IEC11-A

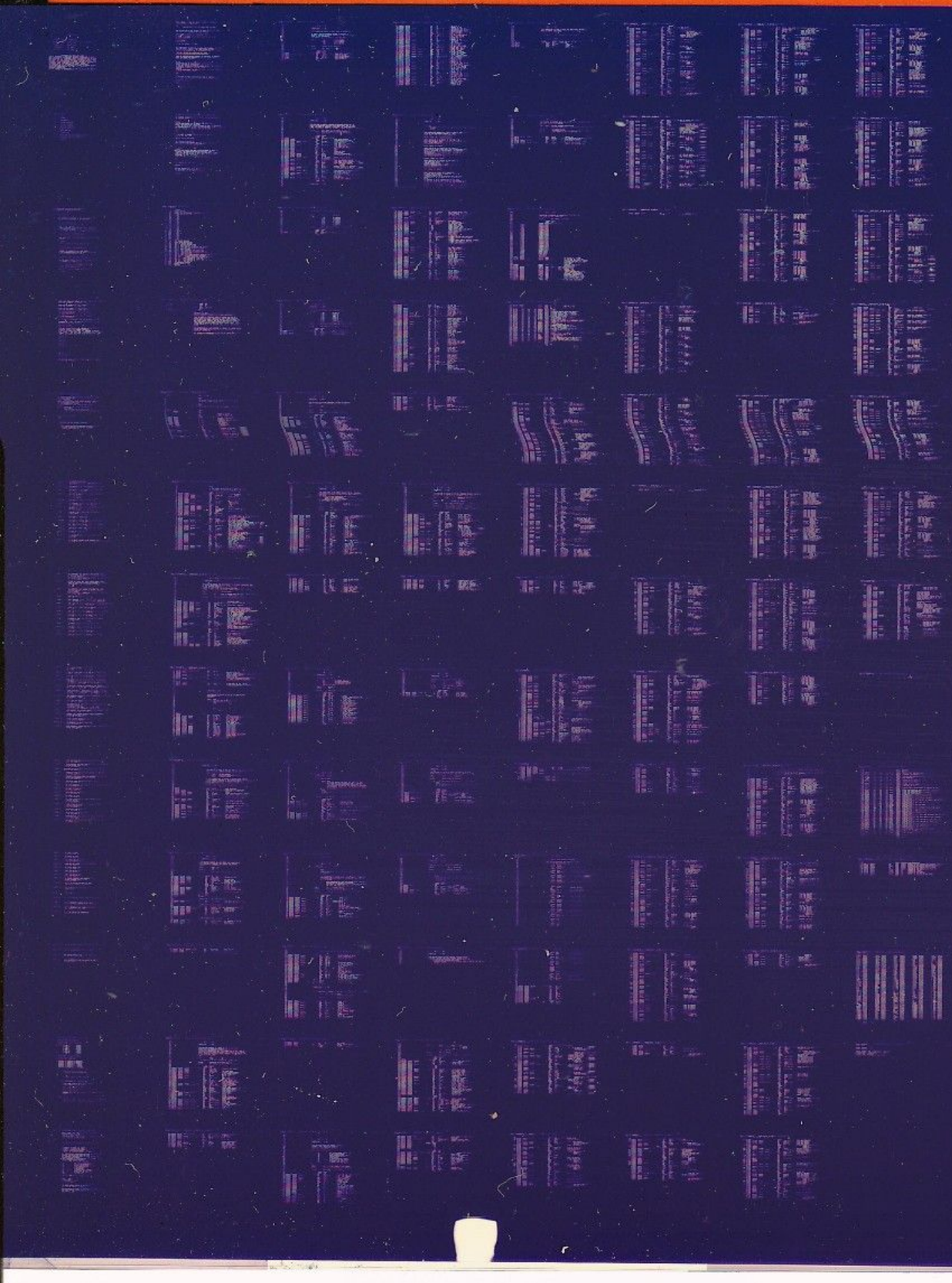
IEC11-A POP-11 DIAG

CZIEBAO

AH-T884A-MC

FICHE 01 OF 01

JUL 1984
digital
Made In USA



IDENTIFICATION

PRODUCT CODE: AC T883A MC

PRODUCT TITLE: CZIEBAO IEC11 A PDP 11 DIAGNOSTIC

PRODUCT DATE: JANUARY 1984

DEPARTMENT: CSS, MUNICH

AUTHOR: BERT HUBER

TABLE OF CONTENTS

```
;
;       ****************************************
;       *    DIAGNOSTIC PROGRAM DESCRIPTION    *
;       ****************************************
;
;
;       1.0   A B S T R A C T
;             ***************
;
;             THIS PROGRAM IS DESIGNED TO TEST AND EXERCISE THE
;             IEC-11A IEC BUS CONTROLLER FOR THE PDP/11.
;
;
;       2.0    R E Q U I R E M E N T S
;              ***********************
;
;         A.    PDP   11/XX  SYSTEM
;               CONSOLE TERMINAL WITH STANDARD UNIBUS ADDRESSES.
;         B.    A MINIMUM OF 5K MEMORY.
;         C.    ALL STANDARD PROCESSOR TESTS MUST BE RUN SUCCESSFULLY
;
;
;       3.0     L O A D I N G   P R O C E D U R E
;               *********************************
;
;               BOOT XXDP MEDIA, THEN RUN DIAGNOSTIC WITH FOLLOWING
;               COMMAND:
;
;               R ZIEB??
;
;               NOTE: ?? REFERS TO VERSION AND REV LEVEL.
;
;       4.0     S T A R T I N G   P R O C E D U R E
;               ***********************************
;
;               PROGRAM  IS STARTED AT LOC 1000 (OCTAL)
;               IF NOT OTHERWISE SPECIFIED THE ABOVE NOTED ADDRESS IS
;               ALSO THE PROGRAM RESTART ADDRESS.
;
;
;       5.0     O P E R A T I N G   P R O C E D U R E
;               *************************************
;
;               THE FOLLOWING INITIAL DIALOGUE IS PERFORMED AFTER
;               THE PROGRAM IS STARTED :
;
;                       CDM V3.2 -- CSS DIAGNOSTIC MONITOR
;
;               HIGHEST MEMORY ADDRESS:XXXXXX
;
;               IEC-11A -   IEC BUS CONTROLLER BASIC TESTS
;
;               AC T883A MC
```

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

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

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

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

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

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

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

NEXT TEST TO RUN?

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

START TEST 1

END TEST 1

START TEST 2

END TEST 2

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

END TEST 14

END OF TESTING


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

```
;
;       6.0     E R R O R   H A N D L I N G
;               ***************************
;
;               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     R E S T R I C T I O N S
;               ***********************
;               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      SIAS DOES NOT BRING CACS.
;               RO = CIR        R1 = SMR

; ERROR 13      SIAS DOES NOT BECOME ZERO AFTER MORE THAN 200 MS.
;               RO = CIR        R1 = SMR

; ERROR 14      NO 'STATE CHANGE' AFTER 'SIAS' 1 TO 0 CHANGE.
;               RO = CIR        R1 = SMR

; ERROR 15      CACS IS NO LONGER .TRUE. WHEN 'SIAS' IS RESET TO 0.
;               RO = CIR        R1 = SMR

; ERROR 16      MC DOES NOT RESET CACS
;               RO = SMR

; ERROR 17      NO <ILLEGAL MESSAGE> TO <SICS> WHEN <SACS>
;               IS NOT SET.
;               RO = CIR

; ERROR 20      NO INTERRUPT OCCURED AFTER SENDING <SIC>.
;               (<ILLEGAL MESSAGE> SHOULD OCCUR AND BRING AN INTERRUPT.)
;               RO = CIR
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
; ERROR 39      NO <ILLEGAL MESSAGE> TO SETTING OF <GTS>
;               IN 'STAND BY' STATE.
;               R0 = CIR                R1 = SMR

; ERROR 40      <TCA> IN <CSBS> DOES NOT OR NOT ONLY PRODUCE <STATE
;               CHANGE>.  R0= CIR       R1 = SMR

; ERROR 41      <TCA> IN <CSBS> DOES NOT OR NOT ONLY  PRODUCE <CACS>.
;               R0 = CIR                R1 = SMR

; ERROR 42      WHEN ADDRESSING THE TALKER IN <CACS>, NO INTERRUPT
;               OCCURED.
;               THE INTERRUPT SHOULD HAVE BEEN SET BY <DATA ACCEPTED>.
;               R0 = 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>.
;               R0 = CIR        R1 = SMR        R2 = IOR

; ERROR 44      AFTER SUCCESSFULLY ADDRESSING THE TALKER, IT IS NOT
;               POSSIBLE TO SET <CSBS> AND <TACS> CORRECTLY BY <GTS>.
;               R0 = CIR        R1 = SMR        R2 = IOR

; ERROR 45      WHEN SENDING A DATA BYTE WITHOUT ADDRESSING THE LISTENER,
;               NO INTERRUPT OCCURED.
;               R0 = CIR        R1 = SMR        R2 = IOR

; ERROR 46      <TCS> DOES NOT PRODUCE <ILLEG. MESSAGE>
;               R0 = 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' = R0
;               BITS 8 15 CAN BE IGNORED BUT SHOULD BE ZERO.

; ERROR 50      WHEN SENDING THE LISTENER ADDRESS, NO INTERRUPT OCCURS.
;               R0 = CIR  R1 = SMR  R2 = IOR

; ERROR 51      SENDING THE LISTENER ADDRESS FORCES AN INTERRUPT THAT
;               WAS NOT OR NOT ONLY CAUSED BY <DATA ACCEPTED>.
;               R0-R2 SAME AS ERROR 50

; ERROR 52      AFTER LISTENER ADDRESSING AND <GTS> NO <LACS> IS SET.
;               R0 R2 SAME AS ERROR 50

; ERROR 53      NO INTERRUPT TO TALKER ADDRESSING
;               R0 R2 SEE ERROR 50

; ERROR 54      WHEN SENDING A SECOND DATA BYTE BEFORE READING THE FIRST
;               AGAIN <DATA ACCEPTED> CAME UP.
;               R0-R2 SEE ERROR 50
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

; ERROR 107     SEE ERROR 105
;
;

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

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

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

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

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

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

9.0 DIAGNOSTIC MONITOR DESCRIPTION

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

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.

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

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

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

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

WHEN AN ERROR OCCURS, THE FOLLOWING TEXT IS PRINTED:

```
ERROR   000XXX
AT LOCATION:    XXXXXX
...     HALT AFTER ERROR    ...
```

THE ERROR NUMBER CAN BE IN THE RANGE 0 [illegible].
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 DIGNOSTIC 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 OVER-
WRITTEN. 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 LOACTION 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
****

TABLE OF CONTENTS

```
                        .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.
;
;
```

```
                                        .PSECT  CSSMON
                                ;
                                ;       MZ001
                                ;       *****
                                ;       THIS CHANGE ALLOWS TO USE A MACRO THAT EITHER CLEARS
                                ;       THE PSW BY A "CLR" OR A "MTPS'.
                                ;
                                ;       GS001
                                ;       *****
                                ;       THIS CHANGE ALLOWS TO START THE PROGRAM AT LOC. 200
                                ;
                                ;
000000                                  .ASECT
                                        .=0
        000004                          .REPT   4
                                        .+2                     ;POINT TO VECTOR +2
                                        IOT                     ;HANDLE ILLEGAL TRAP
                                        .ENDR                   ;
000020  003056                          $..IOT                  ;ILLEGAL TRAP-HANDLER
000022  000000                          0                       ;PRIORITY = 0
000024  003126                          $..PWR                  ;POWERFAIL-RESTART-HANDLER
000026  000000                          0                       ;PRIORITY = 0
000030  002326                          $..EMT                  ;EMT-HANDLER
000032  000000                          0                       ;PRIORITY = 0
000034  002524                          $..TRP                  ;TAP-HANDLER
000036  000000                          0                       ;PRIORITY = 0
        000030                          .REPT   30              ;                       ;**GS001**
                                        .+2                     ;POINTER TO VECTOR +2   ;**GS001**
                                        IOT                     ;HANDLE ILLEGAL TRAP    ;**GS001**
                                        .ENDR                   ;                       ;**GS001**
000200  000137  001000                  JMP     10$             ;INITIALIZE SYSTEM      ;**GS001**
        000137                          .REPT   137             ;                       ;**GS001**
                                        .+2                     ;POINTER TO VECTOR +2   ;**GS001**
                                        IOT                     ;HANDLE ILLEGAL TRAP    ;**GS001**
                                        .ENDR                   ;                       ;**GS001**
001000  000137  001204          10$:    JMP     START           ; INITIALIZE SYSTEM
        000100                          .REPT   100
                                        0                       ;STACK AREA
                                        .ENDR
```

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

```
                                        .SBTTL  MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
                                ;
                                ;  SUBROUTINE:  M..TST
                                ;  ***********  ******
                                ;
                                ;THIS ROUTINE CHECKS THE AVAILABILITY OF A MEMORY MANAGEMENT UNIT
                                ;IN PDP 11 PROCESSORS AND SETS KISAR0...KISAR5 TO THE LOW 24K-WORDS
                                ;OF THE COMPUTER MEMORY .
                                ;KISAR7 IS USED TO MAP THE SO-CALLED I/O PAGE .
                                ;KISAR6 IS USED TO MAP THE HIGHEST MEMORY ADDRESS CHECK POINTER .
                                ;
                                ;INPUT:         (SP)    *RETURN ADDRESS
                                ;
                                ;OUTPUT:        R0      *LOW 16 ADDRESS BITS OF HIGHEST MEMORY ADDRESS
                                ;               R1      *BITS 0,1 ADDRESS BITS 16,17
                                ;
                                ;AUTHOR:        BERNHARD BAUDISCH CSS/DP MUNICH 19 DEC 78
                                ;
                                ;
001516                          M..TST:                         ;REFERENCE LABEL
001516  012737  001776  000250          MOV     #M..TRP,@#250   ;SETUP SEGMENTATION TRAP CATCHER
001524  005037  000252                  CLR     @#252           ;...
001530  012737  001776  000004          MOV     #M..TRP,@#4     ;SETUP TRAP-4 CATCHER
001536  005037  000006                  CLR     @#6             ;...
001542  004737  002006                  CALL    M..MMM          ;SETUP DEFAULT MAPPING,ENABLE MM
001546  012737  007400  172354          MOV     #7400,KISAR6    ;MAP WITH KISAR6 THROUGH MEMORY
001554  012737  000001  007020          MOV     #1,$$MTYP       ;REMEMBER THAT WE HAVE MEMORY MANAGEMENT!
001562  012700  157776                  MOV     #157776,R0      ;GET HIGHEST ADDRESS WITHIN APR6 RANGE
001566  012737  001766  000004          MOV     #M..HIM,@#4     ;SET UP TRAP 4 CATCHER
001574  005710                  20$:    TST     @R0             ;SEE IF ADDRESS PRESENT
001576  103031                          BCC     M..DET          ;CC IF ADDRESS FOUND
001600  162737  000200  172354          SUB     #200,KISAR6     ;MAP TO NEXT LOWER 4K WORD BANK
001606  002401                          BLT     30$             ;THIS CASE SHOULD NEVER HAPPEN
001610  000771                          BR      20$             ;CHECK NEXT LOWER 4-K WORDS
001612                          30$:                            ;HERE ONLY IF MAIN PROCESSOR HARDWARE ERROR
001612                                  PSET    #0              ;SET PRIORITY TO 0
001636                                  WRITE   MEMDEF          ;MEMORY MANAGEMENT DEFECT
001646  012716  001374                  MOV     #M..TSE,(SP)    ;MODIFY RETURN ADDRESS TO UNMAPPED SYSTEM
001652  042737  000001  177572          BIC     #BIT0,SR0       ;DISABLE MM
001660  000422                          BR      M..RET          ;AND GO AHEAD LIKE UNMAPPED SYSTEM
001662                          M..DET:                         ;HIGHEST MEMORY ADDRESS FOUND
001662                                  PUSH    KISAR6          ;GET MAP REGISTER CONTENTS
        000005                          .REPT   5               ;
                                        ASL     (SP)            ;PUT ADDRESS BIT 17 INTO PSW CARRY
                                        .ENDR                   ;
001700  103402                          BCS     40$             ;CS IF ADDRESS BIT 17 SET
001702  005001                          CLR     R1              ;HERE IF BIT17 NOT SET
001704  000402                          BR      45$             ;...
001706  012701  000001          40$:    MOV     #BIT0,R1        ;FLAG BIT 17 FOUND
001712  006316                  45$:    ASL     (SP)            ;GET ADDRESS BIT 16
001714  006101                          ROL     R1              ;INSERT BIT IN MEMORY EXTENSION MASK
001716  042700  160000                  BIC     #160000,R0      ;MASK OUT APR SELECTION BITS
001722  050016                          BIS     R0,(SP)         ;BUILD 16 BIT BASE ADDRESS
001724  012600                          MOV     (SP)+,R0        ;-->R0 FOR OUTPUT TO CALLER
001726  012737  000252  000250  M..RET: MOV     #252,@#250      ;RESET SEGMENTATION TRAP CATCHER
001734  012737  000004  000252          MOV     #4,@#252        ;...
001742  012737  000006  000004          MOV     #6,@#4          ;RESET TRAP-4 CATCHER
```

```
                                                MOV     #4,@#6          ;...
                                                RTS     PC              ;RETURN TO CALLER
                                        U..DET:                         ;HERE IF MEMORY MANAGEMENT ADDRESS NOT FOUND
                                                MOV     #M..TSF,(SP)    ;MODIFY RETURN ADDRESS
                                                BR      M..RET          ;RETURN COMMON
                                        M..HIM:                         ;TRAP-4 AND SEGMENTATION TRAP CATCHER
                                                BIS     #BIT0,2(SP)     ;SET CARRY IN RETURN PSW SETUP
                                                RTI                     ;RETURN WITH CARRY SET INDICATOR
                                        ;
                                        M..TRP:
                                                CMP     (SP)+,(SP)+     ;CLEAN STACK OF INT. VALUES
                                                TST     (SP)+           ;CLEAN STACK FROM RETURN ADDRESS
                                                JMP     U..DET          ;NO MEMORY MANAGEMENT UNIT AVAILABLE
                                        ;
                                        ;
                                        ;
                                        ;       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
                                        ;               APR0 IS PRESET WITH 0
                                        ;               APR1    "           200
                                        ;               APR2    "           400
                                        ;                .                   .
                                        ;                .                   .
                                        ;               APR7    "           7600
                                        ;
                                        ;               MM IS ENABLED
                                        ;
                                        ;
                                        M..MMM:
                                                PUSH    R0,R1,R2        ;SAVE
                                                MOV     #77406,R0       ;GET PAGE DESCRIPTOR REGISTER SETUP
                                                MOV     #8.,R1          ;SETUP FOR 8 PDR'S
                                                MOV     #KISDR0,R2      ;GET KISDR0 ADDRESS
                                        10$:    MOV     R0,(R2)+        ;SETUP NEXT PDR
                                                DEC     R1              ;COUNT SETUPS
                                                BNE     10$             ;NE 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,SR0       ;ENABLE MEMORY MANAGEMENT
                                                POP     R2,R1,R0        ;RESTORE REGS.
                                                RETURN                  ;BACK TO CALLER
                                                .SBTTL  $..MAP MAPPING ROUTINE
```

```
                                ;
                                ; 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 COM (0 4.K)
                                ;
                                ;       AUTHOR: ALBERT BREM CSS MUC 16-JAN-79
                                ;
                                ;
002116                          $..MAP: PUSH    R0,R1           ;SAVE REGISTERS
002122                                  IF 6(SP) GE #200 GOTO 10$;DONT MAP TO SYSTEM AREA
002136  000003                         BPT                     ;DONT MAP TO SYSTEM AREA
002140  000137  002316                 JMP     2$              ;LEAVE
002144  004737  002006          10$:   CALL    M..MMM          ;ENABLE MM,SETUP DEFAULT MAPPING
002150  013700  007022                 MOV     MEMEND,R0       ;SAVE HIGHEST MEMORY LOCATION
        000006                          .REPT   6
                                        ROR     R0              ;COMPUTE 100 OCTAL BYTE BLOCKS
                                        .ENDR
002170  042700  176000                 BIC     #176000,R0      ;"
002174  013701  007024                 MOV     MEMEND+2,R1     ;SAVE MEMORY EXTENSION
        000012                          .REPT   10.
                                        ASL     R1              ;COMPUTE ADDITIONAL BLOCKS
                                        .ENDR
002224  060001                         ADD     R0,R1           ;TOTAL BLOCKS OF 32.WORDS MEMORY
002226  042737  000001  177776         BIC     #BIT0,@#PS      ;CLEAR CARRY
002234  020166  000006                 CMP     R1,6(SP)        ;IS MEMORY AVAILABLE?
002240  002023                         BGE     1$              ;YES
002242                                  PREAD   R0              ;SAVE CURRENT PSW
002262  052700  000001                 BIS     #BIT0,R0        ;SET CARRY FOR ERROR INDICATION
002266                                  PSET    R0              ;'
002306  000403                         BR      2$              ;DONT TOUCH APR6
002310  016637  000006  172354  1$:    MOV     6(SP),KISAR6    ;MAP TO BUFFER AREA
002316                          2$:    POP     R1,R0           ;RESTORE REGISTERS
002322  012616                         MOV     (SP)+,(SP)      ;CLEAN STACK OF ARGUMENT
002324  000207                         RETURN                  ;BACK TO CALLER
                                ;
                                ;
                                ;
```

```
                                        .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
002326  012746  177776          $..EMT: MOV     #-2,-(SP)       ;REBUILD EMT-ADDR.
002332  066616  000002                  ADD     2(SP),(SP)      ;       "
002336  017616  000000                  MOV     @(SP),(SP)      ;GET EMT-CODE
002342  042716  177400                  BIC     #177400,(SP)    ;EXTRACT EMT-NR.
002346  006316                          ASL     (SP)            ;CONVERT TO TABLE-OFFSET
002350  021627  000004                  CMP     (SP),#EMTTND-EMTTBL 2;VALID EMT-NR.?
002354  101413                          BLOS    1$              ;IF LOS, YES
002356                                  WRITE MILEMT            ;ILLEGAL EMT
002366                                  DUMP    OCT,2(SP)       ;
002400  000137  001000                  JMP     @#1000          ;RESTART PROGRAM
002404  062716  002416          1$:     ADD     #EMTTBL,(SP)    ;GET FUCTION CODE ADDR.
002410  017616  000000                  MOV     @(SP),(SP)      ;       "
002414  000136                          JMP     @(SP)+          ;ENTER FUCTION SUBROUTINE
002416  002424                  EMTTBL: EMTO
002420  002432                          EMT1
002422  002470                          EMT2
002424                          EMTTND:
                                ;
                                ;SUBROUTINE TO SAVE SCOPE-LOOP ADDR.
                                ;
002424  011637  007172          EMTO:   MOV     (SP),SCOPAD     ;SAVE ADDR. OF SCOPE-LOOP
002430  000002                          RTI                     ;RETURN TO CALLER
                                ;
                                ;SUBROUTINE TO SETUP A VECTOR
                                ;
002432  017646  000010          EMT1:   MOV     @10(SP),-(SP) ;GET POINTER TO VECTOR ADDR.
002436  016676  000010  000000          MOV     10(SP),@(SP)    ;SETUP VECTOR
002444  062716  000002                  ADD     #2,(SP)         ;POINT TO NEW PS
002450  016636  000006                  MOV     6(SP),@(SP)+    ;SETUP NEW PS
002454  012666  000004                  MOV     (SP)+,4(SP)     ;CLEAN UP STACK
002460  012666  000004                  MOV     (SP)+,4(SP)     ;       "
002464  005726                          TST     (SP)+           ;       "
002466  000002                          RTI                     ;RETURN TO CALLER
                                ;
                                ; SUBROUTINE TO RESET A VECTOR
                                ;
002470  017646  000004          EMT2:   MOV     @4(SP),-(SP)    ;GET POINTER TO VECTOR A
002474  011646                          MOV     (SP),-(SP)      ;DUPLICATE IT
002476  062716  000002                  ADD     #2,(SP)         ;BUILD ADDRESS OF VECTOR
002502  012636                          MOV     (SP)+,@(SP)+    ;RESET VECTOR
```

L2


```
    324 002504  012776  000004  177774          MOV     #4,@-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
```

```
                                        .SBTTL  $..TRP
                                ;*
                                ; SUBROUTINE:   $..TRP
                                ; ***********   ******
                                ;
                                ; THIS ROUTINE HANDLES ALL TRAPS TO LOC 34 THAT MIGHT
                                ; OCCUR IN A CSS-DIGNOSTIC PROGRAM.
                                ; AS THE TRAP INSTRUCTION IN THIS DIAGNOSTICS IS ONLY USED TO
                                ; INDICATE AN ERROR CONDITION, THIS TRAP-HANDLER IS A (PSEUDO-)
                                ; SWITCH-REGISTER CONTROLLED ERROR HANDLER. FOR THE DEFINITION
                                ; OF ALL AVAILABLE 'SWITCH'-OPTIONS, REFER TO SUBROUTINE "$..IAE"
                                ; LATER IN THIS LISTING.
                                ;
                                ; INPUT:       2(SP)   =       PS
                                ;               (SP)   =       PC
                                ;   LOW ORDER BYTE OF TRAP INSTRUCTION = ERROR NUMBER
                                ;
                                ; OUTPUT:      DEPENDING ON OPTIONS SELECTED
                                ; AUTHOR:      BERT HUBER, CSS/DP MUNICH   14 JUL-76
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
002524  012746  177776         $..TRP: MOV     #-2,-(SP)       ;REBUILD TRAP ADDRESS
002530  066616  000002                 ADD     2(SP),(SP)      ;       "       "
002534  017616  000000                 MOV     @(SP),(SP)      ;GET TRAP CODE
002540  042716  177400                 BIC     #177400,(SP)    ;EXTRACT ERROR NUMBER
002544  005737  007166                 TST     $..ER           ;FIRST ERROR OCCURENCE ?
002550  001421                         BEQ     2$              ;IF EQ, FIRST PASS
002552  005737  007160                 TST     PSDSWR          ;SCOPE LOOP ?
002556  100020                         BPL     3$              ;
002560  021637  007166                 CMP     (SP),$..ER      ;SAME SCOPE LOOP ?
002564  001415                         BEQ     3$              ;IF EQ, YES
002566                                 WRITE   MILSCP          ;ILLEGAL SCOPE LOOP
002576                                 DUMP    OCT,2(SP)       ;DUMP ADDRESS OF ERROR +2
002610                                 GOTO    START           ;RESTART PROGRAM
002614  011637  007166         2$:     MOV     (SP),$..ER      ;STORE ERROR NUMBER
002620  032737  040000  007160 3$:     BIT     #BIT14,PSDSWR   ;INHIBIT PRINTOUT ?
002626  001022                         BNE     4$              ;IF NE, YES
002630                                 WRITE   MERROR          ;PRINT ERROR MESSAGE PART 1
002640                                 DUMP    DEC,(SP)        ;DUMP ERROR NUMBER
002652                                 WRITE   MERRP2          ;PRINT ERROR MESSAGE PART 2
002662                                 DUMP    OCT,2(SP)       ;DUMP ERROR PC +2
002674  032737  020000  007160 4$:     BIT     #BIT13,PSDSWR   ;COUNT ERRORS ?
002702  001414                         BEQ     5$              ;IF EQ, NO
002704  005276  000002                 INC     @2(SP)          ;COUNT !
002710  001011                         BNE     5$              ;IF EQ, OVERFLOW
002712                                 WRITE   MERCOV          ;ERROR COUNTER OVERFLOW
002722                                 DUMP    DEC,(SP)        ;DUMP ERROR NUMBER
002734  032737  040000  007160 5$:     BIT     #BIT14,PSDSWR   ;PRINT ? ?
002742  001014                         BNE     6$              ;IF NE, NO
002744  032737  100000  007160         BIT     #BIT15,PSDSWR   ;SCOPE LOOP ?
002752  001010                         BNE     6$              ;IF SCOPE, NO PRINT
002754  032737  010000  007160         BIT     #BIT12,PSDSWR   ;HALT AFTER ERROR SELECTED
002762  001004                         BNE     6$              ;IF NE, YES
002764                                 WRITE   MERHLT          ;STOP MESSAGE
002774  032737  010000  007160 6$:     BIT     #BIT12,PSDSWR   ;HALT ?
003002  001004                         BNE     7$              ;IF EQ, YES
```

```
    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
```

```
                                        .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
003056                          $..IOT: WRITE   MILTR1          ;SEND FIRST MESSAGE PART
003066  162716  000002                  SUB     #2,(SP)         ;COMPUTE VECTOR
003072                                  DUMP    OCT,(SP)        ;DUMP VECTOR ADDRESS
003102                                  WRI     MILTR2          ;SEND SECOND MESSAGE PART
003112                                  DUMP    OCT,4(SP)       ;DUMP PROGRAM PC
003124  000777                          BR      .               ;HANG UP SYSTEM
                                        .DSABL  LSB
```

```
                                        .SBTTL  $..PWR
                                ;
                                ;SUBROUTINE:    $..PWR
                                ;***********    ******
                                ;
                                ;THIS ROUTINE IS DIRECTLY ENTERED AFTER EACH POWER-FAIL TO
                                ;HANDLE THE NECESSARY REGISTER SAVE OPERATIONS, AND IS USED
                                ;AFTER THE POWER-UP INTERRUPT TO RESTORE THE SAVED REGISTERS
                                ;AFTER A SUCCESFULL POWER-FAIL RESTART, A SHORT MESSAGEW IS PRIN'ED.
                                ;
                                ;AUTHOR:         BERT HUBER, CSS/DP MUNICH 27-JULY-76
                                ;       MZ001    SAVE THE OUTFLAG IN 3$ AND RESTORE IT
                                ;       *****    AFTER PRINTOUT OF THE POWERFAIL MESSAGE
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
003126                          $..PWR: PUSH    R0,R1,R2,R3,R4,R5
003142  010627                          MOV     SP,(PC)+        ;SAVE STACK-POINTER
003144  000000                  1$:     .WORD   0               ;STACKPOINTER-SAVE-AREA
003146  012737  003172  000024          MOV     #2$,@#24        ;SETUP VECTOR FOR RESTART
003154  013727  007152                  MOV     OUTFLG,(PC)+    ;SAVE PRINT FLAG        ;MZ001
003160  000000                  3$:     .WORD   0               ;SAVE FOR FLAG  ;MZ001
003162  013727  007150                  MOV     INFLAG,(PC)+    ;SAVE INPUT FLAG
003166  000000                  5$:     .WORD   0               ;SAVE WORD FOR INPUT FLAG
003170  000777                          BR      .               ;WAIT FOR POWER DOWN
003172  005002                  2$:     CLR     R2              ;DELAI
003174  005202                          INC     R2              ;THE
003176  020227  177777                  CMP     R2,#177777      ;POWER UP START
003202  001374                          BNE     .-6             ;FOR 1SECOND
003204  013706  003144                  MOV     1$,SP           ;REBUILD STACKPOINTER
003210  005037  007152                  CLR     OUTFLG          ;ALLOW PRINTOUT OF PWRFL MES.   ;MZ001
003214  005037  007150                  CLR     INFLAG          ;               "
003220  012737  003126  000024          MOV     #$..PWR,@#24    ;REBUILD VECTOR SETUP
003226                                  POP     R5,R4,R3,R2,R1,R0
003242  013727  005700                  MOV     $..MSP,(PC)+    ;SAVE OLD MESSAGE POINTER
003246  000000                  8$:     .WORD   0               ;SAVE WORD FOR OLD MESSAGE POINTER
003250                                  WRITE   MPWRFL          ;SEND POWER-FAIL-MESSAGE        ;MZ001
003260  005737  007152          4$:     TST     OUTFLG          ;WAIT FOR OUTPUT COMPLETE       ;MZ001
003264  001375                          BNE     4$              ;       ;MZ001
003266                                  SET6    @#TKS           ;ENABLE FURTHER TTY-INTERRUPTS
003274  013737  003160  007152          MOV     3$,OUTFLG       ;RESTORE FLAG IF IT WAS SET     ;MZ001
003302  005737  007152                  TST     OUTFLG          ;IF AN OUTPUT BEFORE PWRFL WAS RUNNING
003306  001411                          BEQ     7$              ;
003310  013737  003246  005700          MOV     8$,$..MSP       ;THEN RESTORE OLD MESSAGE POINTER
003316                                  SET6    @#TPS           ;AND ENABLE INT IN TPS
003324  112737  000012  177566          MOVB    #12,@#TPB       ;AND START INTERRUPT
003332  005737  003166          7$:     TST     5$              ;IF AN INP FLG BEFORE PWRFL WAS NOT SET
003336  001407                          BEQ     6$              ;GOTO 6$
003340                                  WRITE   PROMES          ;ELSE WRITE ,
003350  013737  003166  007150          MOV     5$,INFLAG       ;RESTORE INPUT FLAG IF IT WAS SET
003356  000002                  6$:     RTI                     ;CON'INUE WITH PROGRAM
                                        .DSABL  LSB
                                        .SBTTL  $..RSV
                                ;*
                                ; SUBROUTINE:    $..RSV
                                ; ***********    ******
                                ;
```

```
                                ; THIS ROUTINE SAVES R0 R5 ONTO THE STACK
                                ;
                                ;
                                ; INPUT:          (SP)     -       CALLER PC
                                ;
                                ; OUTPUT:          (SP)    -       R5 CONTENT
                                ;                 2(SP)    -       R4 CONTENT
                                ;                 4(SP)    -       R3 CONTENT
                                ;                 6(SP)    -       R2 CONTENT
                                ;                10(SP)    -       R1 CONTENT
                                ;                12(SP)    -       R0 CONTENT
                                ;                14(SP)    -       CALLER PC
                                ;
                                ; AUTHOR:         BERT HUBER, CSS/DP MUNICH   12 APR 76
                                ;
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
003360                          $..RSV: PUSH    R0,R1,R2,R3,R4,R5
003374  000176  000014                  JMP     @14(SP)         ;RETURN TO CALLER
                                        .DSABL  LSB
```

```
$..RRS
.

                                        .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
003400  005726                  $..RRS: TST     (SP)+           ;POINT TO SAVED R5
003402                                  POP     R5,R4,R3,R2,R1,R0
003416  005726                          TST     (SP)+           ;SKIP OLD CALLER PC
003420  000176  177760                  JMP     @ 20(SP)        ;RETURN TO CALLER
                                        .DSABL  LSB
```

```
$..KBI

                                        .SBTTL  $..KBI
                                ;+
                                ; SUBROUTINE:   $..KBI
                                ; ***********   ******
                                ;
                                ; THIS ROUTINE IS ENTERED IMMEDIATELY AFTER AN INTERRUPT
                                ; FROM TTY-KEYBOARD. ALL REGISTERS ARE SAVED ACROSS CALL.
                                ;
                                ; INPUT:        2(SP)=  PS
                                ;                (SP)   =       PC
                                ;
                                ; OUTPUT:       NONE
                                ;
                                ; AUTHOR:       BERT HUBER, CSS/DP MUNICH  5-APR 76
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
003424  005737  007152          $..KBI: TST     OUTFLG          ;;;OUTPUT RUNNING ?
003430  001422                          BEQ     4$              ;;;IF NE, IGNORE INTERRUPT
003432                                  PUSH    R0              ;;SAVE
003434  005037  177560                  CLR     @#TKS           ;;;INHIBIT INTERRUPT
003440  013700  177562                  MOV     @#TKB,R0        ;;;GET CHARACTER
003444  012737  000100  177560          MOV     #100,@#TKS      ;;RE-ENABLE
003452  042700  000200                  BIC     #200,R0         ;;;MASK OUT ASCII
003456  022700  000003                  CMP     #3,R0           ;;;  IS IT <CNTRL C> ?
003462  001003                          BNE     13$             ;;;IF NOT, RETURN
003464  012737  007777  007170          MOV     #7777,SPEFLG    ;;;SET A SPECIAL FLAG
                                                                ;;;THIS FLAG IS USED TO FINISH
                                                                ;;;A RUNNING PRINTOUT AND THEN TO
                                                                ;;;REMEMBER THAT <CNTRL C> WAS HIT
003472                          13$:    POP     R0              ;;;REBUILD
003474  000436                          BR      3$              ;;;  RETURN TO CALLER
003476  004737  003360          4$:     CALL    $..RSV          ;;;SAVE ALL REGISTERS
003502  013700  177562                  MOV     @#TKB,R0        ;;;GET INPUT BYTE
003506  042700  000200                  BIC     #200,R0         ;;;BUILD REAL ASCII
003512  122700  000003                  CMPB    #3,R0           ;;;<CNTRL C> FUNCTION ?
003516  001003                          BNE     1$              ;;;IF NE, NO
003520                          $$SPE:
003520  004737  006136                  CALL    $..PRO          ;;;SEND PROMPT STRING
003524  000420                          BR      2$              ;;;RETURN
003526  005737  007150          1$:     TST     INFLAG          ;;;INPUT FLAG SET ?
003532  001415                          BEQ     2$              ;;;IF EQ, NO
003534  004737  003574          5$:     CALL    $..INP          ;;;PROCESS INPUT
003540  020027  000015                  CMP     R0,#15          ;;;WAS IT A <CR> ?
003544  001010                          BNE     2$              ;;;IF NE, NO
003546  005037  007150                  CLR     INFLAG          ;;;RESET INPUT MODE FLAG
003552  012701  007026                  MOV     #KBBUFF,R1      ;;;REBUILD BUFFER POINTER
003556  010137  007146                  MOV     R1,KBBPNT       ;;;     "       "       "
003562  004737  004010                  CALL    $..IAY          ;;;ANALYZE INPUT
003566  004737  003400          2$:     CALL    $..RRS          ;;;RESTORE REGISTERS
003572  000002                  3$:     RTI                     ;;;RETURN TO CALLER
                                        .DSABL  LSB
```

```
                                        .SBTTL  $..INP
                                ;+
                                ;SUBROUTINE:    $..INP
                                ;***********    ******
                                ;
                                ;THIS SUBROUTINE IS THE INPUT HANDLER FOR ALL TERMINALS
                                ;USED IN ALL CSS-MUNICH DIAGNOSTIC SOFTWARE GENERATED NOW
                                ;AND LATER.
                                ;THE SPECIAL FUCTIONS HANDLED BY THIS ROUTINE ARE:
                                ;       <CNTRL U> FOR LINE RUBOUT
                                ;       <DELETE> FOR CHARACTER RUBOUT
                                ;ALL OTHER INPUT IS HANDLED AS NORMAL ASCI-TEXT AND STORED
                                ;IN THE KEYBOARD INPUT BUFFER.
                                ;
                                ;INPUT:         (SP)    -       CALLER PC
                                ;               R0      -       CURRENT INPUT CHARACTER
                                ;
                                ;OUTPUT:        NONE
                                ;
                                ;AUTHOR:        BERT HUBER CSS/DP MUNICH
                                ;-
                                        .IDENT  /V1.0/
                                   ;    .PSECT  CSSMON
                                        .ENABL  LSB
003574  013701  007146          $..INP: MOV     @KBBPNT,R1              ;;;GET CURRENT BUFFER POINTER
003600  012702  006122                  MOV     #$..KBO,R2      ;;;POINT TO KEYBOARD-ECHO
003604  022701  007146                  CMP     #KBBEND,R1      ;;;INPUT-BUFFER FULL?
003610  001015                          BNE     1$              ;;;IF NE, NO
003612  022700  000025                  CMP     #25,R0          ;;;<CNTRL U>?
003616  001415                          BEQ     2$              ;;;IF EQ, YES.
003620  122700  000177                  CMPB    #177,R0         ;;;<DELETE>?
003624  001407                          BEQ     1$              ;;;IF EQ, YES.
003626  005037  007150                  CLR     INFLAG          ;;;RESET INPUTFLAG
003632                                  WRITE   #KBOVF          ;;;PRINT OVERFLOW MESS.
003642  000451                          BR      7$              ;;;PROMPT FOR INPUT
003644  122700  000025          1$:     CMPB    #25,R0          ;;;<CNTL U>?
003650  001436                          BEQ     6$              ;;;IF NE, NO
003652  122700  000177          2$:     CMPB    #177,R0         ;;;<DELETE>?
003656  001016                          BNE     4$              ;;;IF NE, NO
003660  005737  007162                  TST     RUBFLG          ;;;FIRST RUBOUT?
003664  001005                          BNE     3$              ;;;IF NE, NO
003666  010637  007162                  MOV     SP,RUBFLG       ;;;FLAG RUBOUT MODE
003672  012700  000057                  MOV     #'/,R0          ;;;PRINT "/"
003676  004712                          CALL    @R2             ;;;     "
003700  020127  007026          3$:     CMP     R1,#KBBUFF      ;;;FULL LINE DELETED?
003704  001434                          BEQ     INP..E          ;;;IF EQ, YES
003706  114100                          MOVB    (R1),R0                 ;;;NO, REMOVE CHARACTER
003710  004712                          CALL    @R2             ;;;AND ECHO IT
003712  000431                          BR      INP..E          ;;;WAIT FOR NEXT INP !T
003714  005737  007162          4$:     TST     RUBFLG          ;;;RUBOUT MODE?
003720  001407                          BEQ     5$              ;;;IF EQ, NO
003722                                  PUSH    R0              ;;;YES, SAVE CURRENT INPUT
003724  012700  000057                  MOV     #'/,R0          ;;;NO OF RUBOUT
003730  004712                          CALL    @R2             ;;;ECHO "/"
003732  005037  007162                  CLR     RUBFLG          ;;;RESET RUBOUT MODE FLAG
003736                                  POP     R0              ;;;GET CURRENT INPUT AGAIN
003740  004712                  5$:     CALL    @R2             ;;;ECHO IT
```

```
IEC11 A DIAGNOSTIC      MACRO M1200  30-MAR-84 16:02  PAGE 15 1                                  SEQ 33
$..INP

    647 003742  110021                          MOVB    R0,(R1)+        ;;;SAVE IN IPUT BUFFER
    648 003744  000414                          BR      INP..E          ;;;WAIT FOR NEXT INPUT
    649 003746  012700  000136          6$:     MOV     #136,R0         ;;;ECHO "↑"
    650 003752  004712                          CALL    @R2             ;;;
    651 003754  012700  000125                  MOV     #'U,R0          ;;;ECHO "U"
    652 003760  004712                          CALL    @R2             ;;;     "
    653 003762  005037  007162                  CLR     RUBFLG          ;;;LEAVE RUBOUT MODE
    654 003766  012701  007026          7$:     MOV     #KBBUFF,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),R0       ;;;SAVE LAST CHARACTER
    658 004006  000207                          RETURN                  ;;;RETURN TO CALLER
    659                                         .DSABL  LSB
```

```
                                .SBTTL  $..IAY
                        ;+
                        ;SUBROUTINE:    $..IAY
                        ;***********    ******
                        ;
                        ;LEGAL INPUT TYPES ARE:
                        ;
                        ;       ERR=            FOR ERROR-HANDLING
                        ;       DMP=            FOR CORE-DUMP OPTIONS
                        ;       SET=            FOR DEBUGGING OPTIONS
                        ;       NONFORMATTED INPUT FOR SPECIAL REQ.
                        ;
                        ;INPUT: (SP)=   RETURN PC
                        ;OUTPUT:        NONE
                        ;
                        ;AUTHOR:        BERT HUBER, CSS/DP MUNICH       28-JUN 76
                        ;-
                                ;.PSECT CSSMON
                                .ENABL  LSB
004010  122711  000015  $..IAY: CMPB    #15,(R1)        ;;;<CR>?
004014  001440                  BEQ     IAY..E          ;;;IF EQ., SKIP ANALIZATION
004016  005737  007174  3$:     TST     $.$ASF          ;;;DONT ANALYZE IF ASCII INPUT RUNNING
004022  001035                  BNE     IAY..E          ;;;IF NE, YES.
004024  122711  000104          CMPB    #'D,(R1)        ;;;'DMP" REQUESTED?
004030  001003                  BNE     1$              ;;;IF NE, NO
004032  004737  004210          CALL    $..IAD          ;;;ANALIZE DUMP REQUEST
004036  000427                  BR      IAY..E          ;;;RETURN TO CALLER
004040  122711  000123  1$:     CMPB    #'S,@R1         ;;;'SET" REQUESTED?
004044  001003                  BNE     2$              ;;IF NE, NO
004046  004737  004402          CALL    $..IAS          ;;;ANALYZE SET REQUEST
004052  000421                  BR      IAY..E          ;;;RETURN TO CALLER
004054  122711  000105  2$:     CMPB    #'E,@R1         ;;;"ERR"-REQUEST?
004060  001003                  BNE     4$              ;;;IF NE, NO
004062  004737  004472          CALL    $..IAE          ;;;ANALYZE ERROR REQUEST
004066  000413                  BR      IAY..E          ;;;RETURN TO CALLER
004070  122711  000101  4$:     CMPB    #'A,@R1         ;;;"ABO" REQUESTED ?
004074  001003                  BNE     5$              ;;;IF NE, NO
004076  004737  004120          CALL    $..IAA          ;;;ANALYZE ABO-REQUEST
004102  000405                  BR      IAY..E          ;;;RETURN TO CALLER
004104  005737  007156  5$:     TST     INPREQ          ;;;NO SYNTAX ERROR IF SPECIAL INPUT RUNNING
004110  001002                  BNE     IAY..E          ;;;
004112  004737  006262          CALL    $..STX          ;;;SYNTAX ERROR
004116  000207          IAY..E: RETURN                  ;;;RETURN TO CALLER
                                .DSABL  LSB
```

```
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                        .SBTTL  $..IAA
                                ;+
                                ;SUBROUTINE:    $..IAA
                                ;***********    ******
                                ;
                                ;LEGAL INPUT IS "ABO"
                                ;
                                ;       INPUT:  NONE
                                ;       OUTPUT: IF IN THE PROGRM THE LABEL ""$..SPC"" IS USED,
                                ;               THE STACK WILL BE INITIALIZED AND A JUMP TO
                                ;               THIS LABEL IN THE DEVICE TEST PROGRAM IS PERFORMED.
                                ;               IF THIS LABEL IS NOT USED, THE STACK IS INITIALIZED
                                ;               AND A JUMP TO THE BEGIN OF THE DEVICE TEST PROGRAM
                                ;               IS PERFORMED.
                                ;       MZ001
                                ;       *****
                                ;       THE MACRO "PSET"ALLOWS TO RUN THE CDM EITHER ON PDP
                                ;       OR ON LSI.
                                ;
                                ;
                                ;AUTHOR:        M. ZILLER,      CSS/DP MUNICH   OCT. 1976
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
004120  000005                  $..IAA: RESET                   ;STOP ANY ACTIONS
004122                          SET6    @@TKS                   ;SWITCH ON TTY INT.
004130  012706  001204          MOV     @START,SP               ;INITIALIZE STACK
                                .IF DF $..SPC
004134  012746  011342          MOV     @$..SPC,-(SP)           ;SET THE LABEL ''$..SPC"" FOR DESTINATION
                                .ENDC
                                .IF NDF $..SPC
                                MOV     @CDM..E, (SP)           ;SET END OF MONITOR FOR DESTINATION
                                .ENDC
004140                          PSET    @0                      ;CLEAR PS       ;MZ001
004164  012700  007026          MOV     @KBBUFF,R0              ;RESET BUFFERS
004170  005020                  1$:     CLR     (R0)+           ;       "
004172  020027  007202          CMP     R0,@PROMES              ;       "
004176  001374                  BNE     1$                      ;       "
004200  012737  007026  007146  MOV     @KBBUFF,KBBPNT          ;REBUILD BEGIN ADDRESS
004206  000207                  RETURN                          ;GOTO POINT OF CONTINUATION
                                .DSABL  LSB
```

```
                        ;.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
004210  005201                $..IAD: INC     R1              ;POINT TO "M"
004212  122127  000115                CMPB    (R1)+,#'M       ;IS IT A 'M'
004216  001027                BNE     77$             ;NO IF NE
004220  122127  000120                CMPB    (R1)+,# P       ;IS IT A 'P'
004224  001024                BNE     77$             ;NO IF NE
004226  122127  000057                CMPB    (R1)+,# /       ;IS IT A '/"
004232  001021                BNE     77$             ;NO IF NE
004234  004737  004304                CALL    $..NOR          ;RETRIEFE INPUT
004240  103416                BCS     77$             ;IS CS, INPUT FORMAT ERROR
004242  105737  177564          11$:  TSTB    @#TPS           ;IS THE <CR> FINISHED ? ;MZ001
004246  100375                BPL     11$             ;                       ;MZ001
004250  012737  000012  177566        MOV     #12,@#TPB       ;PRINT A <LF
004256  105737  177564          12$:  TSTB    @#TPS           ;DONE ?
004262  100375                BPL     12$             ;WAIT FOR COMLETION
004264                        JMP     OCT,(R0)                ;AND PRINT
004274  000402                BR      IAD..E          ;GO TO END
004276  004737  006262          77$:  CALL    $..STX          ;WRITE SYNTAX ERROR MESSAGE
004302  000207                IAD..E: RETURN                  ;RETURN TO MAIN
                ;
```

```
                                        ;*
                                        ;       THIS IS THE ROUTINE GET THE OCTAL VALUE OF THE ADDRESS
                                        ;
                                        ;
004304  005000          $..NOR: CLR     R0              ;CLEAR STORAGE OF NUMBER
004306  012702  000006          MOV     #6,R2           ;CLEAR DIGIT COUNTER
004312  121127  000015  1$:     CMPB    @R1,#15         ;IS IT A TERMINATOR ?
004316  001425                  BEQ     NOR..E          ;IF EQ, YES
004320  121127  000057          CMPB    @R1,#'/         ;
004324  001422                  BEQ     NOR..E          ;
004326  142711  000060          BICB    #60,@R1         ;BUILD REAL BINARY
004332  100421                  BMI     NOR.ER          ;IF MI, IT WAS LOWER THAN 60
004334  121127  000007          CMPB    @R1,#7          ;IS IT REAL OCTAL ?
004340  101016                  BHI     NOR.ER          ;IF HI,NO REAL OCTAL
004342  006300                  ASL     R0              ;MULTIPLY
004344  006300                  ASL     R0              ;WITH 10 IN
004346  006300                  ASL     R0              ;OCTAL
004350  152100                  BISB    (R1)+,R0        ;SET THE TRUE BITS
004352  005302                  DEC     R2              ;COUNT DIGITS BUILT
004354  001356                  BNE     1$              ;ALL DONE?
004356  121127  000015          CMPB    @R1,#15         ;IS THE LAST + 1 BYTE A TERMINATOR?
004362  001403                  BEQ     NOR..E          ;IF YES THEN SYNTAX OK
004364  121127  000057          CMPB    @R1,#'/         ;TERMINATOR IN SET COMMAND?
004370  001002                  BNE     NOR.ER          ;IF NOT THEN SYNTAX ERROR
004372  000241          NOR..E: CLC                     ;INDICATE SUCCESSFUL OPERATION
004374  000207                  RETURN                  ;
004376  000261          NOR.ER: SEC                     ;INDICATE FAILURE
004400  000207                  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
                                        ;
                                        ;
004402  005201          $..IAS: INC     R1              ;POINT TO "E"
004404  122127  000105          CMPB    (R1)+,#'E       ;CORRECT
004410  001024                  BNE     76$             ;NO IF NE
004412  122127  000124          CMPB    (R1)+,#'T       ;CORRECT SYNTAX ?
004416  001021                  BNE     76$             ;NO IF NE
004420  122127  000057          CMPB    (R1)+,#'/       ;CORRECT SYNTAX ?
004424  001016                  BNE     76$             ;NO IF NE
004426  004737  004304          CALL    $..NOR          ;BUILD ADDRESS TO MODIFY
004432  103413                  BCS     76$             ;IF CS, FORMAT ERROR
004434  010037  004470          MOV     R0,2$           ;SAVE
004440  122127  000057          CMPB    (R1)+,#'/       ;CORRECT SYNTAX ?
004444  001006                  BNE     76$             ;NO IF NE
004446  004737  004304          CALL    $..NOR          ;BUILD DATA TO INSERT
004452  103403                  BCS     76$             ;IF CS, FORMAT ERROR
004454  010077  000010          MOV     R0,@2$          ;AND LOAD
```

```
    864 004460  000402                      BR      IAS..E          ;GO TO END
    865 004462  004737  006262      76$:    CALL    $..STX          ;WRITE  ERROR MESSAGE
    866 004466  000207              IAS..E: RETURN                  ;RETURN
    867 004470  000000              2$:     .WORD   0               ;SORAGE FOR ADDRESS TO MODIFY
    868                                     .DSABL  LSB
```

```
                        ;
                        ;
                        ;
                                .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     -       DRINT 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
004472  010602          $..IAE: MOV     SP,R2           ;;;FLAG FOR VERY FIRST CHARACTER
004474  012703  007160          MOV     @PSDSWR,R3      ;;;GET 'SWR" ADDRESS
004500  005004                  CLR     R4              ;;;USE AS  CHARACTER 1'
004502  062701  000003          ADD     #3,R1           ;;;SKIP IDENTIFYER
004506  122127  000075          CMPB    (R1)+,#'-       ;;;CORRECT SYNTAX?
004512  001137                  BNE     77$             ;;;BRANCH IF NOT
004514  122127  000057  1$:     CMPB    (R1)+,#'/       ;;;IS NEXT CHAR. A '/'
004520  001403                  BEQ     2$              ;;;IF EQ, YES
004522  005702                  TST     R2              ;;;VERY FIRST?
004524  001132                  BNE     77$             ;;;IF YES: SYNTAX ERROR
004526  000526                  BR      11$             ;;;ASSUME END OF STRING
004530  005002          2$:     CLR     R2              ;;;NO LONGER VERY FIRST
004532  005204                  INC     R4              ;;;SET FOR 'C1' FLAG
004534  121127  000116          CMPB    (R1),#'N        ;;;IS IT THE  /N  OPTION?
004540  001002                  BNE     3$              ;;;BRANCH IF NOT
004542  005201                  INC     R1              ;;;ADVANCE TO NEXT CHARACTER
004544  005004                  CLR     R4              ;;;RESET 'C1' FLAG
004546  121127  000123  3$:     CMPB    (R1),#'S        ;;;IS IT A 'S ?
004552  001426                  BEQ     5$              ;;;BRANCH IF YES
004554  121127  000103          CMPB    (R1),#'C        ;;;IS IT A  C  ?
004560  001437                  BEQ     7$              ;;;     BRANCH IF YES
004562  121127  000110          CMPB    (R1),#'H        ;;;IS IT A H  ?
004566  001472                  BEQ     9$              ;;; BRANCH IF YES
004570  121127  000120          CMPB    (R1),#'P        ;;;IS IT A  P  ?
004574  001106                  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$:     INC     R1              ;;;NEXT CHARACTER
 933 004614  122127  000122                CMPB    (R1)+,#'R       ;;;IS IT A '/PR' ?
 934 004620  001074                        BNE     77$             ;;;IF NO: SYSNTAX 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$:     INC     R1              ;;;NEXT CHARACTER
 942 004644  122127  000103                CMPB    (R1)+,#'C       ;;;IS IT A'/SC' ?
 943 004650  001060                        BNE     77$             ;;;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$:   CMPB    1(R1),#'O       ;;;CHECK FOR '/CONT
 947 004666  001016                        BNE     70$             ;;;IF NE , OTHER
 948 004670  126127  000002  000116        CMPB    2(R1),#'N       ;;;CHECK FOR ;/CONT'
 949 004676  001045                        BNE     77$             ;;;IF NO: SYNTAX ERROR
 950 004700  126127  000003  000124        CMPB    3(R1),#'T       ;;;CHECK FOR '/CONT'
 951 004706  001041                        BNE     77$             ;;;IF NO: SYSNTAX 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$:     INC     R1              ;;;ADVANCE TO NEXT CHARACTER
 960 004740  122127  000124                CMPB    (R1)+,#'T       ;;;IS IT A'/CT  OPTION ?
 961 004744  001022                        BNE     77$             ;;;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$:    INC     R1              ;;;ADVANCE TO NEXT CHARACTER
 969 004770  122127  000124                CMPB    (R1)+,#'T       ;;;IS IT A '/HT' ?
 970 004774  001006                        BNE     77$             ;;;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$:    CMPB    (R1),#15        ;;;WAS LAST BYTE A <CR>?
 974 005010  001402                        BEQ     IAE..E          ;;;BRANCH IF YES
 975 005012  004737  006262        77$:    CALL    $..STX          ;;;SYNTAX ERROR
 976 005016  000207                IAE..E: RETURN                  ;;;RETURN TO CALLER
 977                                       .DSABL LSB
```

```
                        .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 UWITH LEADING ZEROES
                ;INPUT IS RETURNED IN R0.
                ;THE PREVIOUS CONTEND OF R0 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 TRUCTATED TO THE RIGHTMOST (LOW ORDER)BIT.
                ;IN CASE OF LESS THAN 6 DIGITS INPUT, LEADING ZERO DIGITS
                ;ARE INSERTED.
                ;THE RESULT IS RETURNED IN R0,HEREBY OVERWRITING
                ;THE PREVIOUS CONTENT OF R0.
                ;
                ;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 R0, HEREBY OVERWRITING THE
                ;PREVIOUS CONTENT OF R0.
                ;
                ;ASCI 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 R0 IS REQUESTED,ONLY THE FIRST
                ;CHARACTER WILL BE RETURNED IN THE LOW ORDER BYTE
                ;OF R0.
                ;OLD CONTEND OF R0 IS OVERWRITTEN IN THIS CASE.
                ;IF THE ONLY INPUT IS <CR>, ZERO WILL BE RETURNED IN R0
                ;AS WELL AS IN THE FIRST BYTE OF THE SPECIFIED BUFFER.
                ;
                ;AUTHOR:         BERT HUBER, CSS/DP MUNICH       29 JUN 76
                ;
005020  004737  006136  $..RED: CALL    $..PRO          ;PRINT PROMPT STRING
```

```
$..RED

1036 005024  010637  007156                MOV     SP,INPREQ       ;SET SPECIAL REQUST 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 INPUTPIONTER
1045 005070  016602  000014                MOV     14(SP),R2       ;GET INPUTMOFE INDICATOR
1046 005074  000172  005100                JMP     @2$(R2)         ;SELECT REQ. INPUT
1047 005100  005112                2$:     $..OCT                  ;HANDLE OCTAL INPUT
1048 005102  005270                        $..DEC                  ;HANDLE DECIMAL INPUT
1049 005104  005504                        $..BIN                  ;HANDLE BINARY INPUT
1050 005106  005444                        $..BUF                  ;HANDLE ASCII INPUT INTO BUFFER
1051 005110  005464                        $..ASC                  ;HANDLE ASCII INPUT INTO BUFFER
1052 005112  005000                $..OCT: CLR     R0              ;CLEAR INPUT POINTER
1053 005114  005037  005244                CLR     6$              ;RESET COUNTER
1054 005120  121127  000015                CMPB    @R1,#15         ;ZERO INPUT ?
1055 005124  001014                        BNE     3$              ;IF NE, SKIP
1056 005126                                POP     R5,R4,R3,R2,R1  ;***ABOO: RESTORE GEN. REGISTERS
1057 005140  011666  000002                MOV     (SP),2(SP)      ;***ABOO: REAJUST RETURN ADDRESS
1058 005144  005726                        TST     (SP)+           ;***ABOO:
1059 005146  005037  007156                CLR     INPREQ          ;***ABOO: RESET SPECIAL REQUEST COUNTER
1060 005152  000261                        SEC                     ;***ABOO: SET CARRY AS ZERO INPUT INDICATOR
1061 005154  000207                        RETURN                  ;***ABOO: 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  152100                        BISB    (R1)+,R0        ;INSERT INTO BUFFER
1076 005242  005227                        INC     (PC)+           ;COUNT INPUTS HANDLED
1077 005244  000000                6$:     .WORD   0               ;DIGIT COUNTER
1078 005246  121127  000015                CMPB    @R1,#15         ;END OF INPUT DETECTED?
1079 005252  001341                        BNE     3$              ;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                $..DFC: CLR     R0              ;CLEAR TRANSFER BUFFER
1086 005272  005004                        CLR     R4              ;RESET COUNTER
1087 005274  122721  000015        1$:     CMPB    #15,(R1)+       ;END OF INPUT ?
1088 005300  001402                        BEQ     2$              ;IF EQ, YES
1089 005302  005204                        INC     R4              ;DIGIT COUNTER
1090 005304  000773                        BR      1$              ;CEEP COUNTING
1091 005306  012701  007026        2$:     MOV     #KBBUFF,R1      ;SET INPUT BUFFER POINTER
1092 005312  020427  000005                CMP     R4,#5           ;MORE THAN 5 DIGITS ?
```

```
$..RED

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  005037  007174                  CLR     $.$ASF          ;CLEAR SPECIAL ASCII FLAG
1125 005462  000461                          BR      RED..E          ;END OF INPUT
1126 005464  005000                   $..ASC: CLR     R0              ;
1127 005466  005037  007174                  CLR     $.$ASF          ;CLEAR SPECIAL ASCII INPUT FLAG
1128 005472  122711  000015                  CMPB    #15,@R1         ;ZERO INPUT
1129 005476  001453                          BEQ     RED..E          ;IF EQ, YES
1130 005500  111100                          MOVB    @R1,R0          ;GET FIRST BYTE
1131 005502  000451                          BR      RED..E          ;END OF INPUT
1132 005504  005000                   $..BIN: CLR     R0              ;
1133 005506  005037  005624                  CLR     6$              ;RESET COUNTER
1134 005512  122711  000015                  CMPB    #15,@R1         ;ZERO INPUT?
1135 005516  001443                          BEQ     RED..E          ;END OF INPUT
1136 005520  022737  000021  005624   1$:    CMP     #17.,6$         ;TOO MANY DIGITS?
1137 005526  001013                          BNE     2$              ;IF NE, NO
1138 005530                           5$:    WRITE   MILBIN          ;ILLEGAL INPUT MESSAGE
1139 005540                                  POP     R5,R4,R3,R2,R1  ;REST. REGS
1140 005552  000137  005020                  JMP     $..RED          ;REENTER
1141 005556  121127  000060           2$:    CMPB    @R1,#60 ;ZERO DIGIT ?
1142 005562  001407                          BEQ     3$              ;BRANCH IF YES
1143 005564  121127  000061                  CMPB    @R1,#61         ;ONE DIGIT ?
1144 005570  001011                          BNE     4$              ;BRANCH IF NOT
1145 005572  006300                          ASL     R0              ;ADJUST TRANSFER BUFFER
1146 005574  052700  000001                  BIS     #BIT0,R0        ;INSERT INTO TRANSFER BUFFER
1147 005600  000401                          BR      7$              ;
1148 005602  006300                   3$:    ASL     R0              ;ADJUST TRANSFER BUFFER
```

```
1149 005604  005201                7$:     INC     R1              ;POINT TO NEXT INPUT
1150 005606  005237  005624                INC     6$              ;COUNT INPUTS
1151 005612  000742                        BR      1$              ;CHECK NEXT
1152 005614  121127  000015        4$:     CMPB    @R1,#15         ;END OF INPUT ?
1153 005620  001343                        BNE     5$              ;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  000002                MOV     (SP),2(SP)      ;READJUST STACK RETURN ADDRESS
1158 005644  005726                        TST     (SP)+           ;       "
1159 005646  005037  007156                CLR     INPREQ          ;RESET SPECIAL REQUEST POINTER
1160 005652  000207                        RETURN                  ;RETURN TO CALLER
```

```
                                        .SBTTL  $..WRT
                                ;+
                                ;SUBROUTINE:    $..WRT
                                ;***********    ******
                                ;
                                ;THIS ROUTINE IS CALLED TO SETTUP A MESSAGE PRINTOUT ON CONSOLE
                                ;ALL PRINTING IS CONTROLLED BY INTERRUPT EXEPT <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 CSSMON
                                        .ENABL  LSB
005654  005737  007152          $..WRT: TST     OUTFLG          ;OUTPUT RUNNING ?
005660  001375                          BNE     $..WRT          ;IF NE, YES - WAIT
005662  005737  007150                  TST     INFLAG          ;INPUT RUNNING ?
005666  001372                          BNE     $..WRT          ;IF NE, YES - WAIT
005670  010637  007152                  MOV     SP,OUTFLG       ;SET OUTPUTFLAG
005674  016627  000002                  MOV     2(SP),(PC)+             ;GET MESSAGE POINTER
005700  000000                  $..MSP: .WORD   0               ;MESSAGE POINTER
005702  012737  005736  000064          MOV     #1$,@#64        ;SETUP VECTOR
005710  012737  000340  000066          MOV     #340,@#66       ;;;     "
005716  012616                          MOV     (SP)+,(SP)      ;UPDATE STACK
005720  052737  000100  177564          BIS     #100,@#TPS      ;ENABLE INTERRUPTS
005726  005737  007152                  TST     OUTFLG          ;OUTPUT RUNNING?        ;**GS0001**
005732  001375                          BNE     .-4             ;IF NE ,YES->WAIT       ;**GS0001**
005734  000207                          RETURN                  ;RETURN TO CALLER
005736  122737  000023  177562  1$:     CMPB    #23,@#TKB       ;;;  CHECK FOR XOFF
005744  001401                          BEQ     3$              ;;;  XOFF CONDITION WAIT TO CLEAR
005746  000404                          BR      20$             ;;;  NO XOFF THEN CONTINUE
005750  122737  000021  177562  3$:     CMPB    #21,@#TKB       ;;;  CHECK FOR XON
005756  001367                          BNE     1$              ;;;  NO XON GO WAIT FOR IT
005760                          20$:
005760  127727  177714  000135          CMPB    @$..MSP,#']             ;END OF MESSAGE ?
005766  001434                          BEQ     WRT..E          ;;;IF EQ  YES
005770  127727  177704  000133          CMPB    @$..MSP,#'[             ;;;<CR>,<LF>?
005776  001022                          BNE     30$             ;;;IF NE, NO
006000  042737  000100  177564          BIC     #100,@#TPS      ;;;DISABLE INTERRUPTS
006006  012737  000015  177566          MOV     #15,@#TPB       ;;;SEND <CR>
006014  105737  177564                  TSTB    @#TPS           ;;;WAIT FOR DONE
006020  100375                          BPL     .-4             ;;;
006022  005237  005700                  INC     $..MSP          ;;;POINT TO NEXT CHARACTER
006026  012737  000012  177566          MOV     #12,@#TPB       ;;;SEND <LF>
006034  052737  000100  177564          BIS     #100,@#TPS      ;;;RE-ENABLE INTERRUPT
006042  000002                          RTI                     ;;;RETURN TO MAIN PROGRAM
006044  117737  177630  177566  30$:     MOVB    @$..MSP,@#TPB   ;;;SEND NEXT CHARACTER
006052  005237  005700                  INC     $..MSP          ;;;POINT TO NEXT
006056  000002                          RTI                     ;;;RETURN TO CALLER
```

```
IEC11 A DIAGNOSTIC      MACRO M1200  30 MAR 84 16:02  PAGE 20 1                                SEQ 46
$..WRT

 1219 006060  042737  000100  177564  WRT..E: BIC     #100,@#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          ;;;NECCESARY 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
```

```
$..KBO

                                .SBTTL  $..KBO
                        ;*
                        ;SUBROUTINE:    $..KBO
                        ;***********    ******
                        ;
                        ;KEYBOARD ECHO ROUTINE FOR TEMINAL INPUT
                        ;
                        ;INPUT:         (SP)    -       CALLER PC
                        ;               R0      -       PATTERN TO ECHO
                        ;
                        ;OUTPUT:        PRINT PATTERN TO TERMINAL
                        ;
                        ;AUTHOR:        BERT HUBER CSS/DP MUNICH        28-JUN 76
                        ;-
                                ;.PSECT CSSMON
                                .ENABL  LSB
006122  105737  177564  $..KBO: TSTB    @#TPS           ;ECHO RUNNING ?
006126  100375                  BPL     $..KBO          ;IF PL, WAIT
006130  010037  177566          MOV     R0,@#TPB        ;ECHO CHARACTER
006134  000207                  RETURN                  ;RETURN TO CALLER
                                .DSABL  LSB
```

```
                                .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
006136  005037  007150  $..PRO: CLR     INFLAG          ;;;RESET INPUT FLAG
006142                          PREAD   STATUS  ;SAVE STATUS    ;MZ001
006166                          PSET    #0      ;CLEAR PS       ;MZ001
006212                          WRITE   PROMES          ;;;WRITE PROMPT STRING
006222  005737  007152  1$:     TST     OUTFLG          ;WAIT FOR OUTPUT COMPLETE
006226  001375                  BNE     1$              ;
006230                          PSET    STATUS          ;RELOAD STATUS  ;MZ001
006254  010637  007150          MOV     SP,INFLAG               ;;;SET INPUT FLAG
006260  000207                  RETURN                  ;RETURN TO CALLING PROGRAM
                                .DSABL  LSB
```

```
                                .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 '6
                        ;
                        ;
                        ;       MZ001
                        ;       *****
                        ;       THE USE OF THIS MACRO'S ALLOWS TO RUN ONE VERSION OF CDM
                        ;       ON LSI AND PDP.
                        ;
                        ;-
                                ;.PSECT CSSMON
                                .ENABL  LSB
006262  000240          $..STX: NOP                     ;;;DUMMY
006264                          PREAD   STATUS  ;SAVE STATUS    ;MZ001
006310                          PSET    #0      ;CLEAR PS       ;MZ001
006334                          WRITE   MSYERR          ;WRITE SYNTAX ERROR MESSAGE
006344  005737  007152    1$:   TST     OUTFLG  ;WAIT FOR OUTPUT COMPLETE
006350  001375                  BNE     1$              ;
006352                          PSET    STATUS          ;RELOAD PS      ;MZ001
006376  000207                  RETURN                  ;;;RETURN TO CALLER
                                .DSABL  LSB
```

```
                        .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.
                ;
                ;
```

```
                                        ;
                                        ;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
006400                          $..DMP: PUSH    R0                      ;SAVE R0
006402  016600  000004                  MOV     4(SP),R0                        ;GET TYPE ARGUMENT
006406  005737  007152                  TST     OUTFLG                  ;OUTPUT RUNNING ?
006412  001375                          BNE     .-4                     ;IF NE, WAIT
006414  000170  006420                  JMP     @D..TYP(R0)             ;SELECT FUNCTION ROUTINE
006420  006426                  D..TYP: $..DOC
006422  006520                          $..DDC
006424  006630                          $..DBN
006426  012746  000006          $..DOC: MOV     #6,-(SP)                ;SETUP DIGIT COUNT
006432  016646  000010                  MOV     10(SP),-(SP)            ;GET VALUE TO DUMP
006436  012700  000060                  MOV     #'0,R0                  ;SETUP BASIC ASCII VALUE
006442  006316                          ASL     (SP)                    ;BUILD FIRST DIGIT
006444  103001                          BCC     1$                      ;IF CC, DIGIT IS A '0'
006446  005200                          INC     R0                      ;OTHERWISE DIGIT IS A ?
006450  004737  006122          1$:     CALL    $..KBO                  ;DUMP DIGIT
006454  005366  000002                  DEC     2(SP)                   ;COUNT DIGITS DUMPED
006460  001415                          BEQ     4$                      ;IF EQ, ALL DONE
006462  012700  000060                  MOV     #'0,R0                  ;SETUP NEXT DIGIT BASE
006466  006316                          ASL     (SP)                    ;BUILD NEXT DIGIT
006470  103002                          BCC     2$                      ;IF CC, DIGIT INCLUDES ?????
006472  062700  000004                  ADD     #4,R0                   ;DIGIT = BASE+4
006476  006316                  2$:     ASL     (SP)                    ;ANALYZE NEXT BIT
006500  103001                          BCC     3$                      ;IF CC, DIGIT INCLUDES ????
006502  122020                          CMPB    (R0)+,(R0)+             ;DIGIT = DIGIT + 2
006504  006316                  3$:     ASL     (SP)                    ;ANALIZE LAST BIT OF DIGIT
006506  103360                          BCC     1$                      ;IF CC, DIGIT MUST BE EVEN
006510  005200                          INC     R0                      ;OTHERWISE DIGIT IS ODD
006512  000756                          BR      1$                      ;GO AND DUMP
006514  022626                  4$:     CMP     (SP)+,(SP)+             ;READJUST STACK
006516  000466                          BR      DMP..E                  ;END OF DUMP
                                        .DSABL  LSB
                                        .ENABL  LSB
006520  012746  000005          $..DDC: MOV     #5,-(SP)                ;SETUP A DIGIT COUNT
006524  016646  000010                  MOV     10(SP),-(SP)            ;GET VALUE TO DUMP
006530  012700  000060                  MOV     #'0,R0  ;BUILD LEADING ZERO
006534  004737  006122                  CALL    $..KBO                  ;AND PRINT
006540  012700  000060          1$:     MOV     #'0,R0  ;SETUP BASIC ASCII VALUE
006544  027716  000044          2$:     CMP     @$POINT,(SP)            ;CMP VALUE WITH ACT. DEC. DIGIT
006550  101004                          BHI     3$                      ;VALUE IS SMALLER
006552  005200                          INC     R0                      ;COUNT NUMBER OF DIVISIONS
006554  167716  000034                  SUB     @$POINT,(SP)            ;DIVIDE BY ACT. DEC. DIGIT
006560  000771                          BR      2$                      ;DO NEXT
006562  004737  006122          3$:     CALL    $..KBO                  ;PRINT THIS DEC. DIGIT
006566  062737  000002  006614          ADD     #2,$POINT               ;POINT TO NEXT DEC. VALUE
```

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

```
                                        .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
006706                          $..ADP: WRITE   MER1    ;"SHOULD BE"
006716                                  DUMP    OCT,$SHLD       ;DUMP DATA
006730                                  WRITE   MER2            ;"WAS"
006740                                  DUMP    OCT,$WAS        ;DUMP   DATA
006752  000207                          RETURN                  ;RETURN TO MAIN
```

```
                                        .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
                                ;        R0   - 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
                                ;-
                                ;
                                ;
006754  062706  000004          $..TOT: ADD     #4,SP           ;CORRECT SP
006760                                  WRITE   $MNOAV          ;SAY ADDRESS NOT AVAILABLE ON UNIBUS
006770                                  DUMP    OCT,R0          ;DUMP REGISTER ADDRESS
007000  005737  007152                  TST     OUTFLG          ;SYNCHRONIZE PRINTOUT
007004  001375                          BNE     .-4             ;"
007006  012737  000006  000004          MOV     #6,@#4          ;REBUILD TRAP CATCHER
007014  000136                          JMP     @(SP)+          ;JUMP TO CONTINUATION LABEL
                                ;
                                ;
                                ;
                                ;
```

```
                                .SBTTL  $..BUF
                        ;*
                        ;SUBSECTION TO CONTAIN ALL CDM BUFFERS, CONSTANTS AND
                        ;'NOT MACRO DEFINED' PARAMETERS.
                        ;
                        ;AUTHOR:        BERT HUBER CSS/DP MUNICH
                        ;
                                ;.PSECT CSSMON
                                .ENABL  LSB
                        ;MEMORY MANAGEMENT DEFINITIONS
        172360          KDSAR0=172360
        172320          KDSDR0=172320
        172340          KISAR0=172340
        172352          KISAR5=172352
        172354          KISAR6=172354
        172356          KISAR7=172356
        172300          KISDR0=172300
        172314          KISDR6=172314
        172316          KISDR7=172316
        172200          SISDR0=172200
        177660          UDSAR0=177660
        177620          UDSDR0=177620
        177640          UISAR0=177640
        177650          UISAR4=177650
        177652          UISAR5=177652
        177654          UISAR6=177654
        177656          UISAR7=177656
        177600          UISDR0=177600
        177610          UISDR4=177610
        177612          UISDR5=177612
        177614          UISDR6=177614
        177616          UISDR7=177616
        170200          UBMPR=170200
        140000          CMODE=140000
        030000          PMODE=30000
        177572          SR0=177572
        172516          SR3=172516
007016  000000          $$PTYP: .WORD   0               ;IF 0:=LSI; IF  1:=PDP
007020  000000          $$MTYP: .WORD   0               ;IF 0:NO MM; IF 1:WITH MM
007022  000000  000000  MEMEND: .WORD   0,0             ;TOP OF CORE AND CORE EXTENSION
                                                        ;STARTING FROM HERE TILL PROMES
                                                        ;THE CONTENTS WILL BE CLEARED
                                                        ;WITH "ABO"
007026                  KBBUFF: .BLKB   80.             ;TERMINAL INPUT BUFFER
007146                  KBBEND:                         ;REFERENCE LABEL
007146  007026          KBBPNT: .WORD   KBBUFF  ;INPUT BUFFER POINTER
007150  000000          INFLAG: .WORD   0       ;INPUT 'RUNNING" FLAG
007152  000000          OUTFLG: .WORD   0       ;OUTPUT 'RUNNING" FLAG
007154  000000          ERRFLG: .WORD   0               ;'ERROR HANDLING REQ.' FLAG
007156  000000          INPREQ: .WORD   0               ;'WAIT FOR SPECIAL INPUT" FLAG
007160  000000          PSDSWR: .WORD   0               ;SOFTWARE  SWITCHREGISTER
007162  000000          RUBFLG: .WORD   0               ;RUBOUTMODE-INDICATOR FLAG
007164  000000          STATUS: .WORD   0               ;BUFFER PS-CONTENT IF SAVED
007166  000000          $..ER:  .WORD   0               ;HOLDS CURRENT ERROR NUMBER
007170  000000          SPEFLG: .WORD   0               ;USED IN $..KBI AND $..WRT
007172  001000          SCOPAD: 1000            ;SAVE AREA FOR SCOPE LOOP ADDRESS
007174  000000          $.$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 010101      133     111     114 MILTR1: .ASCII  /[ILLEGAL INTERRPT 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 010614                           ;       WRITE   M1              ;IDENTIFY TEST
   010614  012746  021606                   MOV     #M1,-(SP)               ;SAVE #M1 ONTO STACK
   010620  004737  005654                   CALL    $..WRT                  ;INITIATE PRINTOUT
 6                                  ;
 7 010624                           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 010634                                   READ    OCT,R0          ;INPUT REQUEST
   010634  005046                           CLR      (SP)                   ;INDICATE OCTAL INPUT REQUIRED
   010636  004737  005020                   CALL    $..RED                  ;ACCEPT INPUT
 9 010642                                   IF R0 NE #0 GOTO 1$     ;
   010642  005700                           TST     R0                      ;IS R0 ZERO ?
   010644  001402                           BEQ     .+6                     ;
   010646  000137  010656                   JMP     1$                      ;
10 010652  012700  160010                   MOV     #160010,R0      ;SAVE DEFAULT VALUE
11 010656  032700  000007           1$:     BIT     #7,R0           ;CHECK THREE LSB OF INPUT
12 010662  001360                           BNE     RGAIN           ;IF NOT CLEAR,REQUEST AGAIN
13 010664                                   IF R0 LO #160010 GOTO RGAIN     ;
   010664  020027  160010                   CMP     R0,#160010              ;COMPARE R0 AND #160010
   010670  103001                           BHIS    .+4                     ;
   010672  000754                           BR      RGAIN                   ;
14 010674                                   IF R0 HI #177776 GOTO RGAIN     ;
   010674  020027  177776                   CMP     R0,#177776              ;COMPARE R0 AND #177776
   010700  101401                           BLOS    .+4                     ;
   010702  000750                           BR      RGAIN                   ;
15 010704                                   ADRTST  RGAIN
   010704  012737  006754  000004           MOV     #$..TOT,@#04            ;PROVIDE TIMEOUT ENTRY
   010712  012746  010624                   MOV     #RGAIN,-(SP)            ;SAVE RETURN ADDRESS ON STACK
   010716  005710                           TST     (R0)                    ;IS ADDRESS AVAILABLE ?
   010720  012737  000006  000004           MOV     #6,@#04                 ;RESTORE TIMOUT VECTOR
   010726  005726                           TST     (SP)+                   ;CORRECT STACK
16 010730  010037  024204                   MOV     R0,CIR          ;SAVE CIR ADDRESS
17 010734  062700  000002                   ADD     #2,R0           ;BUILD NEXT ADDRESS
18 010740                                   ADRTST  RGAIN
   010740  012737  006754  000004           MOV     #$..TOT,@#04            ;PROVIDE TIMEOUT ENTRY
   010746  012746  010624                   MOV     #RGAIN,-(SP)            ;SAVE RETURN ADDRESS ON STACK
   010752  005710                           TST     (R0)                    ;IS ADDRESS AVAILABLE ?
   010754  012737  000006  000004           MOV     #6,@#04                 ;RESTORE TIMOUT VECTOR
   010762  005726                           TST     (SP)+                   ;CORRECT STACK
19 010764  010037  024206                   MOV     R0,SMR          ;SAVE SMR REG ADDRESS
20 010770  062700  000002                   ADD     #2,R0           ;BUILD NEXT ADDRESS
21 010774                                   ADRTST  RGAIN
   010774  012737  006754  000004           MOV     #$..TOT,@#04            ;PROVIDE TIMEOUT ENTRY
   011002  012746  010624                   MOV     #RGAIN,-(SP)            ;SAVE RETURN ADDRESS ON STACK
   011006  005710                           TST     (R0)                    ;IS ADDRESS AVAILABLE ?
   011010  012737  000006  000004           MOV     #6,@#04                 ;RESTORE TIMOUT VECTOR
   011016  005726                           TST     (SP)+                   ;CORRECT STACK
22 011020  010037  024212                   MOV     R0,IOR          ;SAVE IOR REG ADDRESS
23 011024  062700  000001                   ADD     #1,R0           ;BUILD HIGH-BYTE ADDRESS
24 011030  010037  024210                   MOV     R0,IORHB        ;OF IOR
25 011034  062700  000001                   ADD     #1,R0           ;BUILD NEXT ADDRESS
26 011040                                   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     (R0)                    ;IS ADDRESS AVAILABLE ?
                011054  012737  000006  000004          MOV     #6,@#4                  ;RESTORE TIMOUT VECTOR
                011062  005726                          TST     (SP)+                   ;CORRECT STACK
            27  011064  010037  024214                  MOV     R0,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,R0          ;INPUT REQUEST
                011100  005046                          CLR     -(SP)                   ;INDICATE OCTAL INPUT REQUIRED
                011102  004737  005020                  CALL    $..RED                  ;ACCEPT INPUT
            30  011106                                  IF R0 NE #0 GOTO 2$     ;
                011106  005700                          TST     R0                      ;IS R0 ZERO ?
                011110  001402                          BEQ     .+6                     ;
                011112  000137  011122                  JMP     2$                      ;
            31  011116  012700  000270                  MOV     #270,R0         ;SAVE DEFAULT VALUE
            32  011122                          2$:     IF R0 LO #40 GOTO VECAIN        ;
                011122  020027  000040                  CMP     R0,#40          ;COMPARE R0 AND #40
                011126  103001                          BHIS    .+4                     ;
                011130  000757                          BR      VECAIN                  ;
            33  011132                                  IF R0 HI #774 GOTO VECAIN       ;
                011132  020027  000774                  CMP     R0,#774         ;COMPARE R0 AND #774
                011136  101401                          BLOS    .+4                     ;
                011140  000753                          BR      VECAIN                  ;
            34  011142  020077  013046                  CMP     R0,@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 R0,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,R0          ;INPUT
                011200  005046                          CLR      (SP)                   ;INDICATE OCTAL INPUT REQUIRED
                011202  004737  005020                  CALL    $..RED                  ;ACCEPT INPUT
            41  011206                                  IF R0 NE #0 GOTO 3$     ;
                011206  005700                          TST     R0                      ;IS R0 ZERO ?
                011210  001402                          BEQ     .+6                     ;
                011212  000137  011222                  JMP     3$                      ;
            42  011216  012700  000035                  MOV     #35,R0          ;SAVE DEFAULT VALUE
            43  011222                          3$:     IF R0 LO #1 GOTO ADDRIN         ;
                011222  020027  000001                  CMP     R0,#1           ;COMPARE R0 AND #1
                011226  103001                          BHIS    .+4                     ;
                011230  000757                          BR      ADDRIN                  ;
            44  011232                                  IF R0 HI #36. GOTO ADDRIN       ;
                011232  020027  000044                  CMP     R0,#36.         ;COMPARE R0 AND #36.
                011236  101401                          BLOS    .+4                     ;
                011240  000753                          BR      ADDRIN                  ;
            45  011242  010037  024226                  MOV     R0,IECAAD       ;SAVE IEC11 A BUS ADDRESS
            46  011246  012737  000100  024216  2$:     MOV     #100,TALKER     ;PRE-ADDRESS TALKER
            47  011254  012737  000040  024220          MOV     #40,LISTNR      ;PRE ADDRESS LISTENER
            48  011262  050037  024216                  BIS     R0,TALKER       ;SAVE TALKER ADDRESS
            49  011266  050037  024220                  BIS     R0,LISTNR       ;SAVE LISTENER ADDRESS
            50  011272  004737  011740                  JSR     PC,TEST0        ;TEST LON SWITCH FEATURE; **V02E00**
            51  011276  017700  012712          8$:     MOV     @VSR,R0         ;CHECK VECTOR ADDRESS
```

```
    52 011302  020027  000774                  CMP     R0,#774         ;LEGAL VECTOR ?
    53 011306  101003                          BHI     30$;            ;IF HI NO          ;**V01E02**
    54 011310  020027  000040                  CMP     R0,#40          ;LEGAL VECTOR      ;**V01E02**
    55 011314  103005                          BHIS    4$              ;IF HIS YES        ;**V01E02**
    56 011316                          30$:    WRITE   MES2;           ;REQUEST ANOTHER VECTOR ADDRESS
       011316  012746  021666                  MOV     #MES2, (SP)             ;SAVE #MES2 ONTO STACK
       011322  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
    57 011326  000763                          BR      8$;             ;CHECK AGAIN
    58 011330  020027  000040          4$:     CMP     R0,#40;         ;CHECK RANGE
    59 011334  10?? 0                          BLO     30$;            ;IF LO, INVALID
    60 011336  010037  024222                  MOV     R0,VECTOR       ;GET VECTOR ADDRESS
```

```
                                                .SBTTL  TEST SELECT ROUTINE
011342                                  $..SPC: REMVEC  @VECA
011342  012746  024224                          MOV     @VECA, (SP)             ;SAVE @VECA ONTO STACK
011346  104002                                  EMT     2                       ;RESET @VECA VECTOR
011350  005737  007152                          TST     OUTFLG          ;WAIT FOR ANY OUTPUT COMPLETE
011354  001375                                  BNE     .-4                     ;...
011356  000005                                  RESET                           ;CLEAR THE WORLD
011360                                          SET6    @@TKS                   ;RE-ENABLE TTY
011360  052737  000100  177560                  BIS     @#BIT6,@@TKS
011366                                          WRITE   TPMSG           ;ASK FOR PRINTING START MESSAGES
011366  012746  024070                          MOV     @TPMSG,-(SP)            ;SAVE @TPMSG ONTO STACK
011372  004737  005654                          CALL    $..WRT                  ;INITIATE PRINTOUT
011376                                          READ    ASC             ;Y OR N?
011376  012746  000010                          MOV     @10.,-(SP)              ;INDICATE ASCII INPUT IN R0 REQUIRED
011402  004737  005020                          CALL    $..RED                  ;ACCEPT INPUT
011406                                          IF R0 NE @'Y GOTO SFLAG ;SET FLAG
011406  020027  000131                          CMP     R0,@'Y          ;COMPARE R0 AND @'Y
011412  001402                                  BEQ     .+6                     ;
011414  000137  011426                          JMP     SFLAG                   ;
011420  005037  011670                          CLR     MSGFLG          ;PRINT OUT START/END MESSAGES
011424  000402                                  BR      LAB11
011426  005237  011670                  SFLAG:  INC     MSGFLG          ;NO PRINT OUT
011432                                  LAB11:  WRITE   MESNEX          ;REQUEST NEXT TEST TO RUN
011432  012746  024044                          MOV     @MESNEX,-(SP)           ;SAVE @MESNEX ONTO STACK
011436  004737  005654                          CALL    $..WRT                  ;INITIATE PRINTOUT
011442                                          READ    DEC,R0          ;WAIT FOR INPUT
011442  012746  000002                          MOV     @2, (SP)                ;INDICATE DECIMAL INPUT REQUIRED
011446  004737  005020                          CALL    $..RED                  ;ACCEPT INPUT
011452                                          IF R0 HI @14. GOTO LAB11        ;
011452  020027  000016                          CMP     R0,@14.         ;COMPARE R0 AND @14.
011456  101401                                  BLOS    .+4                     ;
011460  000764                                  BR      LAB11                   ;
011462                                          IF R0 EQ @0 GOTO TESTAL ;
011462  005700                                  TST     R0                      ;IS R0 ZERO ?
011464  001002                                  BNE     .+6                     ;
011466  000137  011536                          JMP     TESTAL                  ;
011472  012737  177777  011666                  MOV     @177777,TSTREP  ;LOOP ON SINGLE TESTS
011500  006300                                  ASL     R0                      ;BUILD WORD OFFSET INTO TABLE
011502  010027                                  MOV     R0,(PC)+                ;AND SAVE
011504  000000                          OFFSET: .WORD   0               ;HOLDS CURRENT TEST NUMBER
011506  013737  011504  011664          SINGLE: MOV     OFFSET,OFFCNT   ;
011514  062737  011626  011664                  ADD     @TT,OFFCNT      ;
011522  017737  000136  011664                  MOV     @@OFFCNT,OFFCNT ;
011530  004777  000130                          JSR     PC,@@OFFCNT     ;GOTO SELECTED TEST
011534  000764                                  BR      SINGLE          ;REPEAT TEST
011536  012737  000000  011666  TESTAL: MOV     @0,TSTREP       ;NO LOOP ON SINGLE TESTS
011544  012727  011630                          MOV     @TT+2,(PC)+     ;GET ADDRESS OF FIRST TEST
011550  000000                          SEQPNT: .WORD   0               ;POINTER FOR TEST IN PROGRESS
011552  013737  011550  011664  SEQ:    MOV     SEQPNT,OFFCNT   ;LOAD ADDRESS OF NEXT TEST
011560  017737  000100  011664                  MOV     @@OFFCNT,OFFCNT ;
011566  004777  000072                          JSR     PC,@@OFFCNT     ;GOTO EXECUTE TEST
011572  023727  011550  011662                  CMP     SEQPNT,@TTE 4   ;ALL DONE ?
011600  001404                                  BEQ     TESTL1          ;IF EQ, YES
011602  062737  000002  011550                  ADD     @2,SEQPNT       ;POINT TO NEXT TEST
011610  000760                                  BR      SEQ             ;AND DO NEXT
011612                                  TESTL1: WRITE MESEND            ;END OF TEST
011612  012746  024166                          MOV     @MESEND, (SP)           ;SAVE @MESEND ONTO STACK
```

```
                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
```

LOCAL MACRO DEFINITIONS

```
                        .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                          ; **V02E00**
                        BIS     #BIT10,@CIR     ;SET INHIBIT SRQ INTERRUPT
                        .ENDM   INHSRQ                          ; **V02E00**
                ;
                        .MACRO  INTENB
                        BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
                        .ENDM   INTENB
                ;
                ;
                        .MACRO  TSTEND  A,?L
                        TST     MSGFLG          ;PRINTOUT OR NOT?
                        BNE     L
                        WRITE   MTE'A;          ;TEST'A END MESSAGE
                L:      RETURN;                 ;RETURN TO CALLER
                        .ENDM   TSTEND
                ;
                        .MACRO  SIC
                        BIS     #BI'6,@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
                ;
```

LOCAL MACRO DEFINITIONS

```
                                        .MACRO  SRE
                                        SET7    @SMR            ;SEND REMOTE ENABLE
                                        .ENDM   SRE
                                ;
                                        .MACRO  TCA
                                        BIS     #BIT1,@SMR      ;TAKE CONTROL ASYNC
                                        .ENDM   TCA
                                ;
                                        .MACRO  TCS
                                        SET0    @SMR            ;TAKE CONTROL SYNCR.
                                        .ENDM   TCS
                                ;
                                        .MACRO  SAVIEC
                                        SAVCIR
                                        SAVSMR
                                        SAVIOR
                                        .ENDM   SAVIEC
                                ;
                                        .MACRO  LOOP    A,B,?C
                                        DEC     (PC)+           ;COUNT LOOPS
                                C:      .WORD   A               ;
                                        IF C NE #0 GOTO B
                                        MOV     #A,C            ;SETUP NEW LOOP COUNT
                                        .ENDM   LOOP


011672  013746  011734          RANDOM: MOV     RA,(SP)
011676  013700  011736                  MOV     RB,R0
011702  006316                          ASL     @SP
011704  005500                          ADC     R0
011706  006200                          ASR     R0
011710  005516                          ADC     @SP
011712  061600                          ADD     @SP,R0
011714  005616                          SBC     @SP
011716  060016                          ADD     R0,@SP
011720  005600                          SBC     R0
011722  012637  011734                  MOV     (SP)+,RA
011726  010037  011736                  MOV     R0,RB
011732  000207                          RETURN
011734  135753                  RA:     .WORD   135753
011736  024642                  RB:     .WORD   024642
                                ;
```

```
                                        .SBTTL  TEST LON SWITCH FEATURE
011740  104000                  TESTO:  SCOPE                                   ; **V02E00**
011742                                  WRITE   SLON            ;SET LON SW MESSAGE     ; **V02E00**
011742  012746  022112                  MOV     #SLON, (SP)             ;SAVE #SLON ONTO STACK
011746  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
011752                                  READ    OCT,R0
011752  005046                          CLR     -(SP)                   ;INDICATE OCTAL INPUT REQUIRED
011754  004737  005020                  CALL    $..RED                  ;ACCEPT INPUT
011760                                  MCLEAR                  ;PRODUCE CLEAN STATUS   ; **V02E00**
011760  052777  000040  012216          BIS     #BIT5,@CIR              ;EXECUTE MASTER CLEAR
011766  027727  012212  000602          CMP     @CIR,#602               ;STATE CHGE,INT,LON     ; **V02E00*
011774  001404                          BEQ     1$                      ;BRANCH IF CORRECT      ; **V02E00**
011776                                  ERROR   114.                    ;CIR CONT NOT #602      ; **V02E00**
011776  104562                          TRAP    114.                    ;ERROR 114.
012000  000000                          .WORD   0                       ;ERROR COUNT
012002  000137  011740                  JMP     TESTO                                   ; **V02E00**
012006  027727  012174  010000  1$:     CMP     @SMR,#BIT12             ;LACS SET EXCLUSIVELY?  ; **V02E00**
012014  001404                          BEQ     2$                      ;BRANCH IF O.K.         ; **V02E00**
012016                                  ERROR   115.                    ;LACS NOT OR NOT ONLY   ; **V02E00**
012016  104563                          TRAP    115.                    ;ERROR 115.
012020  000000                          .WORD   0                       ;ERROR COUNT
012022  000137  011740                  JMP     TESTO                                   ; **V02E00**
012026                          2$:     WRITE   CLON            ;RESET LON SW MSG       ; **V02E00**
012026  012746  022216                  MOV     #CLON, (SP)             ;SAVE #CLON ONTO STACK
012032  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
012036                                  READ    OCT,R0
012036  005046                          CLR     -(SP)                   ;INDICATE OCTAL INPUT REQUIRED
012040  004737  005020                  CALL    $..RED                  ;ACCEPT INPUT
012044  022777  000600  012132          CMP     #600,@CIR               ;STATE CHGE,INT         ; **V02E00**
012052  001404                          BEQ     3$                      ;BRANCH IF O.K.         ; **V02E00**
012054                                  ERROR   116.                    ;CIR CONTENT NOT #600   ; **V02E00**
012054  104564                          TRAP    116.                    ;ERROR 116.
012056  000000                          .WORD   0                       ;ERROR COUNT
012060  000137  011740                  JMP     TESTO                                   ; **V02E00**
012064  027727  012116  010000  3$:     CMP     @SMR,#BIT12             ;LACS STILL SET?        ; **V02E00**
012072  001404                          BEQ     4$                      ;BRANCH IF YES          ; **V02E00**
012074                                  ERROR   117.                    ;ERROR IF NO            ; **V02E00**
012074  104565                          TRAP    117.                    ;ERROR 117.
012076  000000                          .WORD   0                       ;ERROR COUNT
012100  000137  011740                  JMP     TESTO                                   ; **V02E00**
012104  000207                  4$:     RETURN                  ;RETURN FROM SUBTEST    ; **V02E00**
```

```
                                        .SBTTL  TEST 1  EXECUTE MASTER CLEAR
012106  005737  011670          TEST1:  TST     MSGFLG          ;PRINTOUT OR NOT?
012112  001004                  BNE     LOPT1
012114                          WRITE   MT1S            ;TEST1 START MESSAGE
012114  012746  022324                  MOV     #MT1S, (SP)             ;SAVE #MT1S ONTO STACK
012120  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
012124  104000                  LOPT1:  SCOPE                   ;SCOPE LOOP MARKER
012126                          MCLEAR
012126  052777  000040  012050          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
012134  017700  012046                  MOV     @SMR,R0         ;SAVE SMR CONTENT
012140  001404                  BEQ     1$              ;IF EQ, YES
012142                          ERROR   1.              ;ILLEGAL SMR AFTER MC
012142  104401                  TRAP    1.                      ;ERROR 1.
012144  000000                  .WORD   0                       ;ERROR COUNT
012146  000137  012106          JMP     TEST1           ;
012152                  1$:     SAVCIR
012152  017700  012026                  MOV     @CIR,R0         ;SAVE CIR CONTENT
012156  001404                  BEQ     2$              ;IF EQ, NO BIT SET
012160                          ERROR   2.              ;ILLEGAL CIR AFTER MC
012160  104402                  TRAP    2.                      ;ERROR 2.
012162  000000                  .WORD   0                       ;ERROR COUNT
012164  000137  012106          JMP     TEST1           ;
012170  017700  012016          2$:     MOV     @IOR,R0         ;SAVE IOR CONTENT
012174  001404                  BEQ     3$              ;IF EQ, NO BIT SET
012176                          ERROR   3.              ;ILLEGAL IOR AFTER MC
012176  104403                  TRAP    3.                      ;ERROR 3.
012200  000000                  .WORD   0                       ;ERROR COUNT
012202  000137  012106          JMP     TEST1           ;
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
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
```

TEST 2 TEST SACS BIT IN CIR

```
                                        .SBTTL  TEST 2  TEST SACS BIT IN CIR
012250  005737  011670          TEST2:  TST     MSGFLG          ;PRINTOUT OR NOT?
012254  001004                          BNE     LOPT2
012256                                  WRITE   MT2S            ;TEST2 START MESSAGE
012256  012746  022405                  MOV     #MT2S, (SP)             ;SAVE #MT2S ONTO STACK
012262  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
012266                          LOPT2:  MCLEAR                  ;PRODUCE CLEAR CONTROLLER STATE
012266  052777  000040  011710          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
012274  104000                          SCOPE                   ;SCOPE LOOP MARKER
012276                                  SACS                    ;
012276  052777  000001  011700          BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
012304                                  SAVCIR
012304  017700  011674                  MOV     @CIR,R0         ;SAVE CIR CONTENT
012310                                  TSTO    R0              ;
012310  032700  000001                  BIT     #BIT0,R0
012314  001004              .           BNE     1$              ;IF NE, SET
012316                                  ERROR   4.              ;SACS BIT NOT SET
012316  104404                          TRAP    4.                      ;ERROR 4.
012320  000000                          .WORD   0                       ;ERROR COUNT
012322  000137  012250                  JMP     TEST2           ;
012326  104000                  1$:     SCOPE                   ;SCOPE LOOP MARKER
012330                                  CLR0    @CIR            ;TRY TO RESET SACS BIT
012330  042777  000001  011646          BIC     #BIT0,@CIR
012336                                  SAVCIR
012336  017700  011642                  MOV     @CIR,R0         ;SAVE CIR CONTENT
012342                                  TSTO    R0              ;SACS RESET ?
012342  032700  000001                  BIT     #BIT0,R0
012346  001404                          BEQ     2$              ;IF EQ, YES
012350                                  ERROR   5.              ;SACS NOT RESET BY BIC
012350  104405                          TRAP    5.                      ;ERROR 5.
012352  000000                          .WORD   0                       ;ERROR COUNT
012354  000137  012250                  JMP     TEST2
012360  104000                  2$:     SCOPE                   ;SCOPE LOOP MARKER
012362                                  SACS                    ;
012362  052777  000001  011614          BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
012370                                  MCLEAR                  ;
012370  052777  000040  011606          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
012376                                  SAVCIR                  ;IS SACS CLEAR AFTER MC ?
012376  017700  011602                  MOV     @CIR,R0         ;SAVE CIR CONTENT
012402  001404                          BEQ     3$              ;IF EQ, YES
012404                                  ERROR   6.              ;SACS NOT RESET BY MC
012404  104406                          TRAP    6.                      ;ERROR 6.
012406  000000                          .WORD   0                       ;ERROR COUNT
012410  000137  012250                  JMP     TEST2           ;
012414                          3$:     LOOP    1,LOPT2
012414  005327                          DEC     (PC)+           ;COUNT LOOPS
012416  177777                  64$:    .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
012436                                  TSTEND  2
012436  005737  011670                  TST     MSGFLG          ;PRINTOUT OR NOT?
012442  001004                          BNE     66$
012444  012746  022452                  MOV     #MTE2, (SP)             ;SAVE #MTE2 ONTO STACK
012450  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
012454  000207                  66$:    RETURN;                 ;RETURN TO CALLER
```

```
                                        .SBTTL  TEST 3  TEST CIR BIT 10 INHIBIT SRQ INTERRUPT
012456  005737  011670          TEST3:  TST     MSGFLG          ;PRINTOUT OR NOT?
012462  001004                          BNE     LOPT2A
012464                                  WRITE   MT2AS           ;GIVE TEST3 START MSG   ; **V02E00**
012464  012746  022466                  MOV     #MT2AS,-(SP)            ;SAVE #MT2AS ONTO STACK
012470  004737  005654                  CALL    $.MPT                   ;INITIATE PRINTOUT
012474                          LOPT2A: MCLEAR                  ;PRODUCE CLEAR CONTR STATE ; **V02E00**
012474  052777  000040  011502          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
012502  104000                          SCOPE                   ;SCOPE LOOP ENTRY       ; **V02E00**
012504                                  INHSRQ                  ;SET SRQ INT INHIBIT    ; **V02E00**
012504  052777  002000  011472          BIS     #BIT10,@CIR     ;SET INHIBIT SRQ INTERRUPT
012512                                  TST10   @CIR            ;IS BIT 10 IN CIR SET?  ; **V02E00**
012512  032777  002000  011464          BIT     #BIT10,@CIR
012520  001004                          BNE     1$              ;BRANCH IF SET          ; **V02E00**
012522                                  ERROR   105.            ;BIT 10 IN CIR NOT SET  ; **V02E00**
012522  104551                          TRAP    105.                    ;ERROR 105.
012524  000000                          .WORD   0                       ;ERROR COUNT
012526  000137  012456                  JMP     TEST3                                   ; **V02E00**
012532                          1$:     CLR10   @CIR            ;RESET BIT 10 IN CIR    ; **V02E00**
012532  042777  002000  011444          BIC     #BIT10,@CIR
012540                                  TST10   @CIR            ;CLEAR AGAIN ?          ; **V02E00**
012540  032777  002000  011436          BIT     #BIT10,@CIR
012546  001404                          BEQ     2$              ;BRANCH IF YES          ; **V02E00**
012550                                  ERROR   106.            ;BIT 10 NOT CLEAR       ; **V02E00**
012550  104552                          TRAP    106.                    ;ERROR 106.
012552  000000                          .WORD   0                       ;ERROR COUNT
012554  000137  012456                  JMP     TEST3                                   ; **V02E00**
012560  104000                  2$:     SCOPE
012562                                  INHSRQ                  ;SET SRQ INT INHIBIT    ; **V02E00**
012562  052777  002000  011414          BIS     #BIT10,@CIR     ;SET INHIBIT SRQ INTERRUPT
012570                                  TST10   @CIR            ;LOOK IF SET AGAIN      ; **V02E00**
012570  032777  002000  011406          BIT     #BIT10,@CIR
012576  001004                          BNE     3$              ;BRANCH IF O.K.         ; **V02E00**
012600                                  ERROR   107.            ;BIT 10 IN CIR NOT SET AGAIN ; **V02E00**
012600  104553                          TRAP    107.                    ;ERROR 107.
012602  000000                          .WORD   0                       ;ERROR COUNT
012604  000137  012456                  JMP     TEST3
012610                          3$:     MCLEAR                  ;RESET BIT10 IN CIR BY MC ; **V02E00**
012610  052777  000040  011366          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
012616                                  TST10   @CIR            ;LOOK IF DONE           ; **V02E00**
012616  032777  002000  011360          BIT     #BIT10,@CIR
012624  001404                          BEQ     4$              ;BRANCH IF O.K.         ; **V02E00**
012626                                  ERROR   110.            ;BIT10 IN CIR  NOT CLEAR; **V02E00**
012626  104556                          TRAP    110.                    ;ERROR 110.
012630  000000                          .WORD   0                       ;ERROR COUNT
012632  000137  012456                  JMP     TEST3                                   ; **V02E00**
012636                          4$:     LOOP    1,LOPT2A        ;LOOP IN STEP?          ; **V02E00**
012636  005327                          DEC     (PC)+           ;COUNT LOOPS
012640  177777                  64$:    .WORD   1               ;
012642  005737  012640                  TST     64$                     ;IS 64$ ZERO ?
012646  001401                          BEQ     .+4                     ;
012650  000711                          BR      LOPT2A                  ;
012652  012737  177777  012640          MOV     #-1,64$         ;SETUP NEW LOOP COUNT
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
```

```
TEST 5  TEST CIA BIT 10 INHIBIT SRQ INTERRUPT

012672  004737  005654                 CALL    $..WRT          ;INITIATE PRINTOUT
012676  000207              666$:      RETURN;                 ;RETURN TO CALLER
```

```
                                        .SBTTL  TEST 4  TEST SACS SIC SIAS ILLMSG CACS
                        TEST4:  TST     MSGFLG          ;PRINTOUT OR NOT?
                                BNE     LOPT3
                                WRITE   MT3S            ;START MESSAGE TEST 4
012706  012746  022570                  MOV     #MT3S, (SP)             ;SAVE #MT3S ONTO STACK
012712  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
                        LOPT3:  MCLEAR                  ;
012716  052777  000040  011260          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
                                SACS                    ;
012724  052777  000001  011252          BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
                                SCOPE                   ;SCOPE LOOP MARKER
                                SIC                     ;SEND INTERFACE CLEAR
012734  052777  000100  011244          BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
                                NOP                     ;WAIT SOME MICRO SECONDS
                                NOP
                                NOP
                                SAVCIR
012750  017700  011230                  MOV     @CIR,R0         ;SAVE CIR CONTENT
                                SAVSMR                  ;SAVE SMR CONTENT
012754  017701  011226                  MOV     @SMR,R1         ;SAVE SMR CONTENT
                                CLR8    @CIR            ;RESET "STATE CHANGE" BIT
012760  042777  000400  011216          BIC     #BIT8,@CIR
                                TST12   R0              ;ILLEGAL MESSAGE ?
012766  032700  010000                  BIT     #BIT12,R0
                                BEQ     1$              ;IF EQ, NO
                                ERROR   7.              ;ILLEGAL MESSAGE
                                TRAP    7.                      ;ERROR 7.
012776  000000                  .WORD   0                       ;ERROR COUNT
                                JMP     TEST4           ;
                        1$:     TST13   R1              ;SIAS SET ?
013004  032701  020000                  BIT     #BIT13,R1
                                BNE     2$              ;IF NE, YES
                                ERROR   10.             ;SIAS NOT SET
                                TRAP    10.                     ;ERROR 10.
013014  000000                  .WORD   0                       ;ERROR COUNT
                                JMP     TEST4           ;
                        2$:     TST8    R0              ;STATE CHANGE ?
013022  032700  000400                  BIT     #BIT8,R0
                                BNE     3$              ;IF NE, YES
                                ERROR   11.             ;NO STATE CHANGE TO SIAS
                                TRAP    11.                     ;ERROR 11.
013032  000000                  .WORD   0                       ;ERROR COUNT
                                JMP     TEST4           ;
                        3$:     TST8    R1              ;CACS SET ?
013040  032701  000400                  BIT     #BIT8,R1
                                BNE     10$             ;IF NE, YES
                                ERROR   12.             ;CACS NOT SET BY SIAS
                                TRAP    12.                     ;ERROR 12.
013050  000000                  .WORD   0                       ;ERROR COUNT
                                JMP     TEST4           ;
                        10$:    CLR     11$             ;
                        4$:     TST13   @SMR            ;SIAS RESET ?
013062  032777  020000  011116          BIT     #BIT13,@SMR
                                BEQ     5$              ;IF EQ, YES
                                INC (PC)+               ;WAIT A WHILE
                        11$:    .WORD   0               ;
                                BNE     4$              ;
```

```
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
```

```
013274  004737  005654                 CALL    $..WRT                  ;INITIATE PRINTOUT
013300  000207                 66$:    RETURN;                 ;RETURN TO CALLER
```

```
                                                .SBTTL  TEST 5  TEST INT ENB
013302  005737  011670          TEST5:  TST     MSGFLG          ;PRINTOUT OR NOT?
013306  001004                  BNE     LOPT3A
013310                          WRITE   MT3AS           ;START MESSAGE TEST 5
013310  012746  022663          MOV     #MT3AS,-(SP)            ;SAVE #MT3AS ONTO STACK
013314  004737  005654          CALL    $..WRT                  ;INITIATE PRINTOUT
013320                  LOPT3A: MCLEAR                  ;MASTER CLEAR
013320  052777  000040  010656  BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
013326  013701  024204          MOV     CIR,R1          ;GET CIR ADDRESS
013332  104000                  SCOPE                   ;SCOPE LOOP MARKER
013334                          SET6    @R1             ;TRY TO SET INTERRUPT ENABLE
013334  052711  000100          BIS     #BIT6,@R1
013340                          TST6    @R1             ;IS INT ENBL SET ?
013340  032711  000100          BIT     #BIT6,@R1
013344  001004                  BNE     1$              ;IF NE, YES
013346  011100                  MOV     @R1,R0          ;SAV CIR
013350                          ERROR   33.             ;INTERRUPT ENBALE NOT SET
013350  104441                  TRAP    33.                     ;ERROR 33.
013352  000000                  .WORD   0                       ;ERROR COUNT
013354  000752                  BR      TEST5           ;
013356  104000          1$:     SCOPE                   ;SCOPE LOOP MARKER
013360                          CLR6    @R1             ;TRY TO RESET INTERRUPT ENABLE
013360  042711  000100          BIC     #BIT6,@R1
013364                          TST6    @R1             ;IS INT ENABLE RESET ?
013364  032711  000100          BIT     #BIT6,@R1
013370  001404                  BEQ     2$              ;IF EQ, YES
013372  011100                  MOV     @R1,R0          ;SAVE CIR
013374                          ERROR   34.             ;INTERRUPT ENABLE NOT RESET
013374  104442                  TRAP    34.                     ;ERROR 34.
013376  000000                  .WORD   0                       ;ERROR COUNT
013400  000740                  BR      TEST5           ;
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
                        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
```

```
                                                .SBTTL  TEST 6  TEST INTERRUPT WITH STATE CHGE
                                        TEST6:  TST     MSGFLG          ;PRINTOUT OR NOT?
                                                BNE     10$
                                                WRITE   MT4S            ;TEST 6 START MESSAGE
013452  012746  022734                          MOV     #MT4S, (SP)             ;SAVE #MT4S ONTO STACK
013456  004737  005654                          CALL    $..WRT                  ;INITIATE PRINTOUT
                                        10$:    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
                                        LOPT4:  SCOPE                   ;SCOPE LOOP MARKER
                                                MCLEAR
013502  052777  000040  010474                  BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
                                                MOV     #BIT12,R5       ;SETUP INTERRUPT MASK
                                                MOV     #3$,R4          ;SETUP RETURN ADDRESS
                                                INTENB
013520  052777  000100  010456                  BIS     #BIT6,@CIR      ;ENABLE INTERRUFT IN IEC-11A
                                                CLRB    @#PS            ;ENABLE INTERRUPTS
                                                SIC
013532  052777  000100  010446                  BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
                                                CLR     2$
                                        1$:     INC     (PC)+           ;WAIT FOR INTERRUPT
                                        2$:     .WORD   0
                                                BNE     1$
                                                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
                                                ERROR   20.             ;NO INTERRUPT TO SIC
013566  104424                                  TRAP    20.                     ;ERROR 20.
013570  000000                                  .WORD   0                       ;ERROR COUNT
                                                BR      TEST6
                                        3$:     BCC     4$              ;IF CC, INTERRUPT OK
                                                ERROR   21.             ;ILLEGAL INTERRUPT
013576  104425                                  TRAP    21.                     ;ERROR 21.
013600  000000                                  .WORD   0                       ;ERROR COUNT
                                                BR      TEST6           ;
                                        4$:     SCOPE                   ;SCOPE LOOP MARKER
                                                MCLEAR
013606  052777  000040  010370                  BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
                                                MOV     #BIT8,R5        ;SETUP INTERRUPT MASK
                                                MOV     #7$,R4          ;SETUP INTERRUPT RETURN ADDRESS
                                                INTENB
013624  052777  000100  010352                  BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
                                                SACS
013632  052777  000001  010344                  BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
                                                SIC
013640  052777  000100  010340                  BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
                                                CLR     6$              ;WAIT FOR INTERRUPT
                                        5$:     INC     (PC)+
                                        6$:     .WORD   0
                                                BNE     5$
                                                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$:         LOOP    1,LOPT4
    013712  005327                              DEC     (PC)+           ;COUNT LOOPS
    013714  177777                      64$:    .WORD   1               ;
    013716  005737  013714                      TST     64$                     ;IS 64$ ZERO ?
    013722  001401                              BEQ     .+4                     ;
    013724  000665                              BR      LOPT4                   ;
    013726  012737  177777  013714              MOV     #-1,64$         ;SETUP NEW LOOP COUNT
442 013734                                      TSTEND  4
    013734  005737  011670                      TST     MSGFLG          ;PRINTOUT OR NOT?
    013740  001004                              BNE     66$
    013742  012746  023013                      MOV     #MTE4,-(SP)             ;SAVE #MTE4 ONTO STACK
    013746  004737  005654                      CALL    $..WRT                  ;INITIATE PRINTOUT
    013752  000207                      66$:    RETURN;                 ;RETURN TO CALLER
```

```
                                                .SBTTL  TEST 7  TEST STATE CHGE INTERRUPT WITH SRAS
013754  005737  011670         TEST7:  TST     MSGFLG          ;PRINTOUT OR NOT?
013760  001004                         BNE     LOPT5
013762                                 WRITE   MT5S            ;START MESSAGE TEST 7
013762  012746  023027                 MOV     #MT5S, (SP)             ;SAVE #MT5S ONTO STACK
013766  004737  005654                 CALL    $..WRT                  ;INITIATE PRINTOUT
013772                         LOPT5:  MCLEAR                  ;MASTER CLEAR
013772  052777  000040  010204         BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
014000                                 SACS                    ;SET CONTROLLER ACTIVE STATE
014000  052777  000001  010176         BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
014006  104000                         SCOPE                   ;SCOPE LOOP MARKER
014010                                 SRE                     ;SEND REMOTE ENABLE
014010  052777  000200  010170         BIS     #BIT7,@SMR
014016  105777  010164                 TSTB    @SMR            ;SRE SET ?
014022  100407                         BMI     5$              ;IF MI, YES
014024                                 SAVCIR
014024  017700  010154                 MOV     @CIR,R0         ;SAVE CIR CONTENT
014030                                 SAVSMR
014030  017701  010152                 MOV     @SMR,R1         ;SAVE SMR CONTENT
014034                                 ERROR   24.             ;SRE NOT SET
014034  104430                         TRAP    24.                     ;ERROR 24.
014036  000000                         .WORD   0                       ;ERROR COUNT
014040  000745                         BR      TEST7           ;
014042                         5$:     CLR8    @CIR            ;RESET STATE CHANGE
014042  042777  000400  010134         BIC     #BIT8,@CIR
014050                                 TST12   @CIR            ;ILLEGAL MESSAGE ?
014050  032777  010000  010126         BIT     #BIT12,@CIR
014056  001407                         BEQ     1$              ;IE EQ, NO
014060                                 SAVCIR
014060  017700  010120                 MOV     @CIR,R0         ;SAVE CIR CONTENT
014064                                 SAVSMR
014064  017701  010116                 MOV     @SMR,R1         ;SAVE SMR CONTENT
014070                                 ERROR   25.             ;'ILLEGAL MESSAGE
014070  104431                         TRAP    25.                     ;ERROR 25.
014072  000000                         .WORD   0                       ;ERROR COUNT
014074  000727                         BR      TEST7           ;
014076  005037  014114         1$:     CLR     9$              ;WAIT A SHORT ITME
014102                         10$:    TST14   @SMR            ;SRAS SET ?
014102  032777  040000  010076         BIT     #BIT14,@SMR
014110  001014                         BNE     2$              ;IF NE, YES
014112  005227                         INC     (PC)+
014114  000000                 9$:     .WORD   0
014116  001371                         BNE     10$
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
014134                                 ERROR   26.             ;SRAS NOT SET
014134  104432                         TRAP    26.                     ;ERROR 26.
014136  000000                         .WORD   0                       ;ERROR COUNT
014140  000705                         BR      TEST7           ;
014142                         2$:     TST8    @CIR            ;STATE CHANGE ?
014142  032777  000400  010034         BIT     #BIT8,@CIR
014150  001003                         BNE     3$              ;IF NE, YES
014152                                 ERROR   27.             ;NO STATE CHANGE
014152  104433                         TRAP    27.                     ;ERROR 27.
014154  000000                         .WORD   0                       ;ERROR COUNT
```

```
                                        BR      TEST7           ;
                                3$:     CLR7    @SMR            ;RESET SRE
014160  042777  000200  010020          BIC     #BIT7,@SMR
                                        NOP                     ;WAIT SOME MICRO SECONDS
                                        NOP
                                        NOP
                                        SAVCIR                  ;SAVE CIR CONTENT
014174  017700  010004                  MOV     @CIR,R0         ;SAVE CIR CONTENT
                                        SAVSMR                  ;SAVE SMR CONTENT
014200  017701  010002                  MOV     @SMR,R1         ;SAVE SMR CONTENT
                                        TSTB    R1              ;SRE RESET ?
                                        BPL     6$              ;IF PL, YES
                                        ERROR   30.             ;SRE NOT RESET
014210  104436                          TRAP    30.                     ;ERROR 30.
014212  000000                          .WORD   0                       ;ERROR COUNT
                                        BR      TEST7           ;
                                6$:     CLR8    @CIR            ;RESET STATE CHANGE
014216  042777  000400  007760          BIC     #BIT8,@CIR
                                        TST14   R1              ;SRAS RESET ?
014224  032701  040000                  BIT     #BIT14,R1
                                        BEQ     4$              ;IF EQ, YES
                                        ERROR   31.             ;SRAS NOT RESET
014232  104437                          TRAP    31.                     ;ERROR 31.
014234  000000                          .WORD   0                       ;ERROR COUNT
                                        BR      TEST7           ;
                                4$:     SWAB    R0              ;
                                        TST8    R0              ;STATE CHANGE SET ?
014242  032700  000400                  BIT     #BIT8,R0
                                        BNE     7$              ;IF NE, YES
                                        SWAB    R0              ;REBUILD ORIGINAL CONTENT
                                        ERROR   32.             ;NO STATE CHANGE
014252  104440                          TRAP    32.                     ;ERROR 32.
014254  000000                          .WORD   0                       ;ERROR COUNT
                                        BR      TEST7
                                7$:     MCLEAR                  ;MASTER CLEAR
014260  052777  000040  007716          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
                                        SCOPE                   ;SCOPE LOOP MARKER
                                        SRE                     ;SEND REMOTE ENABLE
014270  052777  000200  007710          BIS     #BIT7,@SMR
                                        NOP                     ;WAIT SOME MICRO SECONDS
                                        NOP
                                        NOP
                                        SAVCIR                  ;SAVE CIR CONTENT
014304  017700  007674                  MOV     @CIR,R0         ;SAVE CIR CONTENT
                                        SAVSMR                  ;SAVE SMR CONTENT
014310  017701  007672                  MOV     @SMR,R1         ;SAVE SMR CONTENT
                                        TST12   R0              ;ILLEG. MESSAGE ?
014314  032700  010000                  BIT     #BIT12,R0
                                        BNE     8$              ;IF NE, YES
                                        ERROR   104.            ;NO ILLEG. MESSAGE
014322  104550                          TRAP    104.                    ;ERROR 104.
014324  000000                          .WORD   0                       ;ERROR COUNT
                                        BR      TEST7
                                8$:     LOOP    1,LOPT5
014330  005327                          DEC     (PC)+           ;COUNT LOOPS
014332  177777                  64$:    .WORD   1               ;
014334  005737  014332                  TST     64$                     ;IS 64$ ZERO ?
```

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

```
                                                .SBTTL  TEST 8   IOR DATA TEST
                                TEST8:  TST     MSGFLG          ;PRINTOUT OR NOT?
                                        BNE     10$
                                        WRITE   MTSAS           ;START MESSAGE TEST 8
014400  012746  023127                        MOV     @MTSAS,-(SP)            ;SAVE @MTSAS ONTO STACK
014404  004737  005654                        CALL    $..WRT                  ;INITIATE PRINTOUT
                                10$:    SCOPE
                                LOPTSA: MCLEAR                  ;
014412  052777  000040  007564                BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
014420  013704  024212                        MOV     IOR,R4          ;GET IOR ADDRESS
014424  005003                                CLR     R3              ;PREPARE DATA
014426  110314                        1$:     MOVB    R3,@R4          ;WRITE PATTERN INTO IOR
014430  011402                                MOV     @R4,R2          ;READ BACK PATTERN FROM IOR
014432  020302                                CMP     R3,R2           ;PATTERN EQUAL ?
014434  001403                                BEQ     2$              ;IF EQ, YES
                                                ERROR   47.             ;DATA CHECK IN IOR
014436  104457                                TRAP    47.                     ;ERROR 47.
014440  000000                                .WORD   0                       ;ERROR COUNT
014442  000753                                BR      TEST8           ;RESTART TEST
014444  005203                        2$:     INC     R3              ;BUILD NEXT PATTERN
014446  020327  000400                        CMP     R3,#400         ;ALL DONE ?
014452  001365                                BNE     1$              ;IF NE, NO
                                                LOOP    1000.,LOPTSA
014454  005327                                DEC     (PC)+           ;COUNT LOOPS
014456  001750                        64$:    .WORD   1000.           ;
014460  005737  014456                        TST     (4$                     ;IS 64$ ZERO ?
014464  001401                                BEQ     .+4                     ;
014466  000751                                BR      LOPTSA                  ;
014470  012737  001750  014456                MOV     #1000.,64$              ;SETUP NEW LOOP COUNT
                                                TSTEND  5A              ;RETURN TO MAINPROGRAM
014476  005737  011670                        TST     MSGFLG          ;PRINTOUT OR NOT?
014502  001004                                BNE     66$
014504  012746  023165                        MOV     @MTESA,-(SP)            ;SAVE @MTESA ONTO STACK
014510  004737  005654                        CALL    $..WRT                  ;INITIATE PRINTOUT
014514  000207                        66$:    RETURN;                 ;RETURN TO CALLER
```

```
534                                             .SBTTL  TEST 9  TEST STATE CHGE INTERRUPT WITH GTS AND SIC
535 014516                              TEST9:  SETVEC  #VECTOR,#IECINT,#0               ;SET VECTOR      ; **V03E02**
    014516  012746  024222                      MOV     #VECTOR,-(SP)           ;SAVE #VECTOR ONTO STACK
    014522  012746  021334                      MOV     #IECINT,-(SP)           ;SAVE #IECINT ONTO STACK
    014526  012746  000000                      MOV     #0,-(SP)                ;SAVE #0 ONTO STACK
    014532  104001                              EMT     1                       ;SETUP #VECTOR VECTOR
536 014534  005737  011670                      TST     MSGFLG          ;PRINTOUT OR NOT?
537 014540  001004                              BNE     LOPT6
538 014542                                      WRITE   MT6S            ;START MESSAGE TEST 9
    014542  012746  023201                      MOV     #MT6S, (SP)             ;SAVE #MT6S ONTO STACK
    014546  004737  005654                      CALL    $..WRT                  ;INITIATE PRINTOUT
539 014552                              LOPT6:  MCLEAR                  ;MASTER CLEAR
    014552  052777  000040  007424              BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
540 014560                                      SACS                    ;SET CONTROLER ACTIVE STATE
    014560  052777  000001  007416              BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
541 014566                                      SIC                     ;SEND INTERFACE CLEAR
    014566  052777  000100  007412              BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
542 014574  013700  024204                      MOV     CIR,R0          ;GET CIR ADDRESS
543 014600                              1$:     TST8    @R0             ;WAIT FOR STATE CHANGE
    014600  032710  000400                      BIT     #BIT8,@R0
544 014604  001775                              BEQ     1$              ;PRODUCED BY SIAS
545 014606  104000                              SCOPE                   ;SCOPE LOOP MARKER
546 014610                                      CLR8    @R0             ;RESET STATE CHANGE
    014610  042710  000400                      BIC     #BIT8,@R0
547 014614                                      TST8    @R0             ;WAIT FOR SIAS E**      ;**V01E01**
    014614  032710  000400                      BIT     #BIT8,@R0
548 014620  001775                              BEQ     .-4             ;                       ;**V01E01**
549 014622                                      CLR8    @R0             ;RESET STATE CHANGE     ;**V01E01**
    014622  042710  000400                      BIC     #BIT8,@R0
550 014626                                      GTS                     ;GO TO STAND BY STATE
    014626  052777  000004  007352              BIS     #BIT2,@SMR      ;GO TO STAND BY
551 014634  005037  014652                      CLR     3$              ;WAIT FOR CSBS
552 014640                              2$:     TST9    @SMR            ;
    014640  032777  001000  007340              BIT     #BIT9,@SMR
553 014646  001012                              BNE     4$              ;
554 014650  005227                              INC     (PC)+           ;
555 014652  000000                      3$:     .WORD   0               ;
556 014654  001371                              BNE     2$              ;
557 014656                                      SAVCIR                  ;SAVE CIR CONTENT
    014656  017700  007322                      MOV     @CIR,R0         ;SAVE CIR CONTENT
558 014662                                      SAVSMR                  ;SAVE SMR CONTENT
    014662  017701  007320                      MOV     @SMR,R1         ;SAVE SMR CONTENT
559 014666                                      ERROR   35.             ;NO CSBS
    014666  104443                              TRAP    35.                     ;ERROR 35.
    014670  000000                              .WORD   0                       ;ERROR COUNT
560 014672  000711                              BR      TEST9           ;RESTART TEST
561 014674                              4$:     TST8    @R0             ;DID CSBS BRING STATE CHANGE ?
    014674  032710  000400                      BIT     #BIT8,@R0
562 014700  001007                              BNE     5$              ;IF NE, YES
563 014702                                      SAVCIR                  ;SAVE CIR CONTENT
    014702  017700  007276                      MOV     @CIR,R0         ;SAVE CIR CONTENT
564 014706                                      SAVSMR                  ;SAVE SMR CONTENT
    014706  017701  007274                      MOV     @SMR,R1         ;SAVE SMR CONTENT
565 014712                                      ERROR   36.             ;NO STATE CHANGE TO CSBS
    014712  104444                              TRAP    36.                     ;ERROR 36.
    014714  000000                              .WORD   0                       ;ERROR COUNT
566 014716  000677                              BR      TEST9           ;RESTART TEST
```

```
567 014720                          5$:     CLRB    @R0             ;RESET STATE CHANGE
    014720  042710  000400                  BIC     #BIT8,@R0
568 014724  104000                        SCOPE                   ;SCOPE LOOP MARKER
569 014726                                GTS                     ;FORCE ILLEGAL MESSAGE
    014726  052777  000004  007252        BIS     #BIT2,@SMR      ;GO TO STAND BY
570 014734  005037  014750                CLR     7$              ;WAIT FOR ILLEGAL MESSAGE
571 014740                          6$:     TST12   @R0             ;
    014740  032710  010000                BIT     #BIT12,@R0
572 014744  001012                        BNE     8$              ;
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     @CIR,R0         ;SAVE CIR CONTENT
577 014760                                SAVSMR                  ;SAVE SMR CONTENT
    014760  017701  007222                MOV     @SMR,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$:   INC     R0              ;POINT TO CIR HIGH BYTE
581 014774  104000                        SCOPE                   ;SCOPE LOOP MARKER
582 014776  105010                        CLRB    @R0             ;RESET INTERRUPT CONDITIONS
583 015000                                TCA                     ;TAKE CONTROL ASYNC
    015000  052777  000002  007200        BIS     #BIT1,@SMR      ;TAKE CONTROL ASYNC
584 015006  005037  015022                CLR     10$             ;WAIT FOR CACS
585 015012  122710  000001            9$:   CMPB    #1,@R0          ;                               ; **V02E00**
586 015016  001413                        BEQ     11$             ;
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     @CIR,R0         ;SAVE CIR CONTENT
591 015032                                SAVSMR                  ;SAVE SMR CONTENT
    015032  017701  007150                MOV     @SMR,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$:   MOV     SMR,R1          ;GET ADDRESS OF SMR
595 015052  005201                        INC     R1              ;ADDRESS HIGH BYTE OF SMR
596 015054  122711  000001                CMPB    #1,@R1          ;CACS SET AGAIN ?
597 015060  001410                        BEQ     12$             ;IF EQ, YES
598 015062                                SAVCIR                  ;SAVE CIR CONTENT
    015062  017700  007116                MOV     @CIR,R0         ;SAVE CIR CONTENT
599 015066                                SAVSMR                  ;SAVE SMR CONTENT
    015066  017701  007114                MOV     @SMR,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$:   SCOPE                   ;SCOPE LOOP MARKER
603 015104  105010                        CLRB    @R0             ;RESET INTERRUPT CONDITIONS
604 015106  012705  100000                MOV     #BIT15,R5       ;SETUP INTERRUPT MASK
605 015112  012704  015200                MOV     #15$,R4         ;SETUP INTERRUPT RETURN ADDRESS
606 015116                                INTENB                  ;ENABLE IEC INTERRUPT
    015116  052777  000100  007060        BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
```

```
607 015124  105037  177776            CLRB    @#PS            ;ALLOW SYSTEM INTERRUPTS
608 015130                            SAVCIR                  ;SAVE CIR CONTENT
    015130  017700  007050            MOV     @CIR,R0         ;SAVE CIR CONTENT
609 015134  113777  024216  007050    MOVB    TALKER,@IOR     ;ADDRESS TALKER !
610 015142  005037  015150            CLR     14$             ;WAIT FOR INTERRUPT
611 015146  005227              13$:  INC     (PC)+           ;
612 015150  000000              14$:  .WORD   0               ;
613 015152  001375                    BNE     13$             ;
614 015154                            SAVIOR                  ;SAVE IOR CONTENT
    015154  017700  007032            MOV     @IOR,R2         ;SAVE IOR CONTENT
615 015160                            SAVCIR                  ;SAVE CIR CONTENT
    015160  017700  007020            MOV     @CIR,R0         ;SAVE CIR CONTENT
616 015164                            SAVSMR                  ;SAVE SMR CONTENT
    015164  017701  007016            MOV     @SMR,R1         ;SAVE SMR CONTENT
617 015170                            ERROR   42.             ;NO INTERRUPT TO TALKER ADDRESSING
    015170  104452                    TRAP    42.                     ;ERROR 42.
    015172  000000                    .WORD   0                       ;ERROR COUNT
618 015174  000137  014516            JMP     TEST9           ;RESTART TEST9
619 015200  103004              15$:  BCC     16$             ;IF CC, INTERRUPT OK
620 015202                            ERROR   43.             ;ILLEGAL INTERRUPT
    015202  104453                    TRAP    43.                     ;ERROR 43.
    015204  000000                    .WORD   0                       ;ERROR COUNT
621 015206  000137  014516            JMP TEST9               ;RESTART TEST 9
622 015212                      16$:  GTS                     ;GOTO STANDBY STATE
    015212  052777  000004  006766    BIS     #BIT2,@SMR      ;GO TO STAND BY
623 015220  005037  015234            CLR     18$             ;WAIT FOR CSBS AND TACS
624 015224  122711  000012      17$:  CMPB    #12.,@R1        ;
625 015230  001415                    BEQ     19$             ;
626 015232  005227                    INC     (PC)+           ;
627 015234  000000              18$:  .WORD   0               ;
628 015236  001372                    BNE     17$             ;
629 015240                            SAVCIR                  ;SAVE CIR CONTENT
    015240  017700  006740            MOV     @CIR,R0         ;SAVE CIR CONTENT
630 015244                            SAVSMR                  ;SAVE SMR CONTENT
    015244  017701  006736            MOV     @SMR,R1         ;SAVE SMR CONTENT
631 015250                            SAVIOR                  ;SAVE IOR CONTENT
    015250  017702  006736            MOV     @IOR,R2         ;SAVE IOR CONTENT
632 015254                            ERROR   44.             ;NO CSBS AND TACS
    015254  104454                    TRAP    44.                     ;ERROR 44.
    015256  000000                    .WORD   0                       ;ERROR COUNT
633 015260  000137  014516            JMP     TEST9           ;RESTART TEST9
634 015264                      19$:  TST11   @CIR             ;IS NO LAC SET ?
    015264  032777  004000  006712    BIT     #BIT11,@CIR
635 015272  001012                    BNE     20$             ;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                      20$:  MCLEAR                  ;RESET IFC 11A
    015320  052777  000040  006656    BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
642 015326                            LOOP     1,LOPT6
```

```
        015326  005327                          DEC     (PC)+           ;COUNT LOOPS
        015330  177777                  64$:    .WORD   1               ;
        015332  005737  015330                  TST     64$                     ;IS 64$ ZERO ?
        015336  001402                          BEQ     .+6                     ;
        015340  000137  014552                  JMP     LOPT6                   ;
        015344  012737  177777  015330          MOV     #-1,64$         ;SETUP NEW LOOP COUNT
    643 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                                  TEST10: SETVEC  #VECTOR,#IECINT,#0               ;SET VECTOR      ; **V03E02**
    015400  012746  024222                          MOV     #VECTOR,-(SP)           ;SAVE #VECTOR ONTO STACK
    015404  012746  021334                          MOV     #IECINT,-(SP)           ;SAVE #IECINT ONTO STACK
    015410  012746  000000                          MOV     #0,-(SP)                ;SAVE #0 ONTO STACK
    015414  104001                                  EMT     1                       ;SETUP #VECTOR VECTOR
648 015416  005737  011670                          TST     MSGFLG          ;PRINTOUT OR NOT?
649 015422  001004                                  BNE     LOPT7
650 015424                                          WRITE   MT7S            ;START MESSAGE TEST 10
    015424  012746  023310                          MOV     #MT7S,-(SP)             ;SAVE #MT7S ONTO STACK
    015430  004737  005654                          CALL    $..WRT                  ;INITIATE PRINTOUT
651 015434                                  LOPT7:  MCLEAR                  ;MASTER CLEAR
    015434  052777  000040  006542                  BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
652 015442                                          SACS                    ;SET CONTROLLER ACTIVE STATE
    015442  052777  000001  006534                  BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
653 015450                                          SIC                     ;SEND INTERFACE CLEAR
    015450  052777  000100  006530                  BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
654 015456                                  13$:    TST8    @CIR            ;WAIT FOR STATE CHANGE
    015456  032777  000400  006520                  BIT     #BIT8,@CIR
655 015464  001774                                  BEQ     13$             ;
656 015466                                          CLR8    @CIR            ;RESET INTERRUPT CONDITIONS
    015466  042777  000400  006510                  BIC     #BIT8,@CIR
657 015474                                          TST8    @CIR            ;WAIT FOR SIAS END      ;**V01E01**
    015474  032777  000400  006502                  BIT     #BIT8,@CIR
658 015502  001774                                  BEQ     .-6             ;                       ;**V01E01**
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$:     BIC     #177400,@CIR    ;RESET INTERRUPT CONDITIONS
676 015610                                          TCA                     ;TAKE CONTROL ASYNCR.
    015610  052777  000002  006370                  BIS     #BIT1,@SMR      ;TAKE CONTROL ASYNC
677 015616                                  5$:     TST8    @CIR            ;WAIT FOR STATE CHANGE
    015616  032777  000400  006360                  BIT     #BIT8,@CIR
678 015624  001774                                  BEQ     5$              ;
679 015626                                  6$:     TST8    @SMR            ;CACS SET ?
```

```
     015626  032777  000400  006352          BIT     #BIT8,@SMR
680  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 BITS
683  015646  012705  100000                  MOV     #BIT15,R5       ;SETUP INTERRUPT MASK
684  015652  012704  015740                  MOV     #9$,R4          ;SETUP INTERRUPT RETURN ADDRESS
685  015656                                  SAVCIR                  ;SAVE CIR CONTENT
     015656  017700  006322                  MOV     @CIR,R0         ;SAVE CIR CONTENT
686  015662                                  INTENB                  ;ENABLE IEC-INTERRUPT
     015662  052777  000100  006314          BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
687  015670  105037  177776                  CLRB    @#PS            ;ENABLE SYSTEM INTERRUPT
688  015674  113777  024220  006310          MOVB    LISTNR,@IOR     ;SEND LISTENER ADDRESS
689  015702  005037  015710                  CLR     8$              ;
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          MOVB    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$:    TST9    @SMR            ;WAIT FOR CSBS
    016062  032777  001000  006116               BIT     #BIT9,@SMR
720 016070  001774                               BEQ     15$             ;
721 016072                                       TST12   @SMR            ;LACS SET ?
    016072  032777  010000  006106               BIT     #BIT12,@SMR
722 016100  001012                               BNE     16$             ;IF EQ, NO
723 016102                                       SAVCIR                  ;SAVE CIR CONTENT
    016102  017700  006076                       MOV     @CIR,R0         ;SAVE CIR CONTENT
724 016106                                       SAVSMR                  ;SAVE SMR CONTENT
    016106  017701  006074                       MOV     @SMR,R1         ;SAVE SMR CONTENT
725 016112                                       SAVIOR                  ;SAVE IOR CONTENT
    016112  017702  006074                       MOV     @IOR,R2         ;SAVE IOR CONTENT
726 016116                                       ERROR   52.             ;NO LACS
    016116  104464                               TRAP    52.                     ;ERROR 52.
    016120  000000                               .WORD   0                       ;ERROR COUNT
727 016122  000137  015400                       JMP     TEST10          ;RESTART SUBTEST
728 016126  112777  177777  006056  16$:    MOVB    #-1,@IOR        ;SEND A DATA BYTE
729 016134                                17$:    TST15   @CIR            ;DATA ACCEPTED ?
    016134  032777  100000  006042               BIT     #BIT15,@CIR
730 016142  001774                               BEQ     17$             ;IF EQ, WAIT
731 016144                                       CLR15   @CIR            ;RESET DATA ACCEPTED
    016144  042777  100000  006032               BIC     #BIT15,@CIR
732 016152                                       CLR8    @CIR            ;RESET STATE CHANGE
    016152  042777  000400  006024               BIC     #BIT8,@CIR
733 016160  105377  006026                       DECB    @IOR            ;SEND NEXT DATA BYTE    ;**V01E02**
734 016164  005227                        18$:    INC     (PC)+           ;WAIT A SHORT TIME
735 016166  000000                               .WORD   0               ;
736 016170  001375                               BNE     18$             ;
737 016172                                       TST15   @CIR            ;DATA ACCEPTED ?
    016172  032777  100000  006004               BIT     #BIT15,@CIR
738 016200  001412                               BEQ     19$             ;IF EQ, NO
739 016202                                       SAVCIR                  ;SAVE CIR CONTENT
    016202  017700  005776                       MOV     @CIR,R0         ;SAVE CIR CONTENT
740 016206                                       SAVSMR                  ;SAVE SMR CONTENT
    016206  017701  005774                       MOV     @SMR,R1         ;SAVE SMR CONTENT
741 016212                                       SAVIOR                  ;SAVE IOR CONTENT
    016212  017702  005774                       MOV     @IOR,R2         ;SAVE IOR CONTENT
742 016216                                       ERROR   54.             ;DATA ACCEPTED TO SECOND DATA BYTE
    016216  104466                               TRAP    54.                     ;ERROR 54.
    016220  000000                               .WORD   0                       ;ERROR COUNT
743 016222  000137  015400                       JMP     TEST10          ;RESTART SUBTEST
744 016226                                19$:    SAVIOR                  ;GET IOR CONTENT
    016226  017702  005760                       MOV     @IOR,R2         ;SAVE IOR CONTENT
745 016232  000302                               SWAB    R2              ;
746 016234  120227  000377                       CMPB    R2,#377         ;DATA OK ?
747 016240  001411                               BEQ     20$             ;IF EQ, YES
748 016242  000302                               SWAB    R2              ;NO, REBUILD DATA PATTERN READ
749 016244                                       SAVSMR                  ;SAVE SMR CONTENT
    016244  017701  005736                       MOV     @SMR,R1         ;SAVE SMR CONTENT
750 016250                                       SAVCIR                  ;SAVE CIR CONTENT
    016250  017700  005730                       MOV     @CIR,R0         ;SAVE CIR CONTENT
751 016254                                       ERROR   55.             ;DATA CHECK ON IEC TALKER OR LISTENER
    016254  104467                               TRAP    55.                     ;ERROR 55.
    016256  000000                               .WORD   0                       ;ERROR COUNT
752 016260  000137  015400                       JMP     TEST10          ;RESTART SUBTEST
753 016264                                20$:    TST15   @CIR            ;SECOND DATA ACCEPTED ?
    016264  032777  100000  005712               BIT     #BIT15,@CIR
```

```
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                       21$:    SAVIOR                  ;GET SECONFD 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                                             .ENABL  LSB
778 016356  005037  016604          TST10A: CLR     RANMOD          ;DISABLE RANDOM MODE
779 016362  005046                          CLR     -(SP)           ;CLEAR DATA BUFFER
780 016364  005237  024204                  INC     CIR             ;ADDRESS CIR HIGH BYTE
781 016370  105077  005610                  CLRB    @CIR            ;RESET INTERRUPT BITS
782 016374  005337  024204                  DEC     CIR             ;REBUILD FULL CIR ADDRESS
783 016400  104000                          SCOPE                   ;SET SCOPE LOOP MARKER
784 016402  012705  100000          LOPT7A: MOV     #BIT15,R5       ;SETUP INTERRUPT MASK
785 016406  012704  016474                  MOV     #5$,R4          ;SETUP INTERRUPT RETURN ADDRESS
786 016412  042777  177400  005564        BIC     #177400,@CIR    ;RESET INTERRUPT BITS
787 016420                                  INTENB                  ;ENABLE IEC INTERRUPT
    016420  052777  000100  005556        BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC 11A
788 016426  105037  177776                  CLRB    @#PS            ;ENABLE SYSTEM INTERRUPT
789 016432  011677  005554                  MOV     (SP),@IOR       ;SEND A DATA BYTE
790 016436  005037  016444                  CLR     2$              ;WAIT FOR INTERRUPT
791 016442  005227                  1$:     INC     (PC)+           ;
792 016444  000000                  2$:     .WORD   0               ;
793 016446  001375                          BNE     1$              ;
794 016450                                  SAVIEC                  ;SAVE ALL IEC REGISTERS
    016450  017700  005530                  MOV     @CIR,R0         ;SAVE CIR CONTENT
    016454  017701  005526                  MOV     @SMR,R1         ;SAVE SMR CONTENT
    016460  017702  005526                  MOV     @IOR,R2         ;SAVE IOR CONTENT
795 016464                                  ERROR   60.             ;NO INTERRUPT TO DATA SENDING
    016464  104474                          TRAP    60.                     ;ERROR 60.
    016466  000000                          .WORD   0                       ;ERROR COUNT
796 016470  000137  015400                  JMP     TEST10          ;RESTART TEST
797 016474  103004                  3$:     BCC     4$              ;IF CC, INTERRUPT OK
798 016476                                  ERROR   61.             ;ILLEGAL INTERRUPT TO DATA SENDING
    016476  104475                          TRAP    61.                     ;ERROR 61.
    016500  000000                          .WORD   0                       ;ERROR COUNT
799 016502  000137  015400                  JMP     TEST10          ;RESTART TEST
800 016506  017703  005500          4$:     MOV     @IOR,R3         ;READ BACK DATA BYTE
801 016512  105003                          CLRB    R3              ;PREPARE DATA
802 016514  000303                          SWAB    R3              ;
803 016516                                  IF R3 EQ (SP) GOTO 5$   ;
    016516  020316                          CMP     R3,(SP)         ;COMPARE R3 AND (SP)
    016520  001002                          BNE     .+6                     ;
    016522  000137  016554                  JMP     5$                      ;
804 016526                                  SAVIEC                  ;SAVE ALL IEC REGISTERS
    016526  017700  005452                  MOV     @CIR,R0         ;SAVE CIR CONTENT
    016532  017701  005450                  MOV     @SMR,R1         ;SAVE SMR CONTENT
    016536  017702  005450                  MOV     @IOR,R2         ;SAVE IOR CONTENT
805 016542  011604                          MOV     (SP),R4         ;SAVE 'SHOULD BE   VALUE
806 016544                                  ERROR   62.             ;DATA CHECK ON IEC BUS
    016544  104476                          TRAP    62.                     ;ERROR 62.
    016546  000000                          .WORD   0                       ;ERROR COUNT
807 016550  000137  015400                  JMP     TEST10          ;RESTART TEST
808 016554                          5$:     IF RANMOD NE #0 GOTO 6$
    016554  005737  016604                  TST     RANMOD                  ;IS RANMOD ZERO ?
    016560  001402                          BEQ     .+6                     ;
```

```
    016562  000137  016606                  JMP     6$                      ;
809 016566  005216                          INC     (SP)            ;BUILD NEXT DATA PATTERN
810 016570                                  IF (SP) NE #400 GOTO LOPT7A
    016570  021627  000400                  CMP     (SP),#400               ;COMPARE (SP) AND #400
    016574  001401                          BEQ     .+4                     ;
    016576  000701                          BR      LOPT7A                  ;
811 016600  012727  177777                  MOV     #-1,(PC)+       ;SWITCH TO RANDOM MODE
812 016604  000000              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          10$:    MOV     @IOR,R3         ;READ BACK DATA BYTE
838 016756  105003                          CLRB    R3              ;
839 016760  000303                          SWAB    R3              ;PREPARE FOR CHECK
840 016762                                  IF R3 EQ #222 GOTO 11$  ;
    016762  020327  000222                  CMP     R3,#222         ;COMPARE R3 AND #222
    016766  001002                          BNE     .+6                     ;
    016770  000137  017020                  JMP     11$                     ;
841 016774                                  SAVIEC                  ;SAVE ALL IEC REGISTERS
    016774  017700  005204                  MOV     @CIR,R0         ;SAVE CIR CONTENT
    017000  017701  005202                  MOV     @SMR,R1         ;SAVE SMR CONTENT
    017004  017702  005202                  MOV     @IOR,R2         ;SAVE IOR CONTENT
842 017010                                  ERROR   65.             ;DATA CHECK ON 'LAST BYTE
```

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

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

```
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$             ;IF EQ, NO
903 017414                                      SAVIEC                  ;SAVE ALL IEC REGISTERS
    017414  017700  004564                      MOV     @CIR,R0         ;SAVE CIR CONTENT
    017420  017701  004562                      MOV     @SMR,R1         ;SAVE SMR CONTENT
    017424  017702  004562                      MOV     @IOR,R2         ;SAVE IOR CONTENT
904 017430                                      ERROR   70.             ;LACS STILL SET AFTER (LTN-GTS-TCA LUN-GTS)
    017430  104506                              TRAP    70.                     ;ERROR 70.
    017432  000000                              .WORD   0                       ;ERROR COUNT
905 017434  000137  017046                      JMP     TEST11          ;RESTART SUBTEST
906 017440                              12$:     MCLEAR                  ;MASTER CLEAR
    017440  052777  000040  004536              BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
907 017446                                      LOOP    1,LOPT8
    017446  005327                              DEC     (PC)+           ;COUNT LOOPS
    017450  177777                      64$:    .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
```

```
943 017750                                          GTS             ;GOTO <STAND BY> STATE
    017750  052777  000004  004230                  BIS     #BIT2,@SMR      ;GO TO STAND BY
944 017756                              6$:         TST12   @SMR            ;WAIT FOR <LACS>
    017756  032777  010000  004222                  BIT     #BIT12,@SMR
945 017764  001774                                  BEQ     6$              ;
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$,R4          ;SETUP INTERRUPT RETURN ADDRESS
949 020002                                          INTENB                  ;ENABLE IEC INTERRUPT
    020002  052777  000100  004174                  BIS     #BIT6,@CIR      ;ENABLE INTERRUPT IN IEC-11A
950 020010  005037  177776                          CLR     @#PS            ;ENABLE SYSTEM INTERRUPTS
951 020014  012777  000333  004170                  MOV     #333,@IOR       ;SEND A DATA PATTERN
952 020022  005227                          7$:     INC     (PC)+           ;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                              8$:         SAVIEC                  ;SAVE ALL IEC REGISTERS
    020032  017700  004146                          MOV     @CIR,R0         ;SAVE CIR CONTENT
    020036  017701  004144                          MOV     @SMR,R1         ;SAVE SMR CONTENT
    020042  017702  004144                          MOV     @IOR,R2         ;SAVE IOR CONTENT
957 020046                                          ERROR   73.             ;INTERRUPT WITH <BLOCK DAC>
    020046  104511                                  TRAP    73.                     ;ERROR 73.
    020050  000000                                  .WORD   0                       ;ERROR COUNT
958 020052  000137  017510                          JMP     TEST12          ;RESTART SUBTEST
959 020056  017703  004130              9$:         MOV     @IOR,R3         ;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                              10$:        MCLEAR                  ;MASTER CLEAR
    020124  052777  000040  004052                  BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
967 020132                                          LOOP    20.,LOPT9
    020132  005327                                  DEC     (PC)+           ;COUNT LOOPS
    020134  000024                          65$:    .WORD   20.             ;
    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    $..WRT                  ;INITIATE PRINTOUT
    020174  000207                          67$:    RETURN;                 ;RETURN TO CALLER
```

```
                                                .SBTTL  TEST 13 -- TEST FUNCTION OF <RSV>,<SRQ> AND <SPAS>
 970
 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$:    MOV     LISTNR,@IOR     ;SEND LISTENER ADDRESS
1006            020434                          5$:     TST15   @CIR            ;WAIT FOR <DATA ACCEPTED>
                020434  032777  100000  003542          BIT     #BIT15,@CIR
1007            020442  001774                          BEQ     5$              ;
1008            020444  042777  177400  003532          BIC     #177400,@CIR    ;RESET ALL INTERRUPT CONDITIONS
1009            020452  013777  024216  003532          MOV     TALKER,@IOR     ;SEND TALKER ADDRESS
1010            020460                          6$:     TST15   @CIR            ;WAIT FOR <DATA ACCEPTED>
                020460  032777  100000  003516          BIT     #BIT15,@CIR
1011            020466  001774                          BEQ     6$              ;
1012            020470  042777  177400  003506          BIC     #177400,@CIR    ;RESET ALL INTERRUPT CONDITIONS
1013            020476  012777  000030  003506          MOV     #30,@IOR        ;SEND <SERIAL POLL ENABLE>
1014            020504                          7$:     TST15   @CIR            ;WAIT FOR <DATA ACCEPTED>
                020504  032777  100000  003472          BIT     #BIT15,@CIR
1015            020512  001774                          BEQ     7$              ;
1016            020514  042777  177400  003462          BIC     #177400,@CIR    ;RESET ALL INTERRUPT CONDITIONS
1017            020522                                  GTS                     ;GOTO <STAND BY> STATE
                020522  052777  000004  003456          BIS     #BIT2,@SMR      ;GO TO STAND BY
1018            020530  005037  020546                  CLR     9$              ;
1019            020534                          8$:     TST15   @SMR            ;WAIT FOR <SPAS>
                020534  032777  100000  003444          BIT     #BIT15,@SMR
1020            020542  001015                          BNE     10$             ;
1021            020544  005227                          INC     (PC)+           ;
1022            020546  000000                  9$:     .WORD   0               ;
1023            020550  001371                          BNE     8$              ;
1024            020552                                  SAVIEC                  ;SAVE ALL IEC REGISTERS
                020552  017700  003426                  MOV     @CIR,R0         ;SAVE CIR CONTENT
                020556  017701  003424                  MOV     @SMR,R1         ;SAVE SMR CONTENT
                020562  017702  003424                  MOV     @IOR,R2         ;SAVE IOR CONTENT
1025            020566                                  ERROR   76.             ; NO <SPAS> TO <SPE>
                020566  104514                          TRAP    76.                     ;ERROR 76.
                020570  000000                          .WORD   0                       ;ERROR COUNT
1026            020572  000137  020176                  JMP     TEST13          ;RESTART SUBTEST
1027            020576  042777  177400  003400  10$:    BIC     #177400,@CIR    ;RESET ALL INTERRUPT CONDITIONS
1028            020604  012777  000100  003400          MOV     #BIT6,@IOR      ;SEND STATUS BYTE
1029            020612                          11$:    TST15   @CIR            ;WAIT FOR <DATA ACCEPTED>
                020612  032777  100000  003364          BIT     #BIT15,@CIR
1030            020620  001774                          BEQ     11$             ;
1031            020622                                  SAVIOR                  ;SAVE IOR CONTENT
                020622  017702  003364                  MOV     @IOR,R2         ;SAVE IOR CONTENT
1032            020626  105002                          CLRB    R2              ;PREPARE FOR DATA CHECK
1033            020630  000302                          SWAB    R2              ;
1034            020632                                  IF R2 EQ #BIT6 GOTO 12$ ;
                020632  020227  000100                  CMP     R2,#BIT6                ;COMPARE R2 AND #BIT6
                020636  001002                          BNE     .+6                     ;
                020640  000137  020670                  JMP     12$                     ;
1035            020644                                  SAVIEC                  ;SAVE ALL IEC REGISTERS
                020644  017700  003334                  MOV     @CIR,R0         ;SAVE CIR CONTENT
                020650  017701  003332                  MOV     @SMR,R1         ;SAVE SMR CONTENT
                020654  017702  003332                  MOV     @IOR,R2         ;SAVE IOR CONTENT
1036            020660                                  ERROR   77.             ;IEC DATA CHECK ON STATUS BYTE SENDING
                020660  104515                          TRAP    77.                     ;ERROR 77.
                020662  000000                          .WORD   0                       ;ERROR COUNT
1037            020664  000137  020176                  JMP     TEST13          ;RESTART TEST
1038            020670                          12$:    TST14   @CIR            ;IS <SRQ> OFF ?
                020670  032777  040000  003306          BIT     #BIT14,@CIR
```

```
1039 020676  001412                          BEQ     13$             ;IF EQ, YES
1040 020700                                  SAVIEC                  ;SAVE ALL IEC REGSITERS
     020700  017700  003300                  MOV     @CIR,R0         ;SAVE CIR CONTENT
     020704  017701  003276                  MOV     @SMR,R1         ;SAVE SMR CONTENT
     020710  017702  003276                  MOV     @IOR,R2         ;SAVE IOR CONTENT
1041 020714                                  ERROR   100.            ;<SRQ> NOT OFF AFTER STATUS BYTE SENDING
     020714  104544                          TRAP    100.                    ;ERROR 100.
     020716  000000                          .WORD   0                       ;ERROR COUNT
1042 020720  000137  020176                  JMP     TEST13          ;RESTART TEST
1043 020724  042777  177400  003252  13$:    BIC     #177400,@CIR    ;RESET ALL INTERRUPT CONDITIONS
1044 020732                                  TCA                     ;ENTER <CACS>
     020732  052777  000002  003246          BIS     #BIT1,@SMR      ;TAKE CONTROL ASYNC
1045 020740                          14$:    TST8    @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: TST     MSGFLG          ;PRINTOUT OR NOT?
1066 021120  001004                          BNE     LOPT11
1067 021122                                  WRITE   MT11S           ;START MESSAGE TEST 14
     021122  012746  023711                  MOV     #MT11S,-(SP)            ;SAVE #MT11S ONTO STACK
     021126  004737  005654                  CALL    $..WRT                  ;INITIATE PRINTOUT
1068 021132  104000                  LOPT11: SCOPE                   ;SCOPE LOOP MARKER
1069 021134                                  MCLEAR                  ;MASTER CLEAR
     021134  052777  000040  003042          BIS     #BIT5,@CIR      ;EXECUTE MASTER CLEAR
1070 021142                                  SACS                    ;SET <CONTROLLER ACTIVE> STATE
     021142  052777  000001  003034          BIS     #BIT0,@CIR      ;SET SACS BIT IN CIR
1071 021150                                  SIC                     ;SEND <INTERFACE CLEAR>
     021150  052777  000100  003030          BIS     #BIT6,@SMR      ;SEND INTERFACE CLEAR
1072 021156                          3$:     TST13   @SMR            ;WAIT FOR <SIAS TO GO OFF
     021156  032777  020000  003022          BIT     #BIT13,@SMR
1073 021164  001374                          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  001012                          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                          1$:     CLR3    @SMR            ;TRY TO RESET <RPP>
     021230  042777  000010  002750          BIC     #BIT3,@SMR
1081 021236                                  TST3    @SMR            ;IS BIT RESET ?
     021236  032777  000010  002742          BIT     #BIT3,@SMR
1082 021244  001412                          BEQ     2$              ;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                          2$:     LOOP    1,LOPT11
     021272  005327                          DEC     (PC)+           ;COUNT LOOPS
     021274  177777                  64$:    .WORD   1               ;
     021276  005737  021274                  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 INTERRUPT MASK INTO LOW BYTE
1095 021366                                     CLRO    10(SP)          ;ASSUME ALL IS OK
     021366  042766  000001  000010             BIC     #BIT0,10(SP)
1096 021374  120500                             CMPB    R5,R0           ;SUSPECTED INTERRUPT ?
1097 021376  001411                             BEQ     1$              ;IF EQ, YES
1098 021400  000300                             SWAB    R0              ;REBUILD SAVED CIR CONTENT
1099 021402  000305                             SWAB    R5              ;REBUILD 'SHOULD BE' CONTENT
1100 021404  062706  000012                     ADD     #12,SP          ;REBUILD STACK POINTER
1101 021410                                     POP     R0,R1,R2        ;REBUILD R0,R1 AND R2   ;**V03E01**
     021410  012600                             MOV     (SP)+,R0                ;RESTORE R0 FROM STACK
     021412  012601                             MOV     (SP)+,R1                ;RESTORE R1 FROM STACK
     021414  012602                             MOV     (SP)+,R2                ;RESTORE R2 FROM STACK
1102 021416  000261                             SEC                     ;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                             CLRB    (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  0C0004             MOV     #6,@#4          ;
1116 021506  000111                             JMP     (R1)            ;
```

```
1118                                    .SBTTL  SUBTEST ADDRESS TABLE
```

MESSAGES

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

```
1178
1179 024[illegible] 160010                    .SBTTL  CONSTANTS AND PARAMETERS
1180 024206  160012          CIR:     .WORD   160010  ;ADDRESS OF IEC CONTROL AND INTERRUPT REGISTER
1181 024210  160015          SMR:     .WORD   160012  ;ADDRESS OF IEC STATUS AND MESSAGE REGISTER
1182 024212  160014          IORHB:   .WORD   160015  ;IOR HIGH-BYTE ADDRESS DEFAULT VALUE
1183 024214  160016          IOR:     .WORD   160014  ;ADDRESS OF IEC INPUT/OUTPUT REGISTER
1184 024216  000100          VSR:     .WORD   160016  ;ADDRESS OF IEC VECTOR SWITCH REGISTER
1185 024220  000040          TALKER:  .WORD   100     ;ADDRESS OF TALKER
1186 024222  000270          LISTNR:  .WORD   40      ;ADDRESS OF LISTENER
1187 024224  000270          VECTOR:  .WORD   270     ;ADDRESS OF IEC-VECTOR
1188 024226  000035          VECA:    .WORD   270     ;DEFAULT OF IEC11-A VECTOR ADDRESS
1189         001204          IECAAD:  .WORD   35      ;IEC11 A BUS ADDRESS DEFAULT VALUE
                             .END     START
```

SYMBOL TABLE

```
ADDRIN  011170      LAB11   011432      MTE5    023113      SFLAG   011426      VECA    024224
BIT0  = 000001      LISTNR  024220      MTE5A   023165      SINGLE  011506      VECAIN  011070
BIT1  = 000002      LOPT1   012124      MTE6    023274      SISDR0= 172200      VECTOR  024222
BIT10 = 002000      LOPT10  020214      MTE7    023373      SLON    022112      VSR     024214
BIT11 = 004000      LOPT11  021132      MTE8    023472      SMR     024206      WRT..E  006060
BIT12 = 010000      LOPT2   012266      MTE9    023565      SPEFLG  007170      X     = 000152
BIT13 = 020000      LOPT2A  012474      MT1S    022324      SR0   = 177572      XX    = 000004
BIT14 = 040000      LOPT3   012716      MT10S   023602      SR3   = 172516      Z$$Z  = 000004
BIT15 = 100000      LOPT3A  013320      MT11S   023711      START   001204      Z$$$Z = 000000
BIT2  = 000004      LOPT4   013500      MT2AS   022466      STATUS  007164      $MNOAV  010351
BIT3  = 000010      LOPT5   013772      MT2S    022405      TALKER  024216      $POINT  006614
BIT4  = 000020      LOPT5A  014412      MT3AS   022663      TESTAL  011536      $SHLD   007176
BIT5  = 000040      LOPT6   014552      MT3S    022570      TESTL1  011612      $TABLE  006616
BIT6  = 000100      LOPT7   015434      MT4S    022734      TEST0   011740      $WAS    007200
BIT7  = 000200      LOPT7A  016402      MT5AS   023127      TEST1   012106      $$MTYP  007020
BIT8  = 000400      LOPT8   017064      MT5S    023027      TEST10  015400      $$PTYP  007016
BIT9  = 001000      LOPT9   017526      MT6S    023201      TEST11  017046      $$SPE   003520
CDM..E  010614      MAPER   010472      MT7S    023310      TEST12  017510      $.$ASF  007174
CIR     024204      MEMDEF  010426      MT8S    023410      TEST13  020176      $..ADP  006706
CLON    022216      MEMEND  007022      MT9S    023507      TEST14  021114      $..ASC  005464
CMODE = 140000      MERCOV  007737      MVERSN  010156      TEST2   012250      $..BIN  005504
DMP..E  006674      MERHLT  010002      M..DET  001662      TEST3   012456      $..BUF  005444
DUMMES  007210      MERROR  007652      M..HIM  001766      TEST4   012700      $..DBN  006630
D..TYP  006420      MERRP2  007662      M..MMM  002006      TEST5   013302      $..DDC  006520
EMTTBL  002416      MER1    010304      M..RET  001726      TEST6   013444      $..DEC  005270
EMTTND  002424      MER2    010333      M..SAV  001424      TEST7   013754      $..DMP  006400
EMT0    002424      MESEND  024166      M..TRP  001776      TEST8   014372      $..DOC  006426
EMT1    002432      MESNEX  024044      M..TSE  001374      TEST9   014516      $..EMT  002326
EMT2    002470      MES1    021510      M..TST  001516      TKB   = 177562      $..ER   007166
ERRFLG  007154      MES2    021666      M1      021606      TKS   = 177560      $..IAA  004120
IAD..E  004302      MES5    021750      N     = 000003      TOUT    021446      $..IAD  004210
IAE..E  005016      MES6    022034      NOR.ER  004376      TPB   = 177566      $..IAE  004472
IAS..E  004466      MILADR  007445      NOR..E  004372      TPMSG   024070      $..IAS  004402
IAY..E  004116      MILBIN  007360      NO6     010612      TPS   = 177564      $..IAY  004010
IECAAD  024226      MILDEC  007316      OFFCNT  011664      TSTREP  011666      $..INP  003574
IECINT  021334      MILEMT  007421      OFFSET  011504      TST10A  016356      $..IOT  003056
INFLAG  007150      MILOCT  007256      OUTFLG  007152      TT      011626      $..KBI  003424
INPREQ  007156      MILSCP  007524      PMODE = 030000      TTE     011666      $..KBO  006122
INP..E  003776      MILTR1  010104      PROMES  007202      TYP..E  001252      $..MAP  002116
IOR     024212      MILTR2  010144      PS    = 177776      UBMPR = 170200      $..MSP  005700
IORMB   024210      MKBOVF  007213      PSDSWR  007160      UDSAR0= 177660      $..NOR  004304
I..DMP= 000001      MNOAV   023767      RA      011734      UDSDR0= 177620      $..OCT  005112
KBBEND  007146      MNOSUB  010032      RANDOM  011672      UISAR0= 177640      $..PRO  006136
KBBPNT  007146      MPWRFL  007704      RANMOD  016604      UISAR4= 177650      $..PWR  003126
KBBUFF  007026      MSGFLG  011670      RB      011736      UISAR5= 177652      $..RED  005020
KDSAR0= 172360      MSYERR  007503      RED..E  005626      UISAR6= 177654      $..RRS  003400
KDSDR0= 172320      MTE1    022371      RGAIN   010624      UISAR7= 177656      $..RSV  003360
KISAR0= 172340      MTE10   023674      RUBFLG  007162      UISDR0= 177600      $..SPC  011342
KISAR5= 172352      MTE11   023752      R..TYP= 000001      UISDR4= 177610      $..STX  006262
KISAR6= 172354      MTE2    022452      SAV     011164      UISDR5= 177612      $..TOT  006754
KISAR7= 172356      MTE2A   022554      SCOPAD  007172      UISDR6= 177614      $..TRP  002524
KISDR0= 172300      MTE3    022647      SCOPE = 104000      UISDR7= 177616      $..TYP  001242
KISDR6= 172314      MTE3A   022720      SEQ     011552      U..DET  001760      $..WRT  005654
KISDR7= 172316      MTE4    023013      SEQPNT  011550

. ABS.  024230  000
        000000  001
```

```
IEC11 A DIAGNOSTIC      MACRO M1200  30 MAR 84 16:02  PAGE 52-2                    SEQ 103
SYMBOL TABLE

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

.
```