOUT
288 012474          MCLEAR LOPT2A: ;PRODUCE CLEAR CONTR STATE ; **V02E00**
      012474 052777 000040 011502 BIS #BIT5,#CIR ;EXECUTE MASTER CLEAR
289 012502 104000          SCOPE ;SCOPE LOOP ENTRY ; **V02E00**
290 012504          INHSRQ ;SET SRQ INT INHIBIT ; **V02E00**
      012504 052777 002000 011472 BIS #BIT10,#CIR ;SET INHIBIT SRQ INTERRUPT
291 012512          TST10 #CIR ;IS BIT 10 IN CIR SET? ; **V02E00**
      012512 032777 002000 011464 BIT #BIT10,#CIR
292 012520 001004          BNE 1$ ;BRANCH IF SET ; **V02E00**
293 012522          ERROR 105. ;BIT 10 IN CIR NOT SET ; **V02E00**
      012522 104551 TRAP 105. ;ERROR 105.
      012524 000000 .WORD 0 ;ERROR COUNT
294 012526 000137 012456          JMP TEST3 ; **V02E00**
295 012532          CLR10 #CIR ;RESET BIT 10 IN CIR ; **V02E00**
      012532 042777 002000 011444 BIC #BIT10,#CIR
296 012540          TST10 #CIR ;CLEAR AGAIN ? ; **V02E00**
      012540 032777 002000 011436 BIT #BIT10,#CIR
297 012546 001404          BEQ 2$ ;BRANCH IF YES ; **V02E00**
298 012550          ERROR 106. ;BIT 10 NOT CLEAR ; **V02E00**
      012550 104552 TRAP 106. ;ERROR 106.
      012552 000000 .WORD 0 ;ERROR COUNT
299 012554 000137 012456          JMP TEST3 ; **V02E00**
300 012560 104000          SCOPE 2$:
301 012562          INHSRQ ;SET SRQ INT INHIBIT ; **V02E00**
      012562 052777 002000 011414 BIS #BIT10,#CIR ;SET INHIBIT SRQ INTERRUPT
302 012570          TST10 #CIR ;LOOK IF SET AGAIN ; **V02E00**
      012570 032777 002000 011406 BIT #BIT10,#CIR
303 012576 001004          BNE 3$ ;BRANCH IF O.K. ; **V02E00**
304 012600          ERROR 107. ;BIT 10 IN CIR NOT SET AGAIN ; **V02E00**
      012600 104553 TRAP 107. ;ERROR 107.
      012602 000000 .WORD 0 ;ERROR COUNT
305 012604 000137 012456          JMP TEST3
306 012610          MCLEAR 3$:
      012610 052777 000040 011366 BIS #BIT5,#CIR ;RESET BIT10 IN CIR BY MC ; **V02E00**
307 012616          TST10 #CIR ;EXECUTE MASTER CLEAR
      012616 032777 002000 011360 BIT #BIT10,#CIR ;LOOK IF DONE ; **V02E00**
308 012624 001404          BEQ 4$ ;BRANCH IF O.K. ; **V02E00**
309 012626          ERROR 110. ;BIT10 IN CIR NOT CLEAR ; **V02E00**
      012626 104556 TRAP 110. ;ERROR 110.
      012630 000000 .WORD 0 ;ERROR COUNT
310 012632 000137 012456          JMP TEST3 ; **V02E00**
311 012636          LOOP -1,LOPT2A 4$:
      012636 005327 DEC (PC); ;COUNT LOOPS
      012640 177777 .WORD -1 ;
      012642 005737 012640 64$: TST 64$ ;IS 64$ ZERO ?
      012646 001401 BEQ .+4 ;
      012650 000711 BR LOPT2A ;
      012652 012737 177777 012640 MOV #-1,64$ ;SETUP NEW LOOP COUNT
312 012660          TSTEND 2A ;RETURN TO CALLER
      012660 005737 011670 TST MSGFLG ;PRINTOUT OR NOT?
      012664 001004 BNE 66$ ;
      012666 012746 022554 MOV #MTE2A,-(SP) ;SAVE #MTE2A ONTO STACK

```

D6

IEC113A DIAGNOSTIC MACRO M1200 30 MAR 84 16:02 PAGE 36-1
TEST 3 TEST CIR BIT 10 INHIBIT SRQ INTERRUPT

SEQ 68

012672 004737 005654
012676 000207

66\$: CALL \$.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	MT3S					;START MESSAGE TEST 4
	012706	012746	022570		MOV	#MT3S,-(SP)					;SAVE #MT3S 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				CLR#	#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	1#					;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				1#:	TST13	R1				;SIAS SET ?
	013004	032701	020000		BIT	#BIT13,R1					
333	013010	001004			BNE	2#					;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				2#:	TST8	R0				;STATE CHANGE ?
	013022	032700	000400		BIT	#BIT8,R0					
337	013026	001004			BNE	3#					;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				3#:	TST8	R1				;CACS SET ?
	013040	032701	000400		BIT	#BIT8,R1					
341	013044	001004			BNE	10#					;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		10#:	CLR	11#				
345	013062				4#:	TST13	#SMR				;SIAS RESET ?
	013062	032777	020000	011116	BIT	#BIT13,#SMR					
346	013070	001407			BEQ	5#					;IF EQ, YES
347	013072	005227			INC (PC)						;WAIT A WHILE
348	013074	000000			11#:	.WORD	0				
349	013076	001371			BNE	4#					

350	013100				ERROR 13.				;SIAS NOT RESET
	013100	104415			TRAP 13.				;ERROR 13.
	013102	000000			.WORD 0				;ERROR COUNT
351	013104	000137	012700		JMP TEST4				
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 RESET
	013126	104416			TRAP 14.				;ERROR 14.
	013130	000000			.WORD 0				;ERROR COUNT
357	013132	000137	012700		JMP TEST4				
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		JMP TEST4				
362	013154	104000		7\$:	SCOPE				;SCOPE LOOP MARKER
363	013156				MCLEAR				
	013156	052777	000040	011020	BIS @BIT5,@CIR				;EXECUTE MASTER CLEAR
364	013164				SAVSMR				;SAVE NEW SMR CONTENT
	013164	017701	011016		MOV @SMR,R1				;SAVE SMR CONTENT
365	013170				TST8 R1				;CACS RESET ?
	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		JMP TEST4				
369	013206	104000		8\$:	SCOPE				;SCOPE LOOP MARKER
370	013210				SIC				
	013210	052777	000100	010770	BIS @BIT6,@SMR				;SEND INTERFACE CLEAR
371	013216				SAVCIR				
	013216	017700	010762		MOV @CIR,R0				;SAVE CIR CONTENT
372	013222				TST12 R0				;IS 'ILL. MESS' SET ?
	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		JMP TEST4				
376	013240			9\$:	LOOP -1,LOPT3				
	013240	005327			DEC (PC)+				;COUNT LOOPS
	013242	177777		64\$:	.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

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

G6

SEQ 71

013274 004737 005654
013300 000207

66\$; 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 $..WRT ;INITIATE PRINTOUT
383 013320          LOPT3A: MCLCARE ;MASTER CLEAR
      013320 052777 000040 010656 BIC #BIT5,#CIR ;EXECUTE MASTER CLEAR
384 013326 013701 024204          MOV CIR,R1 ;GET CIR ADDRESS
385 013332 104000          SCOPE ;SCOPE LOOP MARKER
386 013334          SET6 #R1 ;TRY TO SET INTERRUPT ENABLE
      013334 052711 000100 BIC #BIT6,#R1
387 013340          TST6 #R1 ;IS INT ENBL SET ?
      013340 032711 000100 BIT #BIT6,#R1
388 013344 001004          BNE 1$ ;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 CHGE
403 013444 005737 011670 TEST6: TST MSGFLG ;PRINTOUT OR NOT?
404 013450 001004 BNE 10$
405 013452 WRITE MT4S ;TEST 6 START MESSAGE
    013452 012746 022734 MOV #MT4S,-(SP) ;SAVE #MT4S ONTO STACK
    013456 004737 005654 CALL $..WRT ;INITIATE PRINTOUT
406 013462 10$: SETVEC #VECTOR,#IECINT,#0 ;SET VECTOR ; **V03E02**
    013462 012746 024222 MOV #VECTOR,(SP) ;SAVE #VECTOR ONTO STACK
    013466 012746 021334 MOV #IECINT,-(SP) ;SAVE #IECINT ONTO STACK
    013472 012746 000000 MOV #0,-(SP) ;SAVE #0 ONTO STACK
    013476 104001 EMT 1 ;SETUP #VECTOR VECTOR
407 013500 104000 LOPT4: SCOPE ;SCOPE LOOP MARKER
408 013502 MCLER
    013502 052777 000040 010474 BIS #BIT5,#CIR ;EXECUTE MASTER CLEAR
409 013510 012705 010000 MOV #BIT12,R5 ;SETUP INTERRUPT MASK
410 013514 012704 013574 MOV #3$,R4 ;SETUP RETURN ADDRESS
411 013520 INTENB
    013520 052777 000100 010456 BIS #BIT6,#CIR ;ENABLE INTERRUPT IN IEC-11A
412 013526 105037 177776 CLRB #PS ;ENABLE INTERRUPTS
413 013532 SIC
    013532 052777 000100 010446 BIS #BIT6,#SMR ;SEND INTERFACE CLEAR
414 013540 005037 013546 CLR 2$
415 013544 005227 1$: INC (PC)+ ;WAIT FOR INTERRUPT
416 013546 000000 2$: .WORD 0
417 013550 001375 BNE 1$
418 013552 SAVIEC
    013552 017700 010426 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 ERROR 20. ;NO INTERRUPT TO SIC
    013566 104424 TRAP 20. ;ERROR 20.
    013570 000000 .WORD 0 ;ERROR COUNT
420 013572 000724 BR TEST6
421 013574 103003 3$: BCC 4$ ;IF CC, INTERRUPT OK
422 013576 ERROR 21. ;ILLEGAL INTERRUPT
    013576 104425 TRAP 21. ;ERROR 21.
    013600 000000 .WORD 0 ;ERROR COUNT
423 013602 000720 BR TEST6
424 013604 104000 4$: SCOPE ;SCOPE LOOP MARKER
425 013606 MCLER
    013606 052777 000040 010370 BIS #BIT5,#CIR ;EXECUTE MASTER CLEAR
426 013614 012705 000400 MOV #BIT8,R5 ;SETUP INTERRUPT MASK
427 013620 012704 013702 MOV #7$,R4 ;SETUP INTERRUPT RETURN ADDRESS
428 013624 INTENB
    013624 052777 000100 010352 BIS #BIT6,#CIR ;ENABLE INTERRUPT IN IEC-11A
429 013632 SACS
    013632 052777 000001 010344 BIS #BIT0,#CIR ;SET SACS BIT IN CIR
430 013640 SIC
    013640 052777 000100 010340 BIS #BIT6,#SMR ;SEND INTERFACE CLEAR
431 013646 005037 013654 CLR 6$ ;WAIT FOR INTERRUPT
432 013652 005227 5$: INC (PC)+
433 013654 000000 6$: .WORD 0
434 013656 001375 BNE 5$
435 013660 SAVIEC ;SAVE ALL IEC-REGISTERS
    013660 017700 010320 MOV #CIR,R0 ;SAVE CIR CONTENT
    013664 017701 010316 MOV #SMR,R1 ;SAVE SMR CONTENT
    013670 017702 010316 MOV #IOR,R2 ;SAVE IOR CONTENT
    
```

```

436 013674          ERROR 22.          ;NO INTERRUPT TO SIC AND SACS
    013674 104426   TRAP 22.          ;ERROR 22.
    013676 000000   .WORD 0           ;ERROR COUNT
437 013700 000661   BR TEST6
438 013702 103003   7$: BCC 8$       ;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          8$: LOOP -1,LOPT4 ;COUNT LOOPS
    013712 005327   DEC (PC)+
    013714 177777   64$: .WORD -1
    013716 005737 013714   TST 64$ ;IS 64$ ZERO ?
    013722 001401   BEQ .+4
    013724 000665   BR LOPT4
    013726 012737 177777 013714   MOV 0-1,64$ ;SETUP NEW LOOP COUNT
442 013734          TSTEND 4
    013734 005737 011670   IST MSGFLG ;PRINTOUT OR NOT?
    013740 001004   BNE 66$
    013742 012746 023013   MOV 0MTE4, -(SP) ;SAVE 0MTE4 ONTO STACK
    013746 004737 005654   CALL $, .WRT ;INITIATE PRINTOUT
    013752 000207   66$: RETURN; ;RETURN TO CALLER

```

444					.SBITL	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	MT55	;START MESSAGE TEST 7	
	013762	012746	023027		MOV	#MT55,-(SP)	;SAVE #MT55 ONTO STACK	
	013766	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT	
448	013772				LOPT5: MCL	LEAR	;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 ITME
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#:	TSTB	@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          CLR7    @SMR          ; RESET SRE
      014160 042777 000200 010020 3$:  BIC    @BIT7,@SMR
479 014166 000240          NOP
480 014170 000240          NOP
481 014172 000240          NOP
482 014174          SAVCIR
      014174 017700 010004      MOV    @CIR,R0          ; SAVE CIR CONTENT
483 014200          SAVSMR
      014200 017701 010002      MOV    @SMR,R1          ; SAVE CIR CONTENT
484 014204 105701          TSTB   R1              ; SAVE SMR CONTENT
485 014206 100003          BPL    6$              ; SAVE SMR CONTENT
486 014210          ERROR   30.          ; SRE RESET ?
      014210 104436          TRAP   30.          ; IF PL, YES
      014212 000000          .WORD  0              ; SRE NOT RESET
487 014214 000657          BR      TEST7          ; ERROR 30.
488 014216          CLR8    @CIR          ; ERROR COUNT
      014216 042777 000400 007760 6$:  BIC    @BIT8,@CIR
489 014224          TST14   R1              ; RESET STATE CHANGE
      014224 032701 040000      BIT    @BIT14,R1
490 014230 001403          BEQ    4$              ; SRAS RESEY ?
491 014232          ERROR   31.          ; IF EQ, YES
      014232 104437          TRAP   31.          ; SRAS NOT RESET
      014234 000000          .WORD  0              ; ERROR 31.
492 014236 000646          BR      TEST7          ; ERROR COUNT
493 014240 000300          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          MCLARE
      014260 052777 000040 007716 7$:  BIS    @BIT5,@CIR
500 014266 104000          SCOPE
501 014270          SRE
      014270 052777 000200 007710      BIS    @BIT7,@SMR
502 014276 000240          NOP
503 014300 000240          NOP
504 014302 000240          NOP
505 014304          SAVCIR
      014304 017700 007674      MOV    @CIR,R0          ; SAVE CIR CONTENT
506 014310          SAVSMR
      014310 017701 007672      MOV    @SMR,R1          ; SAVE CIR CONTENT
507 014314          TST12   R0              ; SAVE SMR CONTENT
      014314 032700 010000      BIT    @BIT12,R0
508 014320 001003          BNE    8$              ; ILLEG. MESSAGE ?
509 014322          ERROR   104.         ; IF NE, YES
      014322 104550          TRAP   104.         ; NO ILLEG. MESSAGE
      014324 000000          .WORD  0              ; ERROR 104.
510 014326 000612          BR      TEST7          ; ERROR COUNT
511 014330          LOOP    -1,LOP5
      014330 005327          DEC    (PC)+
      014332 177777          .WORD  -1
      014334 005737 014332          TST    64$

```

; IS 64\$ ZERU ?

	014340	001401			BEG	.+4		
	014342	000613			BR	LOPT5		
	014344	012737	177777	014332	MOV	0-1,64\$;SETUP NEW LOOP COUNT
512	014352				TSTEND	5		
	014352	005737	011670		IST	MSGFLG		;PRINTOUT OR NOT?
	014356	001004			BNE	66\$		
	014360	012746	023113		MOV	#MTE5, -(SP)		;SAVE #MTE5 ONTO STACK
	014364	004737	005654		CALL	\$..WRT		;INITIATE PRINTOUT
	014370	000207			RETURN;			;RETURN TO CALLER

66\$:

```

514
515 014372 005737 011670
516 014376 001004
517 014400
    014400 012746 023127
    014404 004737 005654
518 014410 104000
519 014412
    014412 052777 000040 007564
520 014420 013704 024212
521 014424 005003
522 014426 110314
523 014430 011402
524 014432 020302
525 014434 001403
526 014436
    014436 104457
    014440 000000
527 014442 000753
528 014444 005203
529 014446 020327 000400
530 014452 001365
531 014454
    014454 005327
    014456 001750
    014460 005737 014456
    014464 001401
    014466 000751
    014470 012737 001750 014456
532 014476
    014476 005737 011670
    014502 001004
    014504 012746 023165
    014510 004737 005654
    014514 000207

          .SBTTL TEST 8 - IOR DATA TEST
TEST8:  TST  MSGFLG ;PRINTOUT OR NOT?
        BNE  10$
        WRITE MT5AS ;START MESSAGE TEST 8
        MOV  #MT5AS,-(SP) ;SAVE #MT5AS ONTO STACK
        CALL $..WRT ;INITIATE PRINTOUT

10$:    SCOPE
LOPT5A: MCLER
        ;
        BIS  #BIT5,&CIR ;EXECUTE MASTER CLEAR
        MOV  IOR,R4 ;GET IOR ADDRESS
        CLR  R3 ;PREPARE DATA
1$:    MOVB  R3,&H4 ;WRITE PATTERN INTO IOR
        MOV  &R4,R2 ;READ BACK PATTERN FROM IOR
        CMP  R3,R2 ;PATTERN EQUAL ?
        BEQ  2$ ;IF EQ, YES
        ERROR 47. ;DATA CHECK IN IOR
        TRAP 47. ;ERROR 47.
        .WORD 0 ;ERROR COUNT
        BR  TEST8 ;RESTART TEST
2$:    INC  R3 ;BUILD NEXT PATTERN
        CMP  R3,#400 ;ALL DONE ?
        BNE  1$ ;IF NE, NO
        LOOP 1000.,LOPT5A ;COUNT LOOPS
        DEC  (PC)+
64$:   .WORD 1000.
        TST  64$
        BEQ  .+4 ;IS 64$ ZERO ?
        BR  LOPT5A
        MOV  #1000.,64$ ;SETUP NEW LOOP COUNT
        TSTEND 5A ;RETURN TO MAINPROGRAM
        TST  MSGFLG ;PRINTOUT OR NOT?
        BNE  66$
        MOV  #MTE5A,-(SP) ;SAVE #MTE5A ONTO STACK
        CALL $..WRT ;INITIATE PRINTOUT
66$:   RETURN; ;RETURN TO CALLER

```

B7

```

534
535 014516          TEST9: .SBTTL TEST 9 TEST STATE CHGE INTERRUPT WITH GTS AND SIC
    014516 012746 024222      SETVEC @VECTOR,@IECINT,@0          ;SET VECTOR ;**V03E02**
    014522 012746 021334      MOV @VECTOR,-(SP)          ;SAVE @VECTOR ONTO STACK
    014526 012746 000000      MOV @IECINT,-(SP)        ;SAVE @IECINT ONTO STACK
    014532 104001          MOV @0,-(SP)            ;SAVE @0 ONTO STACK
    536 014534 005737 011670    EMT 1                    ;SETUP @VECTOR VECTOR
    537 014540 001004          TST MSGFLG              ;PRINTOUT OR NOT?
    538 014542          BNE LOPT6
    014542 012746 023201      WRITE MT6S              ;START MESSAGE TEST 9
    014546 004737 005654      MOV @MT6S,-(SP)        ;SAVE @MT6S ONTO STACK
    539 014552          CALL $.WRT              ;INITIATE PRINTOUT
    014552 052777 000040 007424 LDPT6: MCLR              ;MASTER CLEAR
    540 014560          BIS @BIT5,@CIR          ;EXECUTE MASTER CLEAR
    014560 052777 000001 007416 SACS              ;SET CONTROLLER ACTIVE STATE
    541 014566          BIS @BIT0,@CIR          ;SET SACS BIT IN CIR
    014566 052777 000100 007412 SIC              ;SEND INTERFACE CLEAR
    542 014574 013700 024204    BIS @BIT6,@SMR          ;SEND INTERFACE CLEAR
    543 014600          MOV CIR,R0              ;GET CIR ADDRESS
    014600 032710 000400      1$: TST @R0              ;WAIT FOR STATE CHANGE
    014604 001775          BIT @BIT8,@R0
    544 014606          BEQ 1$
    014610 104000          SCOPE
    545 014610          CLR @R0
    014610 042710 000400      BIC @BIT8,@R0
    546 014614          TST @R0
    014614 032710 000400      BIT @BIT8,@R0
    547 014620 001775          BEQ -4
    548 014622          CLR @R0
    014622 042710 000400      BIC @BIT8,@R0
    549 014626          GTS
    014626 052777 000004 007352 BIS @BIT2,@SMR
    550 014634 005037 014652    CLR 3$
    551 014640          2$: TST @SMR
    014640 032777 001000 007340 BIT @BIT9,@SMR
    552 014646          BNE 4$
    014646 001012          INC (PC)+
    553 014650 005227          .WORD 0
    554 014652 000000          BNE 2$
    555 014654 001371          SAVCIR
    014654 001371          MOV @CIR,R0
    556 014656          SAVSMR
    014656 017700 007322      MOV @SMR,R1
    557 014662          ERROR 35.
    014662 017701 007320      TRAP 35.
    558 014666          .WORD 0
    014666 104443          BR TEST9
    559 014670 000000          3$: TST @R0
    014670 000000          BIT @BIT8,@R0
    560 014672 000711          BNE 5$
    014672 000711          SAVCIR
    561 014674          MOV @CIR,R0
    014674 032710 000400      SAVSMR
    562 014700 001007          MOV @SMR,R1
    014700 001007          ERROR 35.
    563 014702 017700 007276    TRAP 35.
    014702 017700 007276    .WORD 0
    564 014706 017701 007274    BR TEST9
    014706 017701 007274    4$: TST @R0
    565 014712          BIT @BIT8,@R0
    014712 104444          BNE 5$
    014712 000000          SAVCIR
    566 014716 000677          MOV @CIR,R0
    014716 000677          SAVSMR
    014716 000677          MOV @SMR,R1
    014716 000677          ERROR 36.
    014716 000677          TRAP 36.
    014716 000677          .WORD 0
    014716 000677          BR TEST9

```

C7

567	014720				5#:	CLRB	0RO	;RESET STATE CHANGE
	014720	042710	000400			BIC	0BIT8,0RO	
568	014724	104000				SCOPE		;SCOPE LOOP MARKER
569	014726					GTS		;FORCE ILLEGAL MESSAGE
	014726	052777	000004	007252		BIS	0BIT2,0SMR	;GO TO STAND BY
570	014734	005037	014750			CLR	7#	;WAIT FOR ILLEGAL MESSAGE
571	014740				6#:	TST12	0RO	
	014740	032710	010000			BIT	0BIT12,0RO	
572	014744	001012				BNE	8#	
573	014746	005227				INC	(PC)+	
574	014750	000000			7#:	.WORD	0	
575	014752	001372				BNE	6#	
576	014754					SAVCIR		;SAVE CIR CONTENT
	014754	017700	007224			MOV	0CIR,RO	;SAVE CIR CONTENT
577	014760					SAVSMR		;SAVE SMR CONTENT
	014760	017701	007222			MOV	0SMR,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			8#:	INC	RO	;POINT TO CIR HIGH BYTE
581	014774	104000				SCOPE		;SCOPE LOOP MARKER
582	014776	105010				CLRB	0RO	;RESET INTERRUPT CONDITIONS
583	015000					TCA		;TAKE CONTROL ASYNC
	015000	052777	000002	007200		BIS	0BIT1,0SMR	;TAKE CONTROL ASYNC
584	015006	005037	015022			CLR	10#	;WAIT FOR CACS
585	015012	122710	000001		9#:	CMPB	01,0RO	
586	015016	001413				BEQ	11#	; **VO2E00**
587	015020	005227				INC	(PC)+	
588	015022	000000			10#:	.WORD	0	
589	015024	001372				BNE	9#	
590	015026					SAVCIR		;SAVE CIR CONTENT
	015026	017700	007152			MOV	0CIR,RO	;SAVE CIR CONTENT
591	015032					SAVSMR		;SAVE SMR CONTENT
	015032	017701	007150			MOV	0SMR,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		11#:	MOV	SMR,R1	;GET ADDRESS OF SMR
595	015052	005201				INC	R1	;ADDRESS HIGH BYTE OF SMR
596	015054	122711	000001			CMPB	01,0R1	;CACS SET AGAIN ?
597	015060	001410				BEQ	12#	;IF EQ, YES
598	015062					SAVCIR		;SAVE CIR CONTENT
	015062	017700	007116			MOV	0CIR,RO	;SAVE CIR CONTENT
599	015066					SAVSMR		;SAVE SMR CONTENT
	015066	017701	007114			MOV	0SMR,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			12#:	SCOPE		;SCOPE LOOP MARKER
603	015104	105010				CLRB	0RO	;RESET INTERRUPT CONDITIONS
604	015106	012705	100000			MOV	0BIT15,R5	;SETUP INTERRUPT MASK
605	015112	012704	015200			MOV	015,R4	;SETUP INTERRUPT RETURN ADDRESS
606	015116					INTENB		;ENABLE IEC INTERRUPT
	015116	052777	000100	007060		BIS	0BIT6,0CIR	;ENABLE INTERRUPT IN IEC 11A

D7

607	015124	105037	177776		CLRB	@#PS				; ALLOW SYSTEM INTERRUPTS
608	015130				SAVCIR					; SAVE CIR CONTEN.
	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			INC	(PC)+				
612	015150	000000			.WORD	0				
613	015152	001375			BNE	13#				
614	015154				SAVIOR					; SAVE IOR CONTENT
	015154	017702	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			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				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		CMPB	@12,@R1				
625	015230	001415			BEQ	19#				
626	015232	005227			INC	(PC)+				
627	015234	000000			.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				TST11	@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				MCLEAR					; RESET IEC-11A
	015320	052777	000040	006656	BIS	@BIT5,@CIR				; EXECUTE MASTER CLEAR
642	015326				LOOP	-1,LOPT6				

E7

```
015326 005327          DEC      (PC)+      ;COUNT LOOPS
015330 177777          .WORD    -1          ;
015332 005737 015330 64$:   TST      64$          ; IS 64$ ZERO ?
015336 001402          BEQ      .+6          ;
015340 000137 014552    JMP      LOPT6         ;
643 015344 012737 177777 015330 MOV     @-1,64$      ;SETUP NEW LOOP COUNT
015352          REMVEC  @VECTOR          ;REMOVE VECTOR          ; **V03E02**
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      66$          ;
015366 012746 023274    MOV     @MTE6,-(SP)  ;SAVE @MTE6 ONTO STACK
015372 004737 005654    CALL   $..WRT        ;INITIATE PRINTOUT
015376 000207          66$:   RETURN;      ;RETURN TO CALLER
```

```

646
647 015400 012746 024222 TEST10: .SBTTL TEST 10 TEST DATA TRANSFER WITH INTERRUPT
      015400 012746 021334      MOV  #VECTOR,#IECINT,#0      ;SET VECTOR      ;**V03E02**
      015404 012746 000000      MOV  #VECTOR,-(SP)      ;SAVE #VECTOR ONTO STACK
      015410 012746 000000      MOV  #IECINT,-(SP)      ;SAVE #IECINT ONTO STACK
      015414 104001      MOV  #0,-(SP)      ;SAVE #0 ONTO STACK
      015416 005737 011670      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 $.WRT      ;INITIATE PRINTOUT
651 015434      LOPT7: MCLR      ;MASTER CLEAR
      015434 052777 000040 006542      BIS  #BIT5,#CIR      ;EXECUTE MASTER CLEAR
652 015442      SACS      ;SET CONTROLLER ACTIVE STATE
      015442 052777 000001 006534      BIS  #BIT0,#CIR      ;SET SACS BIT IN CIR
653 015450      SIC      ;SEND INTERFACE CLEAR
      015450 052777 000100 006530      BIS  #BIT6,#SMR      ;SEND INTERFACE CLEAR
654 015456      13$: 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
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$: BIC  #177400,#CIR      ;RESET INTERRUPT CONDITIONS
676 015610      TCA      ;TAKE CONTROL ASYNCR.
      015610 052777 000002 006370      BIS  #BIT11,#SMR      ;TAKE CONTROL ASYNC
677 015616      5$: TST8 #CIR      ;WAIT FOR STATE CHANGE
      015616 032777 000400 006360      BIT  #BIT8,#CIR
678 015624 001774      BEQ  5$
679 015626      6$: TST8 #SMR      ;

```

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

```

719 016062          15$:  TST9    @SMR      ;WAIT FOR CSBS
    016062 032777 001000 006116 BIT    @BIT9,@SMR
720 016070          BEQ    15$
    016070 001774          TST12   @SMR      ;LACS SET ?
721 016072          BIT    @BIT12,@SMR
    016072 032777 010000 006106 BNE    16$
722 016100          SAVCIR
    016100 001012          MOV    @CIR,R0
723 016102          SAVSMR
    016102 017700 006076          MOV    @SMR,R1
724 016106          SAVIOR
    016106 017701 006074          MOV    @IOR,R2
725 016112          ERROR 52.
    016112 017702 006074          TRAP  52.
726 016116          .WORD  0
    016116 104464          JMP    TEST10
    016120 000000          MOV    @-1,@IOR
727 016122 000137 015400 16$:  TST15   @CIR
    016122 000137 015400 17$:  BIT    @BIT15,@CIR
728 016126 112777 177777 006056 BEQ    17$
    016126 112777 177777 006056 CLR15  @CIR
729 016134          BIC    @BIT15,@CIR
    016134 032777 100000 006042 CLR8   @CIR
730 016142          BIC    @BIT8,@CIR
    016142 001774          DECB  @IOR
731 016144          INC    (PC)+
    016144 042777 100000 006032 .WORD  0
732 016152          BNE    18$
    016152 042777 000400 006024 TST15  @CIR
733 016160          BIT    @BIT15,@CIR
    016160 105377 006026          BEQ    19$
734 016164          SAVCIR
    005227          MOV    @CIR,R0
735 016166          SAVSMR
    000000          MOV    @SMR,R1
736 016170          SAVIOR
    001375          MOV    @IOR,R2
737 016172          ERROR 54.
    016172 032777 100000 006004 TRAP  54.
738 016200          .WORD  0
    001412          JMP    TEST10
739 016202          SAVIOR
    016202 017700 005776          MOV    @IOR,R2
740 016206          SWAB   R2
    016206 017701 005774          CMPB  R2,@377
741 016212          BEQ    20$
    016212 017702 005774          SWAB  R2
742 016216          SAVSMR
    016216 104466          MOV    @SMR,R1
    016220 000000          SAVCIR
743 016222          MOV    @CIR,R0
    000137 015400          ERROR 55.
744 016226          TRAP  55.
    016226 017702 005760          .WORD  0
745 016232          JMP    TEST10
    000302          MOV    @IOR,R2
746 016234          SWAB   R2
    120227 000377          CMPB  R2,@377
747 016240          BEQ    20$
    001411          SWAB  R2
748 016242          SAVSMR
    000302          MOV    @SMR,R1
749 016244          SAVCIR
    016244 017701 005736          MOV    @CIR,R0
750 016250          ERROR 55.
    016250 017700 005730          TRAP  55.
751 016254          .WORD  0
    016254 104467          JMP    TEST10
    016256 000000          TST15  @CIR
752 016260          BIT    @BIT15,@CIR
    000137 015400          ;RESTART SUBTEST
753 016264          ;SECOND "DATA ACCEPTED" ?
    016264 032777 100000 005712

```

754	016272	001012		BNE	21\$; IF NE, YES
755	016274			SAVCIR			; SAVE CIR CONTENT
	016274	017700	005704	MOV	@CIR,R0		; SAVE CIR CONTENT
756	016300			SAVSMR			; SAVE SMR CONTENT
	016300	017701	005702	MOV	@SMR,R1		; SAVE SMR CONTENT
757	016304			SAVIOR			; SAVE IOR CONTENT
	016304	017702	005702	MOV	@IOR,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		21\$:	; GET SECONF0 DATA BYTE
	016320	017702	005666	MOV	@IOR,R2		; SAVE IOR CONTENT
761	016324	000302		SWAB	R2		
762	016326	122702	000376	CMPB	#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	@CIR,R0		; SAVE CIR CONTENT
766	016342			SAVSMR			; SAVE SMR CONTENT
	016342	017701	005640	MOV	@SMR,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

```

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 CLR -(SP) ;ADDRESS CIR HIGH BYTE
781 016370 105077 005610 INC CIR ;RESET INTERRUPT BITS
782 016374 005337 024204 CLRB BCIR ;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 BIC #177400,BCIR ;RESET INTERRUPT BITS
787 016420 INTENB ;ENABLE IEC INTERRUPT
788 016426 052777 000100 005556 BIS #BIT6,BCIR ;ENABLE INTERRUPT IN IEC-11A
789 016432 105037 177776 CLRB #PS ;ENABLE SYSTEM INTERRUPT
790 016436 011677 005554 MOV (SP),BIOR ;SEND A DATA BYTE
791 016442 005037 016444 CLR 2 ;WAIT FOR INTERRUPT
792 016444 000000 1$: INC (PC)+ ;
793 016446 001375 2$: .WORD 0 ;
794 016450 BNE 1$ ;
795 016450 SAVIEC ;SAVE ALL IEC REGISTERS
796 016450 017700 005530 MOV BCIR,R0 ;SAVE CIR CONTENT
797 016454 017701 005526 MOV #SMR,R1 ;SAVE SMR CONTENT
798 016460 017702 005526 MOV #IOR,R2 ;SAVE IOR CONTENT
799 016464 ERROR 60. ;NO INTERRUPT TO DATA SENDING
800 016464 104474 TRAP 60. ;ERROR 60.
801 016466 000000 .WORD 0 ;ERROR COUNT
802 016470 000137 015400 JMP TEST10 ;RESTART TEST
803 016474 103004 3$: BCC 4$ ;IF CC, INTERRUPT OK
804 016476 104475 ERROR 61. ;ILLEGAL INTERRUPT TO DATA SENDING
805 016476 104475 TRAP 61. ;ERROR 61.
806 016500 000000 .WORD 0 ;ERROR COUNT
807 016502 000137 015400 JMP TEST10 ;RESTART TEST
808 016506 017703 005500 4$: MOV #IOR,R3 ;READ BACK DATA BYTE
809 016512 105003 CLRB R3 ;PREPARE DATA
810 016514 000303 SWAB R3 ;
811 016516 IF R3 EQ (SP) GOTO 5$ ;
812 016516 020316 CMP R3,(SP) ;COMPARE R3 AND (SP)
813 016520 001002 BNE .+6 ;
814 016522 000137 016554 JMP 5$ ;
815 016526 SAVIEC ;SAVE ALL IEC REGISTERS
816 016526 017700 005452 MOV BCIR,R0 ;SAVE CIR CONTENT
817 016532 017701 005450 MOV #SMR,R1 ;SAVE SMR CONTENT
818 016536 017702 005450 MOV #IOR,R2 ;SAVE IOR CONTENT
819 016542 011604 MOV (SP),R4 ;SAVE 'SHOULD BE ' VALUE
820 016544 ERROR 62. ;DATA CHECK ON IEC BUS
821 016544 104476 TRAP 62. ;ERROR 62.
822 016546 000000 .WORD 0 ;ERROR COUNT
823 016550 000137 015400 807: JMP TEST10 ;RESTART TEST
824 016554 005737 016604 5$: IF RANMOD NE #0 GOTO 6$
825 016560 001402 TST RANMOD ;IS RANMOD ZERO ?
826 BEQ .+6 ;

```

809	016562	000137	016606		JMP	6\$		
	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		RANMOD:	.WORD	0		;RANDOM MODE FLAG
813	016606	004737	011672	6\$:	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	000222		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			7\$:	TST15	@CIR		
	016654	032777	100000	005322	BIT	#BIT15,@CIR		
825	016662	001015			BNE	9\$		
826	016664	005227			INC	(PC)+		
827	016666	000000		8\$:	.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			9\$:	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		10\$:	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"

L7

```

      017010 104501          TRAP      65.          ;ERROR 65.
      017012 000000          .WORD    0           ;ERROR COUNT
843 017014 000137 015400    JMP      TEST10        ;RESTART SUBTEST
844 017020          11$:   MCLEAR          ;MASTER CLEAR
      017020 052777 000040 005156    BIS      @BIT5,@CIR    ;EXECUTE MASTER CLEAR
845 017026          TSTEND 7           ;END OF SUBTEST 7
      017026 005737 011670    YST      MSGFLG       ;PRINTOUT OR NOT?
      017032 001004          BNE      68$
      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

```

```

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

```

887	017302						SAVIEC		;SAVE IEC REGISTERS
	017302	017700	004676				MOV	@CIR,R0	;SAVE CIR CONTENT
	017306	017701	004674				MOV	@SMR,R1	;SAVE SMR CONTENT
	017312	017702	004674				MOV	@IOR,R2	;SAVE IOR CONTENT
888	017316						ERROR	67.	;NO LACS AFTER (LTN-GTS-TCA-GTS)
	017316	104503					TRAP	67.	;ERROR 67.
	017320	000000					.WORD	0	;ERROR COUNT
889	017322	000137	017046				JMP	TEST11	;RESTART SUBTEST
890	017326	042777	177400	004650	9\$:		BIC	@177400,@CIR	;RESET ALL INTERRUPT CONDITIONS
891	017334						TCA		;GO BACK TO CACS
	017334	052777	000002	004644			BIS	@BIT1,@SMR	;TAKE CONTROL ASYNC
892	017342				10\$:		TSTB	@SMR	;WAIT FOR CACS
	017342	032777	000400	004636			BIT	@BIT8,@SMR	
893	017350	001774					BEQ	10\$	
894	017352						LUN		;LOCAL UNLISTEN
	017352	052777	000040	004626			BIS	@BIT5,@SMR	
895	017360	000240					NOP		
896	017362	000240					NOP		
897	017364	000240					NOP		
898	017366						GTS		;GO AGAIN TO STAND BY STATE
	017366	052777	000004	004612			BIS	@BIT2,@SMR	;GO TO STAND BY
899	017374				11\$:		TST9	@SMR	;WAIT FOR STAND BY STATE
	017374	032777	001000	004604			BIT	@BIT9,@SMR	
900	017402	001774					BEQ	11\$	
901	017404						TST12	@SMR	;IS LACS STILL SET ?
	017404	032777	010000	004574			BIT	@BIT12,@SMR	
902	017412	001412					BEQ	12\$	
903	017414						SAVIEC		;IF EQ, NO
	017414	017700	004564				MOV	@CIR,R0	;SAVE ALL IEC REGISTERS
	017420	017701	004562				MOV	@SMR,R1	;SAVE CIR CONTENT
	017424	017702	004562				MOV	@IOR,R2	;SAVE SMR CONTENT
904	017430						ERROR	70.	;SAVE IOR CONTENT
	017430	104506					TRAP	70.	;LACS STILL SET AFTER (LTN-GTS-TCA-LUN-GTS)
	017432	000000					.WORD	0	;ERROR 70.
905	017434	000137	017046				JMP	TEST11	;ERROR COUNT
906	017440				12\$:		MCLEAR		;RESTART SUBTEST
	017440	052777	000040	004536			BIS	@BIT5,@CIR	;MASTER CLEAR
907	017446						LOOP	-1,LOPT8	;EXECUTE MASTER CLEAR
	017446	005327					DEC	(PC)+	;COUNT LOOPS
	017450	177777			64\$:		.WORD	-1	
	017452	005737	017450				TST	64\$;IS 64\$ ZERO ?
	017456	001401					BEQ	.+4	
	017460	000601					BR	LOPT8	
	017462	012737	177777	017450			MOV	#-1,64\$;SETUP NEW LOOP COUNT
908	017470						TSTEND	8	
	017470	005737	011670				TST	MSGFLG	;PRINTOUT OR NOT?
	017474	001004					BNE	66\$	
	017476	012746	023472				MOV	@MTE8,-(SP)	;SAVE @MTE8 ONTO STACK
	017502	004737	005654				CALL	\$..WRT	;INITIATE PRINTOUT
	017506	000207			66\$:		RETURN;		;RETURN TO CALLER

910					.SBTTL	TEST 12 -- TEST FUNCTION OF <BLOCK DAC>	
911	017510	005737	011670		TEST12: 1ST	MSGFLG	;PRINTOUT OR NOT?
912	017514	001004			BNE	LOPT9	
913	017516				WRITE	MT9S	;START MESSAGE TEST 12
	017516	012746	023507		MOV	#MT9S,-(SP)	;SAVE #MT9S ONTO STACK
	017522	004737	005654		CALL	\$.WRT	;INITIATE PRINTOUT
914	017526	104000			LOPT9: SCOPE		;SET SCOPE LOOP MARKER
915	017530				MCLEAR		;MASTER CLEAR
	017530	052777	000040	004446	BIS	#BIT5,@CIR	;EXECUTE MASTER CLEAR
916	017536				SET3	@CIR	;TRY TO SET <BLOCK DAC>
	017536	052777	000010	004440	BIS	#BIT3,@CIR	
917	017544				TST3	@CIR	;IS <BLOCK DAC> SET ?
	017544	032777	000010	004432	BIT	#BIT3,@CIR	
918	017552	001011			BNE	1:	;IF NE, YES
919	017554				SAVIEC		;SAVE ALL IEC REGISTERS
	017554	017700	004424		MOV	@CIR,R0	;SAVE CIR CONTENT
	017560	017701	004422		MOV	@SMR,R1	;SAVE SMR CONTENT
	017564	017702	004422		MOV	@IOR,R2	;SAVE IOR CONTENT
920	017570				ERROR	71.	; <BLOCK DAC> NOT SET
	017570	104507			TRAP	71.	;ERROR 71.
	017572	000000			.WORD	0	;ERROR COUNT
921	017574	000745			BR	TEST12	;RESTART SUBTEST
922	017576				1: CLR3	@CIR	;TRY TO RESET <BLOCK DAC>
	017576	042777	000010	004400	BIC	#BIT3,@CIR	
923	017604				TST3	@CIR	;IS <BLOCK DAC> RESET
	017604	032777	000010	004372	BIT	#BIT3,@CIR	
924	017612	001411			BEQ	2:	;IF EQ, YES
925	017614				SAVIEC		;SAVE ALL IEC REGISTERS
	017614	017700	004364		MOV	@CIR,R0	;SAVE CIR CONTENT
	017620	017701	004362		MOV	@SMR,R1	;SAVE SMR CONTENT
	017624	017702	004362		MOV	@IOR,R2	;SAVE IOR CONTENT
926	017630				ERROR	72.	; <BLOCK DAC> NOT RESET
	017630	104510			TRAP	72.	;ERROR 72.
	017632	000000			.WORD	0	;ERROR COUNT
927	017634	000725			BR	TEST12	;RESTART SUBTEST
928	017636	104000			2: SCOPE		;SET SCOPE LOOP MARKER
929	017640				MCLEAR		;MASTER CLEAR
	017640	052777	000040	004336	BIS	#BIT5,@CIR	;EXECUTE MASTER CLEAR
930	017646				SACS		;SET CONTROLLER ACTIVE
	017646	052777	000001	004330	BIS	#BIT0,@CIR	;SET SACS BIT IN CIR
931	017654				SIC		;SEND INTERFACE CLEAR
	017654	052777	000100	004324	BIS	#BIT6,@SMR	;SEND INTERFACE CLEAR
932	017662				3: TST13	@SMR	;WAIT FOR SIAS OFF
	017662	032777	020000	004316	BIT	#BIT13,@SMR	
933	017670	001374			BNE	3:	
934	017672				SET3	@CIR	;SET <BLOCK DAC>
	017672	052777	000010	004304	BIS	#BIT3,@CIR	
935	017700	013777	024220	004304	MOV	LISTNR,@IOR	;SEND LISTENER ADDRESS
936	017706				4: TST15	@CIR	;WAIT FOR <DATA ACCEPTED>
	017706	032777	100000	004270	BIT	#BIT15,@CIR	
937	017714	001774			BEQ	4:	
938	017716	042777	177400	004260	BIC	#177400,@CIR	;RESET ALL INTERRUPT CONDITIONS
939	017724	013777	024216	004260	MOV	TALKER,@IOP	;SEND TALKER ADDRESS
940	017732				5: TST15	@CIR	;WAIT FOR <DATA ACCEPTED>
	017732	032777	100000	004244	BIT	#BIT15,@CIR	
941	017740	001774			BEQ	5:	
942	017742	042777	177400	004234	BIC	#177400,@CIR	;RESET ALL INTERRUPT CONDITIONS

943	017750				GTS				;GOTO <STAND BY> STATE
	017750	052777	000004	004230	BIS	#BIT2,#SMR			;GO TO STAND BY
944	017756				TST12	#SMR	6#:		;WAIT FOR <LACS>
	017756	032777	010000	004222	BIT	#BIT12,#SMR			
945	017764	001774			BEQ	6#			
946	017766	042777	177400	004210	BIC	#177400,#CIR			;RESET ALL INTERRUPT CONDITIONS
947	017774	005005			CLR	R5			;SETUP INTERRUPT MASK
948	017776	012704	020032		MOV	#8#,#R4			;SETUP INTERRUPT RETURN ADDRESS
949	020002				INTEND				;ENABLE IEC INTERRUPT
	020002	052777	000100	004174	BIS	#BIT6,#CIR			;ENABLE INTERRUPT IN IEC-11A
950	020010	005037	177776		CLR	#OPS			;ENABLE SYSTEM INTERRUPTS
951	020014	012777	000333	004170	MOV	#333,#IOR			;SEND A DATA PATTERN
952	020022	005227			INC	(PC),	7#:		;WAIT A SHORT TIME
953	020024	000000			.WORD	0			
954	020026	001375			BNE	7#			
955	020030	000412			BR	9#			;OK, NO INTERRUPT OCCURED
956	020032				SAVIFC		8#:		;SAVE ALL IEC REGISTERS
	020032	017700	004146		MOV	#CIR,#R0			;SAVE CIR CONTENT
	020036	017701	004144		MOV	#SMR,#R1			;SAVE SMR CONTENT
	020042	017702	004144		MOV	#IOR,#R2			;SAVE IOR CONTENT
957	020046				ERROR	73.			;INTERRUPT WITH <BLOCK DAC>
	020046	104511			TRAP	73.			;ERROR 73.
	020050	000000			.WORD	0			;ERROR COUNT
958	020052	000137	017510		JMP	TEST12			;RESTART SUBTEST
959	020056	017703	004130		MOV	#IOR,#R3	9#:		;GET INPUT BUFFER CONTENT
960	020062	105003			CLRB	R3			;PREPARE FOR CHECK
961	020064	000303			SWAB	R3			
962	020066				IF R3 EQ #333 GOTO 10#				
	020066	020327	000333		CMP	R3,#333			;COMPARE R3 AND #333
	020072	001002			BNE	.+6			
	020074	000137	020124		JMP	10#			
963	020100				SAVIEC				;SAVE ALL IEC REGISTERS
	020100	017700	004100		MOV	#CIR,#R0			;SAVE CIR CONTENT
	020104	017701	004076		MOV	#SMR,#R1			;SAVE SMR CONTENT
	020110	017702	004076		MOV	#IOR,#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		10#:		;MASTER CLEAR
	020124	052777	000040	004052	BIS	#BIT5,#CIR			;EXECUTE MASTER CLEAR
967	020132				LOOP	20.,#LOPT9			
	020132	005327			DEC	(PC),			;COUNT LOOPS
	020134	000024			.WORD	20.	65#:		
	020136	005737	020134		TST	65#			;IS 65# ZERO?
	020142	001402			BEQ	.+6			
	020144	000137	017526		JMP	LOPT9			
	020150	012737	000024	020134	MOV	#20.,#65#			;SETUP NEW LOOP COUNT
968	020156				TSTEND	9			;END OF TEST 12
	020156	005737	011670		TST	MSGFLG			;PRINTOUT OR NOT?
	020162	001004			BNE	67#			
	020164	012746	023565		MOV	#MTE9,-(SP)			;SAVE #MTE9 ONTO STACK
	020170	004737	005654		CALL	#,WRT			;INITIATE PRINTOUT
	020174	000207			RETURN;		67#:		;RETURN TO CALLER

970					.SBITL	TEST 13 -- TEST FUNCTION OF <RSV>, <SRQ> AND <SPAS>	
971	020176	005737	011670		TEST13: TST	MSGFLG	; PRINTOUT OR NOT?
972	020202	001004			BNE	LOPT10	
973	020204				WRITE	MT10S	; START MESSAGE TEST 13
	020204	012746	023602		MOV	#MT10S, -(SP)	; SAVE #MT10S ONTO STACK
	020210	004737	005654		CALL	\$.WRT	; INITIATE PRINTOUT
974	020214	104000			LOPT10: SCOPE		; SCOPE LOOP MARKER
975	020216				MCLEAR		; MASTER CLEAR
	020216	052777	000040	003760	BIS	#BIT5, @CIR	; EXECUTE MASTER CLEAR
976	020224				SACS		; SET CONTROLLER ACTIVE
	020224	052777	000001	003752	BIS	#BIT0, @CIR	; SET SACS BIT IN CIR
977	020232				SIC		; SEND INTERFACE CLEAR
	020232	052777	000100	003746	BIS	#BIT6, @SMR	; SEND INTERFACE CLEAR
978	020240				1\$: TST13	@SMR	; WAIT FOR SIAS TO GO OFF
	020240	032777	020000	003740	BIT	#BIT13, @SMR	
979	020246	001374			BNE	1\$	
980	020250	042777	177400	003726	BIC	#177400, @CIR	; RESET ALL INTERRUPT CONDITIONS
981	020256				SET4	@CIR	; SET <RSV>
	020256	052777	000020	003720	BIS	#BIT4, @CIR	
982	020264	005037	020302		CLR	3\$; WAIT FOR <SRQ>
983	020270				2\$: TST14	@CIR	
	020270	032777	040000	003706	BIT	#BIT14, @CIR	
984	020276	001015			BNE	4\$	
985	020300	005227			INC	(PC),	
986	020302	000000			3\$: .WORD	0	
987	020304	001371			BNE	2\$	
988	020306				SAVIEC		; SAVE ALL IEC REGISTERS
	020306	017700	003672		MOV	@CIR, R0	; SAVE CIR CONTENT
	020312	017701	003670		MOV	@SMR, R1	; SAVE SMR CONTENT
	020316	017702	003670		MOV	@IOR, R2	; SAVE IOR CONTENT
989	020322				ERROR	75.	; NO <SRQ> TO <RSV>
	020322	104513			TRAP	75.	; ERROR 75.
	020324	000000			.WORD	0	; ERROR COUNT
990	020326	000137	020176		JMP	TEST13	; RESTART TESTPART
991	020332				4\$: TST7	@CIR	; SEE IF INT BIT SET ; **V02E00**
	020332	032777	000200	003644	BIT	#BIT7, @CIR	
992	020340	001004			BNE	41\$; BRANCH IF SET ; **V02E00**
993	020342				ERROR	111.	; INT BIT NOT SET AT SRQ ; **V02E00**
	020342	104557			TRAP	111.	; ERROR 111.
	020344	000000			.WORD	0	; ERROR COUNT
994	020346	000137	020176		JMP	TEST13	; **V02E00**
995	020352				41\$: INHSRQ		; SET SRQ INT INHIBIT ; **V02E00**
	020352	052777	002000	003624	BIS	#BIT10, @CIR	; SET INHIBIT SRQ INTERRUPT
996	020360				TST7	@CIR	; INTERRUPT BIT SET? ; **V02E00**
	020360	032777	000200	003616	BIT	#BIT7, @CIR	
997	020366	001404			BEQ	42\$; BRANCH IF NOT ; **V02E00**
998	020370				ERROR	112.	; INT BIT NOT CLEAR ; **V02E00**
	020370	104560			TRAP	112.	; ERROR 112.
	020372	000000			.WORD	0	; ERROR COUNT
999	020374	000137	020176		JMP	TEST13	; **V02E00**
1000	020400				42\$: CLR10	@CIR	; RESET SRQ INT INHIBIT ; **V02E00**
	020400	042777	002000	003576	BIC	#BIT10, @CIR	
1001	020406				TST7	@CIR	; LOOK FOR INT BIT ; **V02E00**
	020406	032777	000200	003570	BIT	#BIT7, @CIR	
1002	020414	001004			BNE	43\$; BRANCH IF SET AGAIN ; **V02E00**
1003	020416				ERROR	113.	; BIT 7 STILL CLEAR ; **V02E00**
	020416	104561			TRAP	113.	; ERROR 113.

```

020420 000000 .WORD 0 ;ERROR COUNT
1004 020422 000137 020176 JMP TEST13 ; **V02E00**
1005 020426 013777 024220 003556 43$: MOV LISTNR,@IOR ;SEND LISTENER ADDRESS
1006 020434 032777 100000 003542 5$: TST15 @CIR ;WAIT FOR <DATA ACCEPTED>
BIT @BIT15,@CIR
BEQ 5$
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 032777 100000 003516 6$: TST15 @CIR ;WAIT FOR <DATA ACCEPTED>
BIT @BIT15,@CIR
BEQ 6$
1011 020466 001774 BIC @177400,@CIR ;RESET ALL INTERRUPT CONDITIONS
1012 020470 042777 177400 003506 MOV @30,@IOR ;SEND <SERIAL POLL ENABLE>
1013 020476 012777 000030 003506 7$: TST15 @CIR ;WAIT FOR <DATA ACCEPTED>
BIT @BIT15,@CIR
BEQ 7$
1014 020504 032777 100000 003472 BIC @177400,@CIR ;RESET ALL INTERRUPT CONDITIONS
1015 020512 001774 GTS ;GOTO <STAND BY> STATE
1016 020514 042777 177400 003462 BIS @BIT2,@SMR ;GO TO STAND BY
1017 020522 052777 000004 003456 CLR 9$ ;
1018 020530 005037 020546 8$: TST15 @SMR ;WAIT FOR <SPAS>
BIT @BIT15,@SMR
BNE 10$
1019 020534 032777 100000 003444 9$: .WORD 0 ;
BNE 8$ ;
1020 020542 001015 SAVIEC ;SAVE ALL IEC REGISTERS
1021 020544 005227 MOV @CIR,R0 ;SAVE CIR CONTENT
1022 020546 000000 MOV @SMR,R1 ;SAVE SMR CONTENT
1023 020550 001371 MOV @IOR,R2 ;SAVE IOR CONTENT
1024 020552 017700 003426 ERROR 76. ; NO <SPAS> TO <SPE>
020556 017701 003424 TRAP 76. ;ERROR 76.
020562 017702 003424 .WORD 0 ;ERROR COUNT
1025 020566 104514 JMP TEST13 ;RESTART SUBTEST
020570 000000 BIC @177400,@CIR ;RESET ALL INTERRUPT CONDITIONS
1026 020572 000137 020176 003400 10$: MOV @BIT6,@IOR ;SEND STATUS BYTE
1027 020576 042777 177400 003400 11$: TST15 @CIR ;WAIT FOR <DATA ACCEPTED>
BIT @BIT15,@CIR
BEQ 11$
1028 020604 012777 000100 003400 SAVIOR ;SAVE IOR CONTENT
1029 020612 032777 100000 003364 MOV @IOR,R2 ;SAVE IOR CONTENT
1030 020620 001774 CLR R2 ;PREPARE FOR DATA CHECK
1031 020622 017702 003364 SWAB R2 ;
1032 020626 105002 IF R2 EQ @BIT6 GOTO 12$ ;
1033 020630 000302 CMP R2,@BIT6 ;COMPARE R2 AND @BIT6
1034 020632 020227 000100 BNE .+6 ;
020636 001002 JMP 12$ ;
1035 020644 000137 020670 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 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 032777 040000 003306 12$: TST14 @CIR ;IS <SRQ> OFF ?
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	@BIT8,@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	+6		
	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
1065 021114 005737 011670 TEST14: .SBTTL TEST 14 --- TEST RPP BIT R/W
1066 021120 001004 BNE MSGFLG ;PRINTOUT OR NOT?
1067 021122 WRITE LOPT11
      021122 012746 023711 MOV MT11S ;START MESSAGE TEST 14
      021126 004737 005654 CALL #MT11S,-(SP) ;SAVE #MT11S ONTO STACK
1068 021132 104000 LOPT11: $.WRT ;INITIATE PRINTOUT
1069 021134 SCOPE ;SCOPE LOOP MARKER
      021134 052777 000040 003042 MCLEAR ;MASTER CLEAR
1070 021142 BIS #BIT5,@CIR ;EXECUTE MASTER CLEAR
      021142 052777 000001 003034 SACS ;SET <CONTROLLER ACTIVE> STATE
1071 021150 BIS #BIT0,@CIR ;SET SACS BIT IN CIR
      021150 052777 000100 003030 SIC ;SEND <INTERFACE CLEAR>
1072 021156 BIS #BIT6,@SMR ;SEND INTERFACE CLEAR
      021156 032777 020000 003022 3$: TST13 @SMR ;WAIT FOR <SIAS TO GO OFF>
      BIT #BIT13,@SMR
1073 021164 BNE 3$
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 BNE 1$ ;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 CLR3 @SMR ;TRY TO RESET <RPP>
      021230 042777 000010 002750 1$: BIC #BIT3,@SMR
1081 021236 TST3 @SMR ;IS BIT RESET ?
      021236 032777 000010 002742 BIT #BIT3,@SMR
1082 021244 BEQ 2$ ;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 LOOP -1,LOPT11 ;COUNT LOOPS
      021272 005327 DEC (PC)+
      021274 177777 .WORD -1
      021276 005737 021274 64$: TST 64$ ;IS 64$ ZERO ?
      021302 001401 BEQ .+4
      021304 000712 BR LOPT11
      021306 012737 177777 021274 MOV # -1,64$ ;SETUP NEW LOOP COUNT
1087 021314 TSTEND 11
      021314 005737 011670 TST MSGFLG ;PRINTOUT OR NOT?
      021320 001004 BNE 66$
      021322 012746 023752 MOV #MTE11,-(SP) ;SAVE #MTE11 ONTO STACK
      021326 004737 005654 CALL $.WRT ;INITIATE PRINTOUT
      021332 000207 66$: RETURN ;RETURN TO CALLER

```

```

1089          SBTTL      IEC-INTERRUPT HANDLER
1090 021334          IECINT: PUSH      R0,R1,R2
          021334 010046          MOV      R0,-(SP)          ;SAVE R0 ONTO STACK
          021336 010146          MOV      R1,-(SP)          ;SAVE R1 ONTO STACK
          021340 010246          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 INYERRUPT MASK INTO LOW BYTE
1095 021366          CLR0      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          ;SET ERROR INDICATOR ;**V03E01**
1103 021420 000114          JMP      (R4)          ;RETURN
1104 021422 010466 000006          1$: MOV      R4,6(SP)          ;BUILD RETURN ADDRESS
1105 021426 013704 024204          MOV      CIR,R4          ;PREPARE TO CLEAR INTERRUPT BITS
1106 021432 005204          INC      R4          ;
1107 021434 105014          CLR0     (R4)          ;CLEAR INTERRUPT BITS
1108 021436          POP      R2,R1,R0
          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          ;RETURN FROM INTERRUPT
1110 021446 062706 000004          TOUT: ADD      @4,SP          ;INCREASE SP BY 4
1111 021452          WRITE     MNOAV
          021452 012746 023767          MOV      @MNOAV,-(SP)          ;SAVE @MNOAV ONTO STACK
          021456 004737 005654          CALL     $.WRT          ;INITIATE PRINTOUT
1112 021462          DUMP      OCT,R0
          021462 010046          MOV      R0,-(SP)          ;SAVE R0 ONTO STACK
          021464 005046          CLR      -(SP)
          021466 004737 006400          CALL     $.DMP          ;ENTER DUMP ROUTINE
1113 021472 005737 007152          1$: TST      OUTFLG
1114 021476 001375          BNE     1$
1115 021500 012737 000006 000004          MOV      @6,@4
1116 021506 000111          JMP      (R1)

```

IEC11-A DIAGNOSTIC
SUBTEST ADDRESS TABLE

MACRO M1200 30-MAR-84 16:02 PAGE 50

I8

SEQ 99

1118

.SBTTL SUBTEST ADDRESS TABLE

```

1120 .SBTTL MESSAGES
1121 .NLIST 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 MES6: .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                                     .SBTTL  CONSTANTS AND PARAMETERS
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  IORHB: .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                                     .END  START
```

ADDRIN	011170	LAB11	011432	MTE5	023113	SFLAG	011426	VECA	024224
BIT0	000001	LISTNK	024230	MTE5A	023165	SINGLE	011506	VECAIN	011070
BIT1	000002	LOPT1	012124	MTE6	023274	SISDR0	172200	VECTOR	024222
BIT10	002000	LOPT10	020214	MTE7	023373	SIGN	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	SRO	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
CTR	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
LMP..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..MM	002006	TEST5	013302	\$.DDC	006520
EMTBL	002416	MER1	010304	M..RET	001726	TEST6	013444	\$.DEC	005270
EMTND	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
IORHR	024210	MKBOVF	007213	PSDSWR	007160	UDSARO	177660	\$.NDR	004304
I..DMP	000001	MNOAV	023767	RA	011734	UDSDRO	177620	\$.OCT	005112
KBBEND	007146	MNOSUB	010032	RANDOM	011672	UISARO	177640	\$.PRO	006136
KBBPNT	007146	MPWRFL	007704	RANMOD	016604	UISAR4	177650	\$.PWR	003126
KBBUFF	007026	MSGFLG	011670	RB	011736	UISAR5	177652	\$.RED	005020
KDSARO	172360	MSYERR	007503	RED..E	005626	UISAR6	177654	\$.RRS	003400
KSDRO	172320	MTE1	022371	RGAIN	010624	UISAR7	177656	\$.RSV	003360
KISARO	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								
	000000	000							
		001							

IEC11-A DIAGNOSTIC
SYMBOL TABLE

MACRO M1200 30-MAR-84 16:02 PAGE 52-2

M8

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
ZIEBAO,ZIEBAO/-SP=NEWMAC/ML,CDM,ZIEBAO

```

;          TABLE OF CON.....B1
;          .....C1
;          .....D1
;          .....E1
;          .....F1
;          .SBTTL IEC.....G1
; ERROR 21 THE .....H1
; ERROR 37 NO <.....I1
; ERROR 55 FIRS.....J1
; ERROR 75 NO <.....K1
; ERROR 114 CIR .....L1
;          *****.....M1
;          C)USING A PR.....N1

```

```

;          PRES.....B2
;          THE .....C2
; TABLE OF CONTENTS .....D2
;          .....E2
LOW-CORE .....F2
LOW-CORE .....G2
MEMORY TEST WITH MEM.....H2
MEMORY TEST WITH MEM.....I2
$.MAP MAPPING ROUTI.....J2
$.EMT .....K2
$.EMT .....L2
$.TRP .....M2
$.TRP .....N2

```

```

$.IOT .....B3
$.PWR .....C3
$.RSV .....D3
$.RRS .....E3
$.KBI .....F3
$.INP .....G3
$.INP .....H3
$.IAY .....I3
$.IAY .....J3
$.IAA .....K3
$.IAD .....L3
$.IAD .....M3
$.IAD .....N3

```

```

$.IAE .....B4
$.RED .....C4
$.RED .....D4
$.RED .....E4
$.RED .....F4
$.WRT .....G4
$.WRT .....H4
$.KBO .....I4
$.PRO .....J4
$.STX .....K4
$.DMP .....L4
$.DMP .....M4
$.DMP .....N4

```

```

$.ADP .....B5
$.TOT .....C5
$.BUF .....D5
$.BUF .....E5
$.BUF .....F5
$.BUF .....G5
$.BUF .....H5
TEST SELECT ROUTINE .....I5
TEST SELECT ROUTINE .....J5
LOCAL MACRO DEFINITI.....K5
LOCAL MACRO DEFINITI.....L5
TEST LON SWITCH FEAT.....M5
TEST 1 EXECUTE MAST.....N5

```

```

TEST 2 TEST SACS BI.....B6
TEST 3 TEST CIR BIT.....C6
TEST 3 TEST CIR BIT.....D6
TEST 4 TEST SACS SI.....E6
TEST 4 TEST SACS SI.....F6
TEST 4 TEST SACS SI.....G6
TEST 5 TEST INT ENB.....H6
TEST 6 TEST INTERRU.....I6
TEST 6 TEST INTERRU.....J6
TEST 7 TEST STATE C.....K6
TEST 7 TEST STATE C.....L6
TEST 7 TEST STATE C.....M6
TEST 8 - IOR DATA TE.....N6

```

```

TEST 9 TEST STATE C.....B7
TEST 9 TEST STATE C.....C7
TEST 9 TEST STATE C.....D7
TEST 9 TEST STATE C.....E7
TEST 10 TEST DATA TR.....F7
TEST 10 TEST DATA TR.....G7
TEST 10 TEST DATA TR.....H7
TEST 10 TEST DATA TR.....I7
TEST 10A -- RANDOM.....J7
TEST 10A -- RANDOM.....K7
TEST 10A -- RANDOM.....L7
TEST 11 -- TEST FUNC.....M7
TEST 11 -- TEST FUNC.....N7

```

```

TEST 12 -- TEST FUNC.....B8
TEST 12 -- TEST FUNC.....C8
TEST 13 -- TEST FUNC.....D8
TEST 13 -- TEST FUNC.....E8
TEST 13 -- TEST FUNC.....F8
TEST 14 --- TEST RPP.....G8
IEC INTERRUPT HANDLE.....H8
SUBTEST ADDRESS TABL.....I8
MESSAGE .....J8
CONSTANTS AND PARAME.....K8
SYMBOL TABLE .....L8
SYMBOL TABLE .....M8

```