

IEC11-B

IEC11-B POP-11 DIAG
CZIECAO

COPYRIGHT (c) 1979-84
AH-T880A-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

IDENTIFICATION

PRODUCT CODE: AC T879A-MC
PRODUCT TITLE: CZIECAO IEC11 B PDP 11 DIAGNOSTIC
PRODUCT DATE: JANUARY 1984
DEPARTMENT: CSS, MUNICH
AUTHOR: J. MOLLER

COPYRIGHT (C) FIRST, 1979, 1984
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED
AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE
AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR
OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO
AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.
THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT
NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
EQUIPMENT CORPORATION.
DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	PRELIMINARY DIAGNOSTICS
	LOADING PROCEDURE
3.1	METHOD
4.0	STARTING PROCEDURE
5.0	OPERATING PROCEDURES
5.1	SWITCH SETTINGS
5.2	PRINTOUTS
6.0	ERRORS
6.1	ERROR FORM*
6.2	ERROR DESCRIPTION
7.0	EXECUTION TIME
8.0	MISCELLANEOUS
8.1	RESTRICTIONS
9.0	PROGRAM DESCRIPTION
10.0	DIAGNOSTIC MONITOR DESCRIPTION
11.0	LISTING

1.0 A B S T R A C T
.....

THIS PROGRAM IS DESIGNED TO TEST AND EXERCISE
AN IEC11 B INTERFACE TOGETHER WITH AN IEC11 A
INTERFACE.

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

PDP11 WITH AT LEAST 8K MEMORY, CONSOLE TERMINAL
WITH STANDARD ADDRESSES, INTERRUPT LEVEL AND
PRIORITY.
IEC11-A INTERFACE
IEC11-B INTERFACE
TESTCABLE (BC08S-XX)

2.1 P R E L I M I N A R Y D I A G N O S T I C S
.....

ALL DIAGNOSTIC PROGRAMS FOR THE IEC11 A MUST RUN
SUCCESSFUL.

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

3.1 M E T H O D
SEE DIAGNOSTIC MONITOR DESCRIPTION

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

SEE DIAGNOSTIC MONITOR DESCRIPTION

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

5.1 S W I T C H S E T T I N G S
SEE DIAGNOSTIC MONITOR DESCRIPTION

5.2 P R I N T O U T S

AFTER THE IDENTIFICATION PRINTOUT FOLLOW A FEW
 QUESTIONS FOR THE ADDRESSES AND THE VECTORS.
 IF THE INTERFACE HAS THE DEFAULT ADDRESSES
 AND VECTORS, ENTER JUST A <CR> AND THROUGH ALL THE
 TESTS THESE VALUES WILL BE USED. A TYPICAL, ERROR
 FREE PASS WILL BE EXECUTED IN THE FOLLOWING ORDER:

(NOTE THAT USER INPUT IS UNDERLINED>

CDM V3.0 CSS DIAGNOSTIC MONITOR

HIGHEST MEMORY ADDRESS :xxxxxxx
 IEC11 B TEST IN CONNECTION
 WITH AN IEC11-A INTERFACE
 AC-T879A-MC
 CSS MUNICH MARCH 1979

ENTER FIRST REGISTER ADDRESS OF IEC11 A
 (DEFAULT IS 160010)
 CDM><CR>

ENTER VECTOR ADDRESS OF IEC11 A
 (DEFAULT IS 270)
 CMD><CR>

ENTER IEC-BUS ADDRESS OF IEC11-A
 (DEFAULT IS 35)
 CMD><CR>

ENTER FIRST REGISTER ADDRESS OF IEC11-B
 (DEFAULT IS 160020)
 CDM><CR>

ENTER VECTOR ADDRESS OF IEC11 A
 (DEFAULT IS 274)
 CMD><CR>

ENTER IEC-BUS ADDRESS OF IEC11 B
 (DEFAULT IS 36)
 CMD><CR>

NEXT TEST TO RUN ?
 CMD><CR>

```

TEST 1: REGISTER STATIC TEST
END OF TEST
TEST 2: TALKER AND LISTENER FUNCTION TEST
END OF TEST
TEST 3: GENERAL INTERRUPT AND DMA FUNCTION TEST
END OF TEST
TEST 4: DMA-TRANSFER FROM B TO A (B IS TALKER)
END OF TEST
TEST 5: DMA TRANSFER FROM A TO B (B IS LISTENER)
END OF TEST
TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)
END OF TEST
TEST 7: SERIAL POLL PROCEEDURE TEST
END OF TEST
TEST 1: REGISTER STATIC TEST
<CTRL C>
*****
CDM>ABO<CR>
*****
NEXT TEST TO RUN ?
CDM>

```

```

:
:
:
:
THE ABOVE PRINTOUT IS A TYPICAL EXAMPLE FOR AN ERROR FREE RUN.

```

6.0 E R R O R S
.....

6.1 E R R O R F O R M A T

THE ERROR FORMAT IS DESCRIBED IN SECTION 10,
DIAGNOSTIC MONITOR.

6.2 E R R O R D E S C R I P T I O N

SEE PROGRAM DESCRIPTION (FLOWCHARTS)

ERRORS FROM 'REGIST' SUBROUTINE : MAY APPEAR IN TEST 1

ERROR 253: WRITE AND READ ERROR IN A REGISTER
AFTER LOADING THE R/W BITS WITH A RANDOM PATTERN
ERROR 254: WRITE AND READ ERROR IN A REGISTER
AFTER LOADING ALL THE R/W BITS WITH 0
ERROR 255: WRITE AND READ ERROR IN A REGISTER
AFTER LOADING ALL THE R/W BITS WITH 1

7.0 E X E C U T I O N T I M E
.....

THE EXECUTION TIME OF ONE SEQUENTIAL PASS DEPENDS ON
THE CONSOLE TERMINAL (BAUD RATE) AND THE TYPE OF PROCESSOR.

ONLY TEST 6 TAKES MORE THAN 10 SECONDS AFTER IDENTIFICATION.

SEE PROGRAM DESCRIPTION

8.0 M I S C E L L A N E O U S
.....

8.1 R E S T R I C T I O N S
.....

DON'T USE ADDRESS "0" FOR IEC BUS ADDRESS
ON TEST CONFIGURATION !!

MAKE SURE THAT ALL SWITCHES ON IEC11-A AND IEC11 B
ARE SET CORRECTLY

9.0 PROGRAM DESCRIPTION
.....

THE IEC11 B TEST PROGRAM IS A SET OF
SEVEN INDEPENDENT ROUTINES THAT CAN BE CALLED
FROM A SELECT ROUTINE.
WHEN THE QUESTION OCCURS.

NEXT TEST TO RUN ?

ANYONE OF THE SEVEN TESTS CAN BE SELECTED.

IF <CR> IS PRESSED, THEN ALL TESTS WILL
RUN SEQUENTIALLY.

ALL TESTS CAN BE ABORTED BY PRESSING <CTRL C>
AND TYPING "ABO"

ANY SINGLE SELECTED TEST RUNS IN AN ENDLESS LOOP
WITH NO PRINTOUT (E.G. END OF TEST).

TO RUN A TEST ERROR FREE, ALL PREVIOUS TESTS MUST PROVE
SUCCESSFUL !!

THE FOLLOWING SECTION CONTAINS THE FLOWCHARTS FOR ALL TESTS.

ALL ERROR NUMBERS WHICH ARE MARKED WITH A "*" APPEAR AFTER
A PROGRAMMED DELAY.

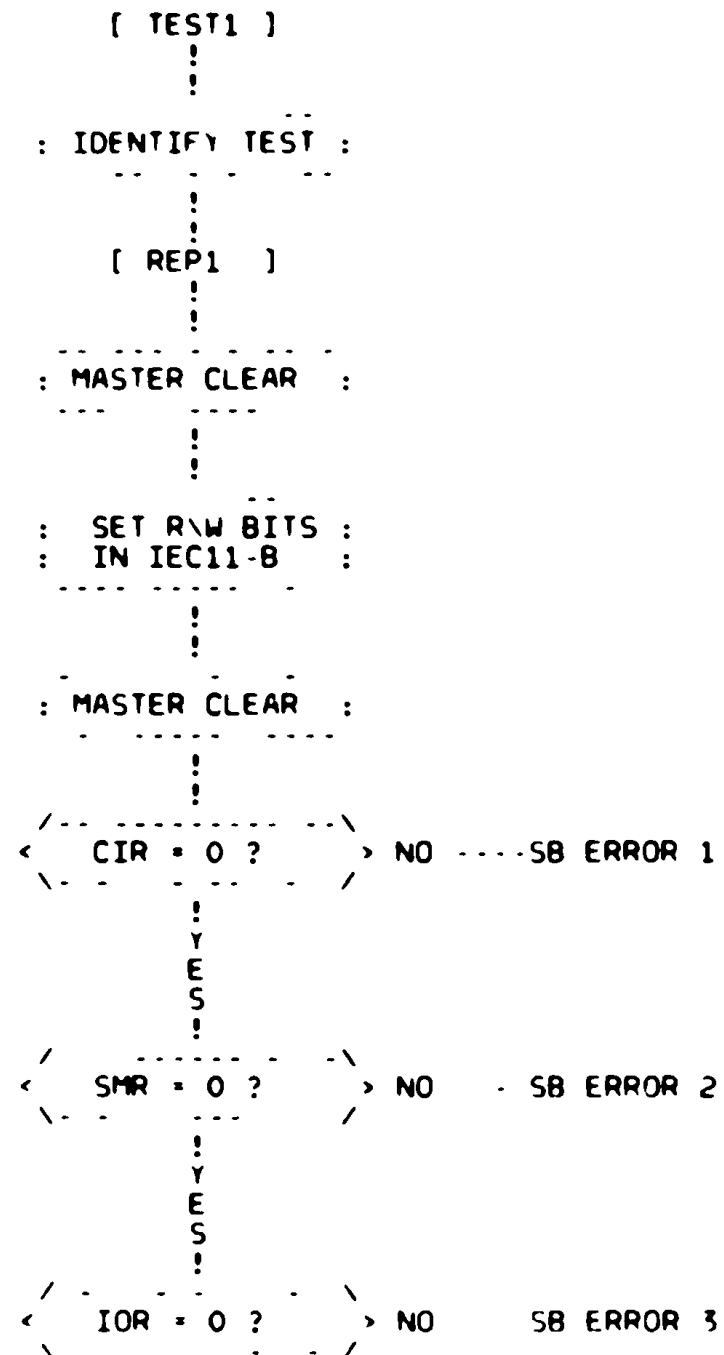
SB ERROR 35 : SB STANDS FOR AN ADDITIONAL PRINTOUT BEFORE
ERROR NUMBER PRINT OF THE FORM :

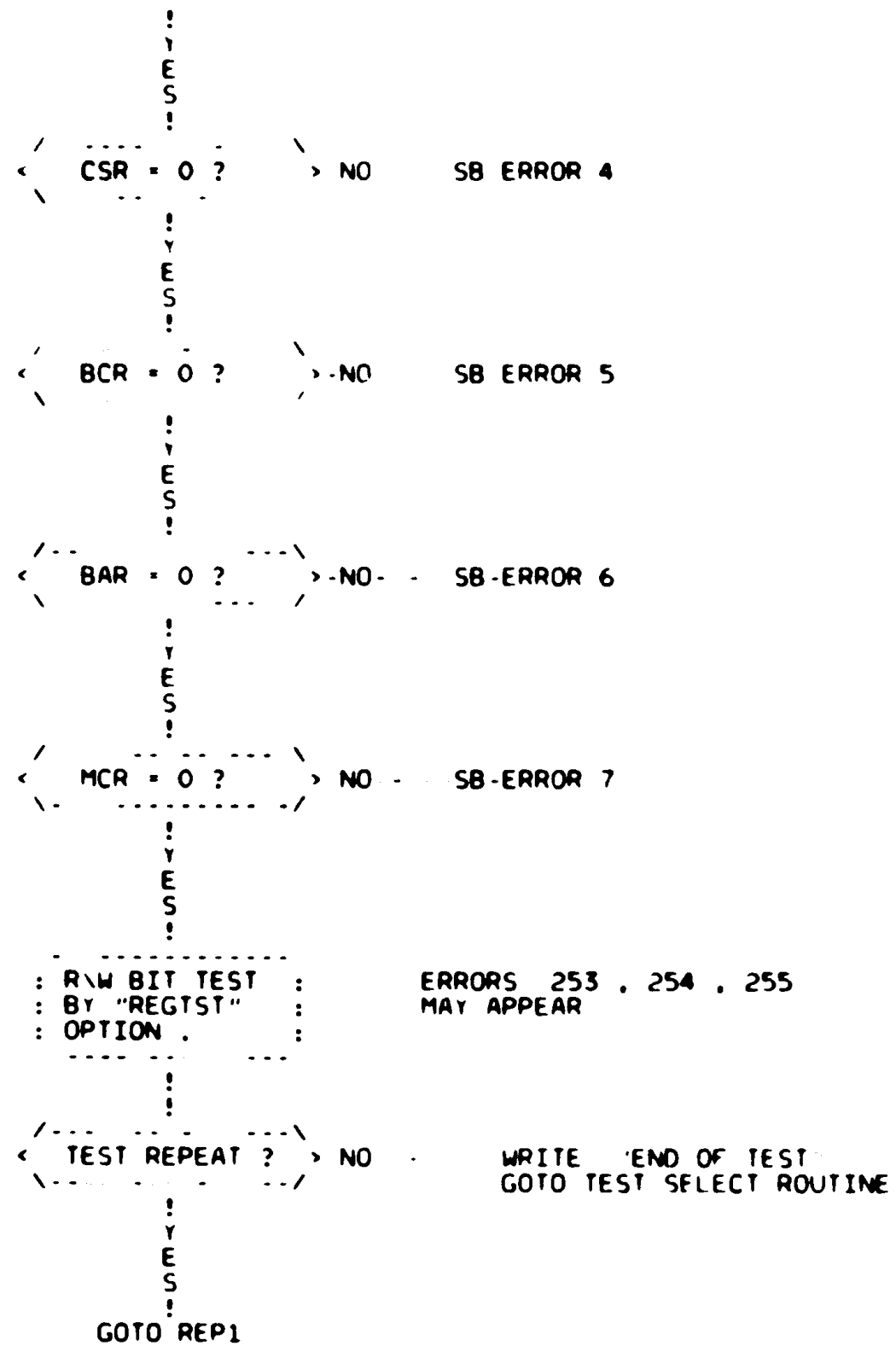
CONTENTS SHOULD BE : 000026 BUT WAS : 000025

[TEST XX] : THESE LABELS REFER TO LABELS IN THE PROGRAM
LISTING.

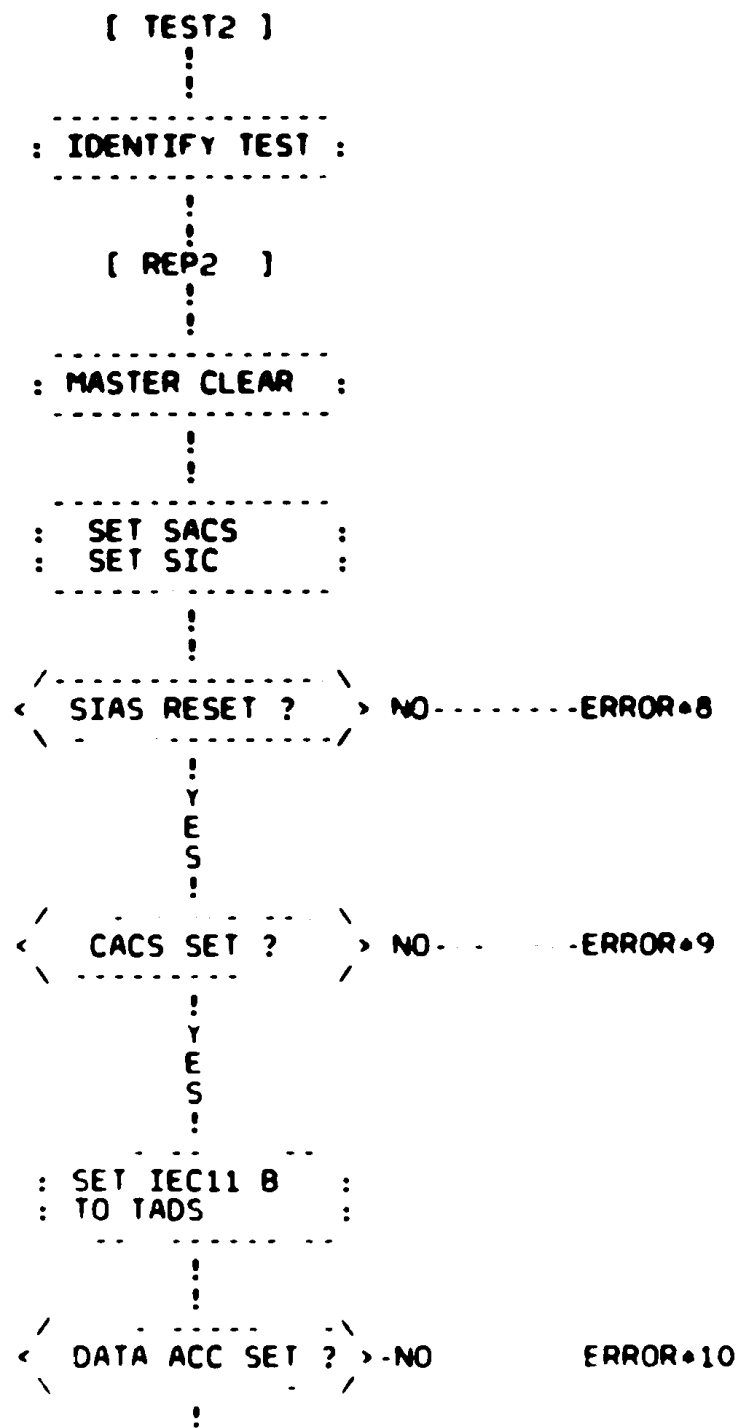
TEST 1: REGISTER STATIC TEST

THE MASTER CLEAR FUNCTION OF OF IEC11 A AND IEC11B
IS TESTED.
ALL READ/WRITE BITS IN IEC11 B ARE CHECKED WITH
ALTERNATE PATTERN (EXCEPT BIT 0 IN CSR.)





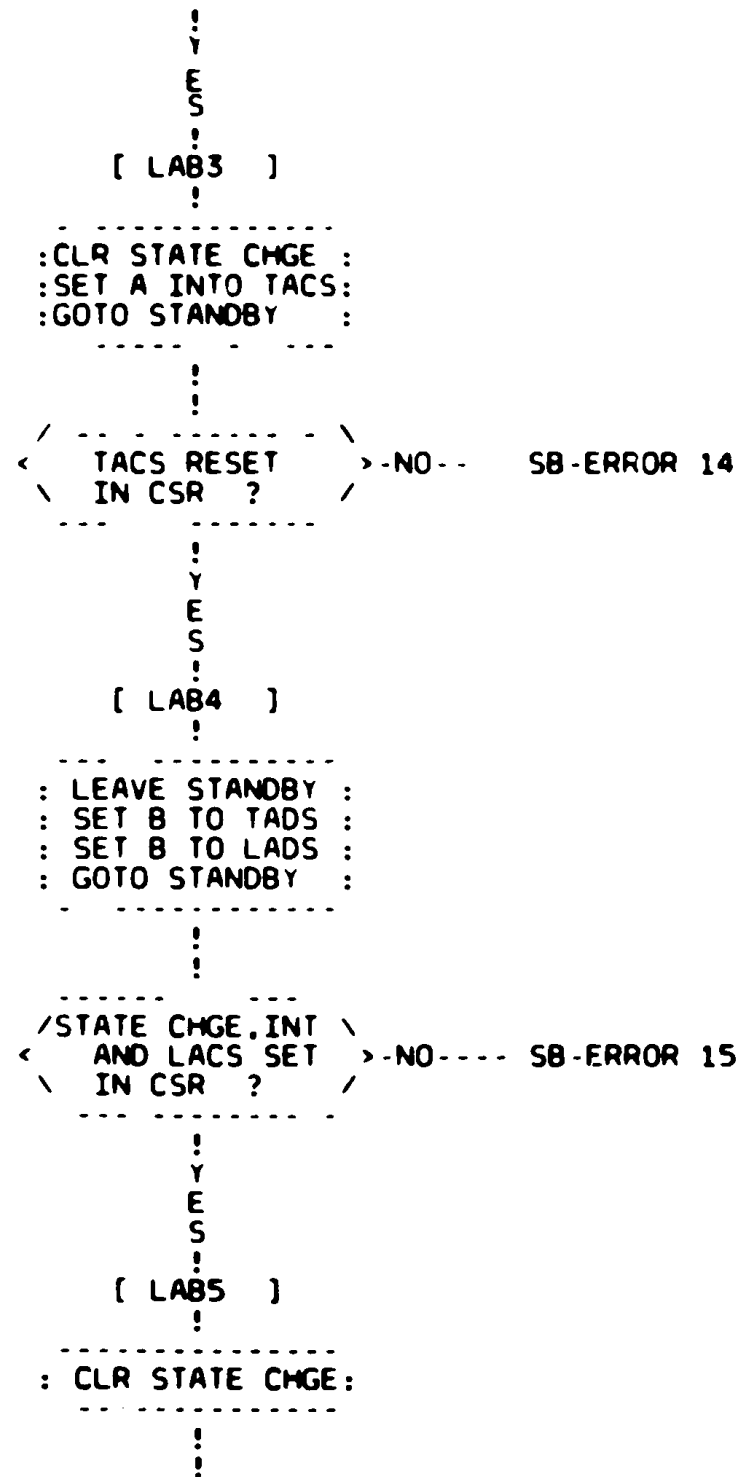
TEST 2 : TALKER AND LISTENER FUNCTION TEST

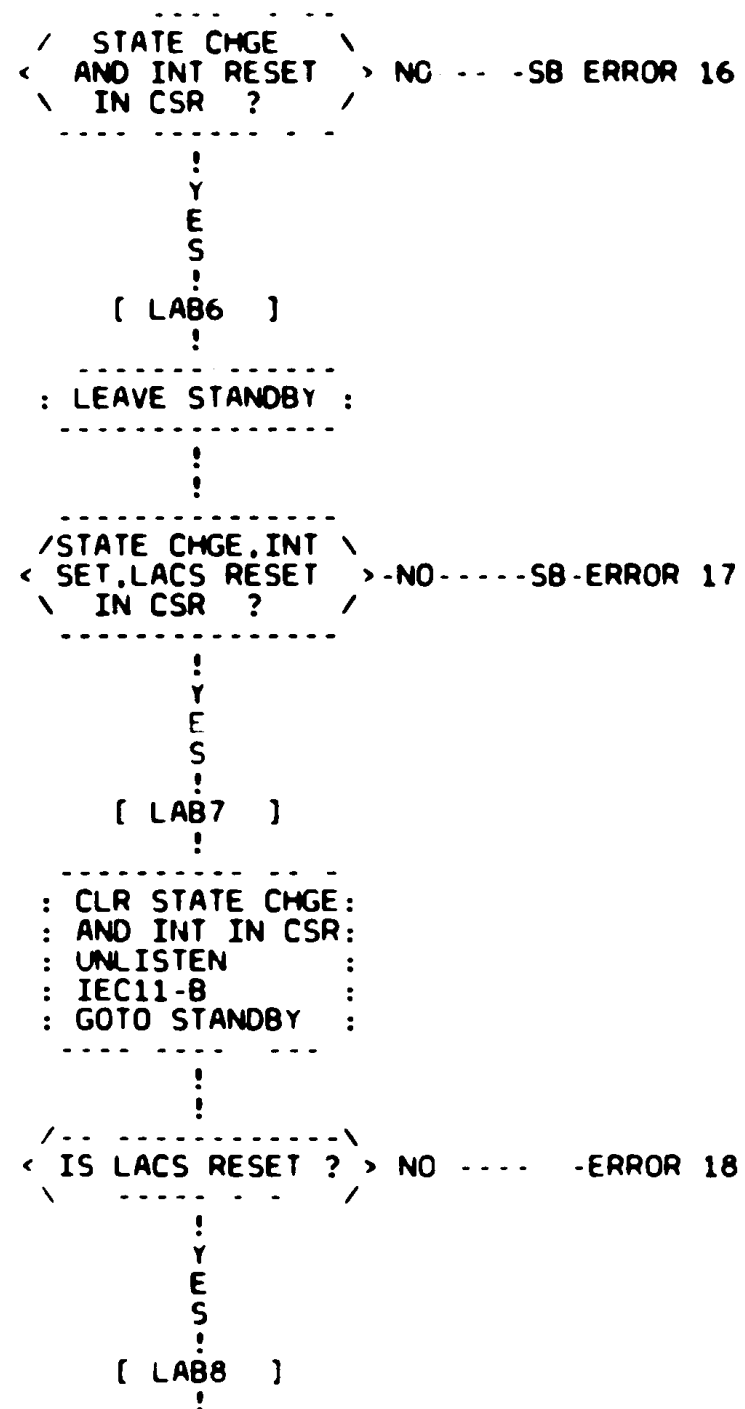


```

      YES
      :
      :
      : GOTO STANDBY :
      : STATE,SET B :
      : TO TACS :
      :
      :
      :
      /STATE CHGE,INT \
< AND TACS SET > NO--- SB ERROR 11
\ IN CSR ? /
      :
      :
      : YES
      :
      :
      [ LAB1 ]
      :
      :
      : CLEAR STATE :
      : CHGE IN CSR :
      :
      :
      :
      /STATE CHGE \
< AND INT RESET > NO- SB ERROR 12
\ IN CSR ? /
      :
      :
      : YES
      :
      :
      [ LAB2 ]
      :
      :
      : LEAVE STANDBY :
      : STATE :
      :
      :
      :
      /STATE CHGE,INT \
< SET,TACS RESET > NO-----SB ERROR 13
\ IN CSR ? /

```





```

: LEAVE STANDBY :
: SET B TO LADS :
: SET B TO TADS :
: GOTO STANDBY :
: ..... :
: :
: :
: STATE CHGE,INT \
< AND TACS SET > NO - -SB ERROR 19
\ IN CSR ? /

:
: YES
: :
[ LAB9 ]
:
< TEST REPEAT ? > -NO ..... -WRITE "END OF TEST"
\ / GOTO TEST SELECT ROUTINE

:
: YES
: :
GOTO REP2

```

.....

TEST 3 : GENERAL INTERRUPT AND DMA FUNCTION TEST

THE FUNCTION OF ALL BITS WHICH CAN AFFECT "INT" IN CSR
IS CHECKED.
THOSE BITS ARE: "STATE CHGE", "NEX", "BC OVFL", "END" IN CSR
ALSO THE BUS REQUEST LEVEL OF IEC11-B IS IDENTIFIED.
IF THE TESTS RUN SEQUENTIALLY.

```

[ TEST3 ]
:
: .....
: IDENTIFY TEST :
:
: :
[ REP3 ]
:

```

```

!
: MASTER CLEAR :
: SET A TO CACS :
: SET A TO TACS :
: SET B TO LACS :
: GOTO STANDBY :
.....
!
[STCHGE ]
!
: ENABLE INTERR.:
.....
!
< /-----\
  INTERRUPT ? > NO - - - -ERROR#20
< \-----/
!
YES
!
< /-----\
  INT ENB RESET ? > NO - - - - -ERROR 21
< \-----/
!
YES
!
< /-----\
  INTERRUPT
  CAUSED BY
  STATE CHGE ? > NO - - - -SB ERROR 22
< \-----/
!
YES
!
[NEXBIT ]
!
< /-----\
  128 K MEMORY ? > NO - - - - GOTO NEX1
< \-----/
!
YES
!

```

```

:GET NEX ADDRESS:
: FROM I/O PAGE :
: AND PUT IT TO :
: BAR           :
:             :
: ENABLE DMA    :
: AND INTERRUPT :
:             :

: BYTE TO IOR  :
:             :
: /-----\
< DATA ACCEPTED ? > NO -----ERROR#23
\-----/
:
: YES
:
: /-----\
< INTERRUPT ? > NO - --ERROR#24
\-----/
:
: YES
:
: /-----\
< INTERRUPT CAUSED BY NEX ? > NO -- SB ERROR 25
\-----/
:
: YES
:
[ NEX1 ]
:
: /-----\
< 400000 IS NON-EXISTANT MEMORY ? > NO-----GOTO NEX2
\-----/
:
: YES

```

```

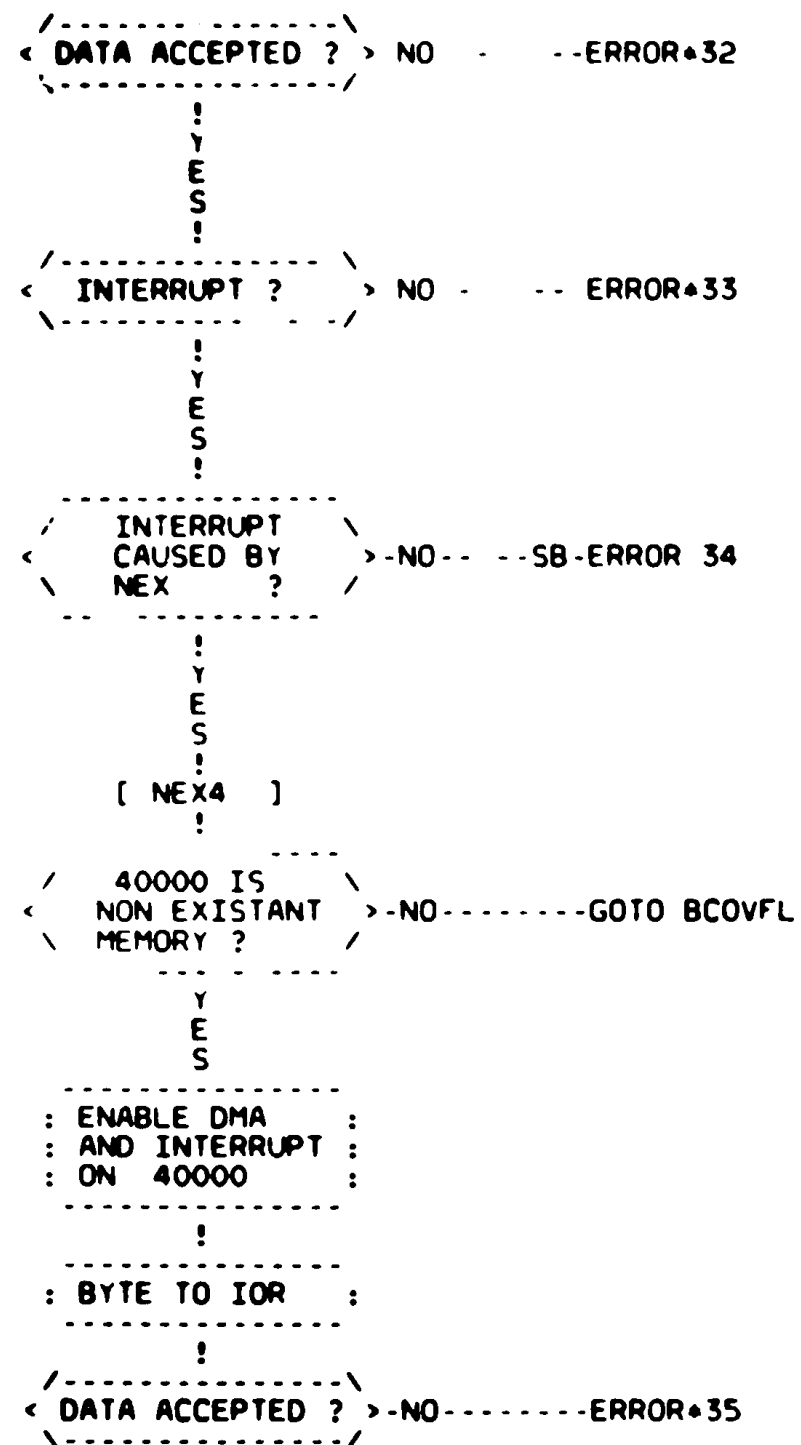
:-----
:  ENABLE DMA      :
:  AND INTERRUPT  :
:  ON 400000      :
:-----
:
:  BYTE TO IOR    :
:-----
:
< / DATA ACCEPTED ? \ > NO- -----ERROR#26
\ ----- /
:
: YES
:
< / INTERRUPT ? \ > -NO-----ERROR#27
\ ----- /
:
: YES
:
< / INTERRUPT
  CAUSED BY
  NEX ? \ > -NO-----SB-ERROR 28
\ ----- /
:
: YES
:
[ NEX2 ]
:
< / 200000 IS
  NON-EXISTANT \ > -NO ----- GOTO NEX3
  MEMORY ? /
\ ----- /
:
: YES
:
:-----
:  ENABLE DMA      :
:  AND INTERRUPT  :
:  ON 200000      :
:-----

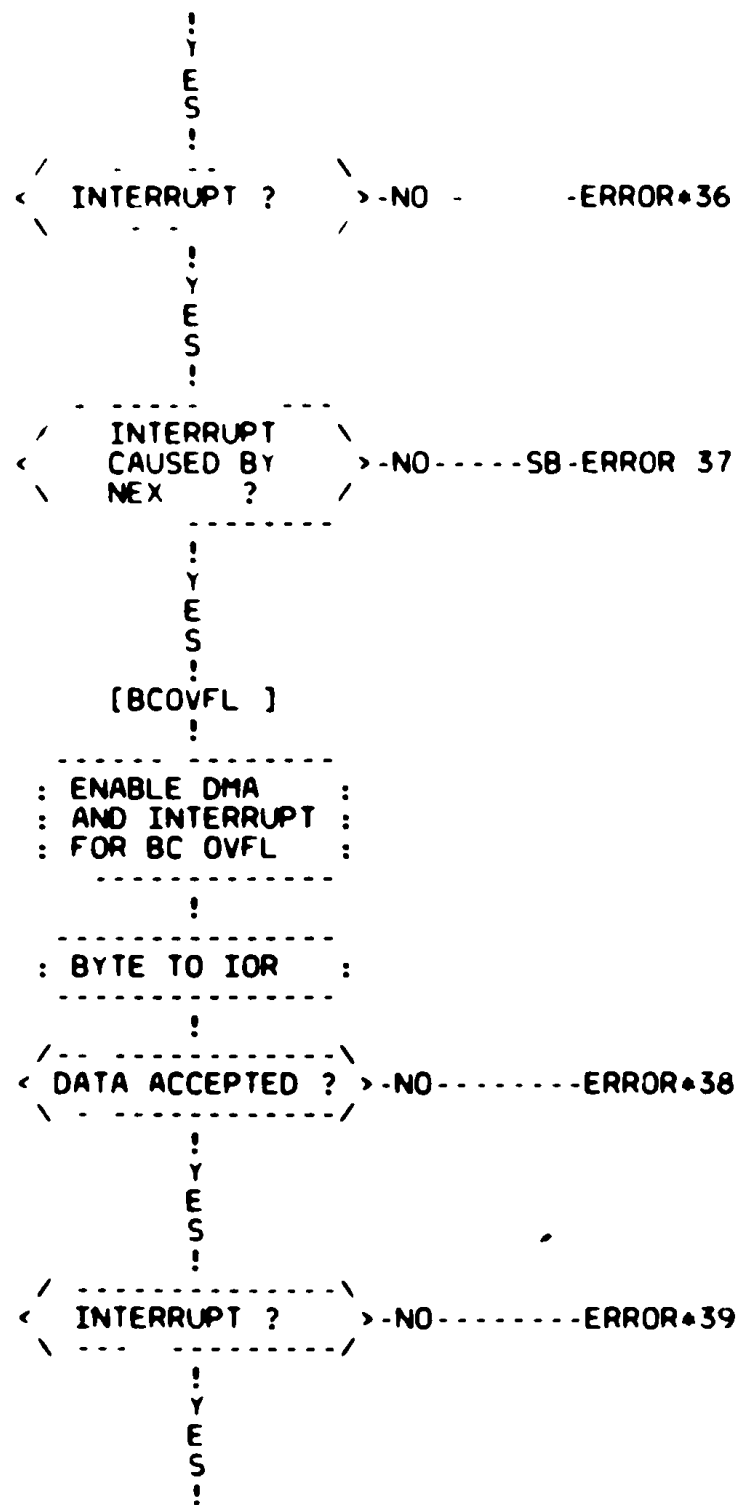
```

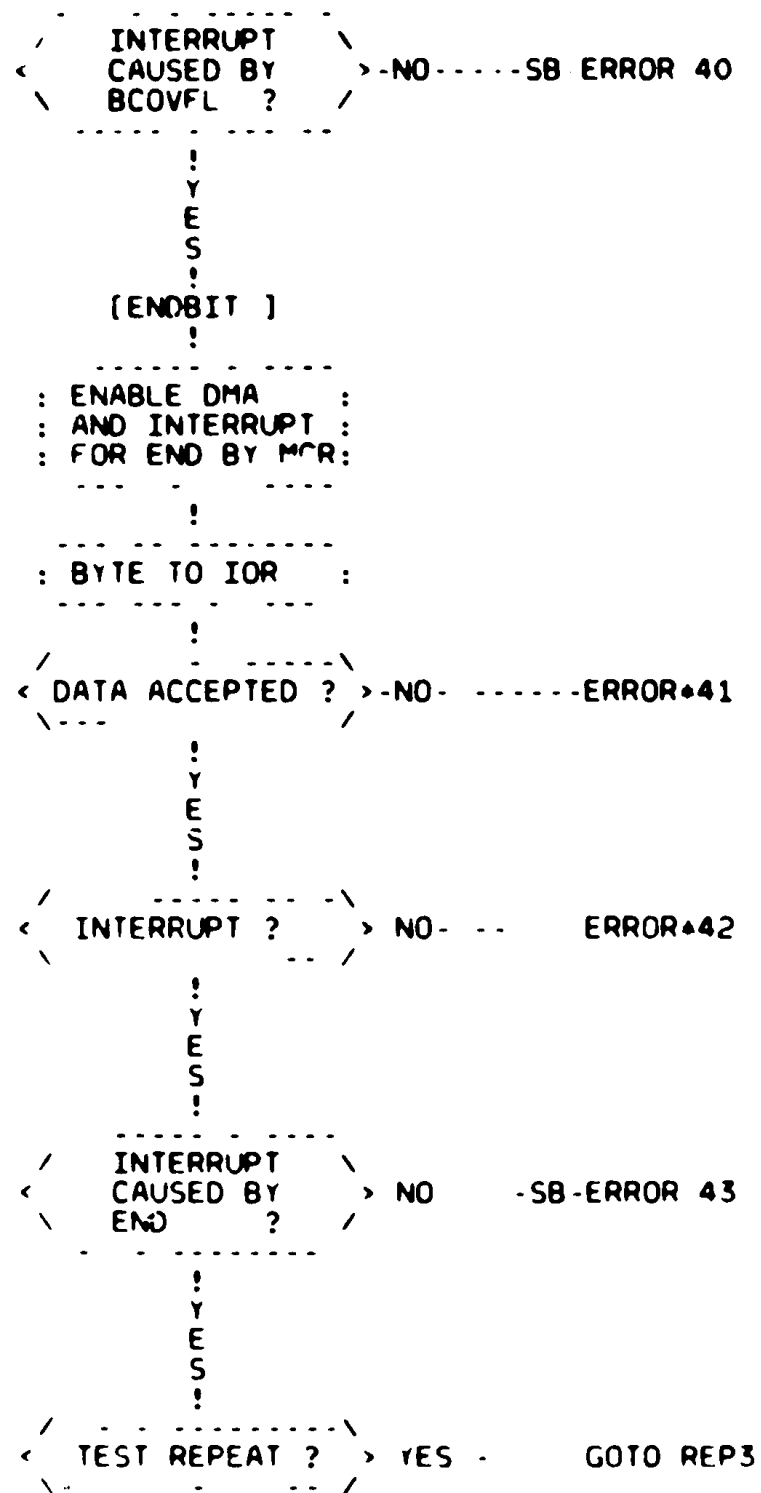
```

!
: BYTE TO IOR :
!
< DATA ACCEPTED ? > -NO----- -ERROR#29
\
!
YES
!
< INTERRUPT ? > NO - - -ERROR#30
\
!
YES
!
< INTERRUPT CAUSED BY NEX ? > -NO-- - SB ERROR 31
\
!
YES
!
[ NEX3 ]
!
< 100000 IS NON EXISTANT MEMORY ? > -NO----- --GOTO NEX4
\
!
YES
!
: ENABLE DMA :
: AND INTERRUPT :
: ON 100000 :
!
: BYTE TO IOR :
!

```







```

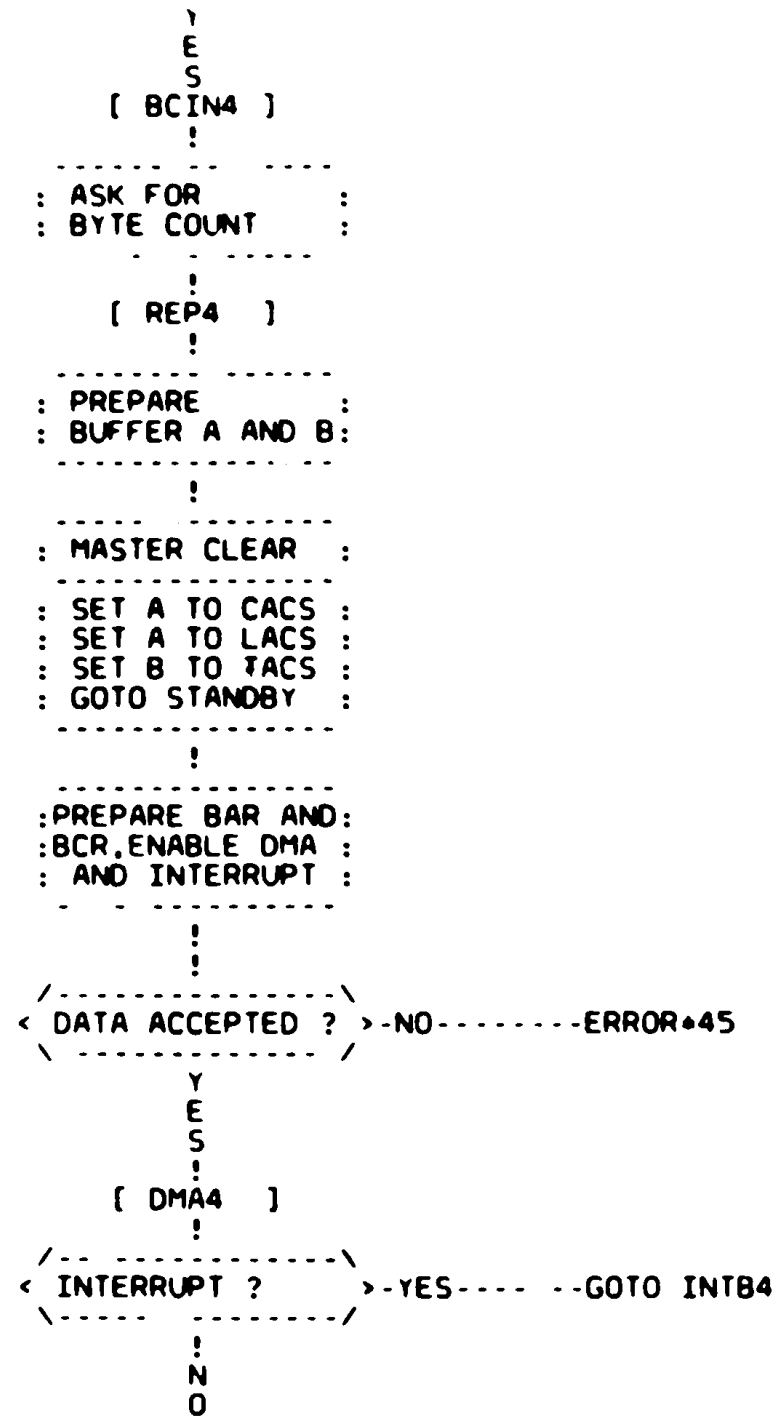
/ INTERRUPT ON \
< ANY BUS REQUEST > NO      ERROR-44
\ LEVEL ?      /

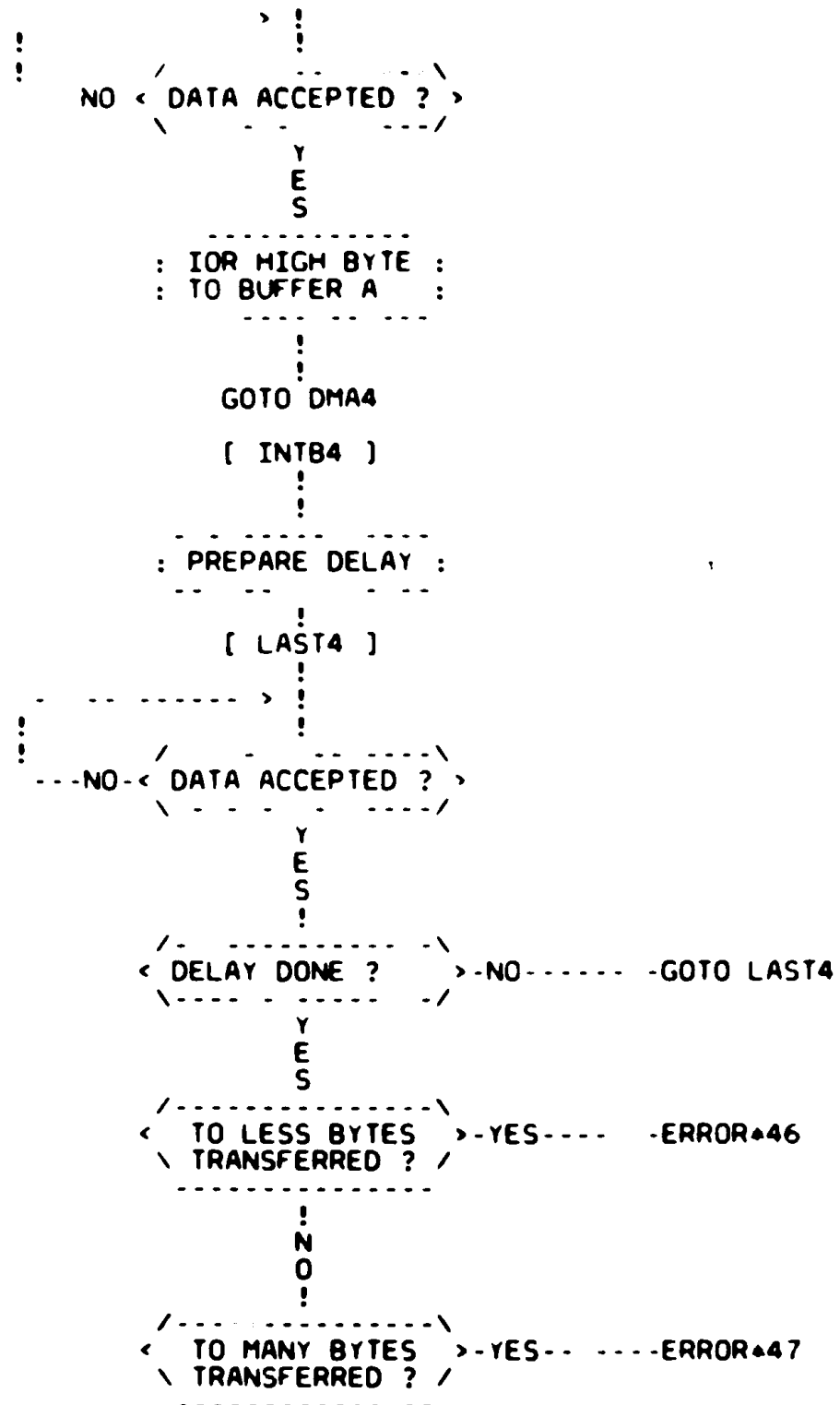
```

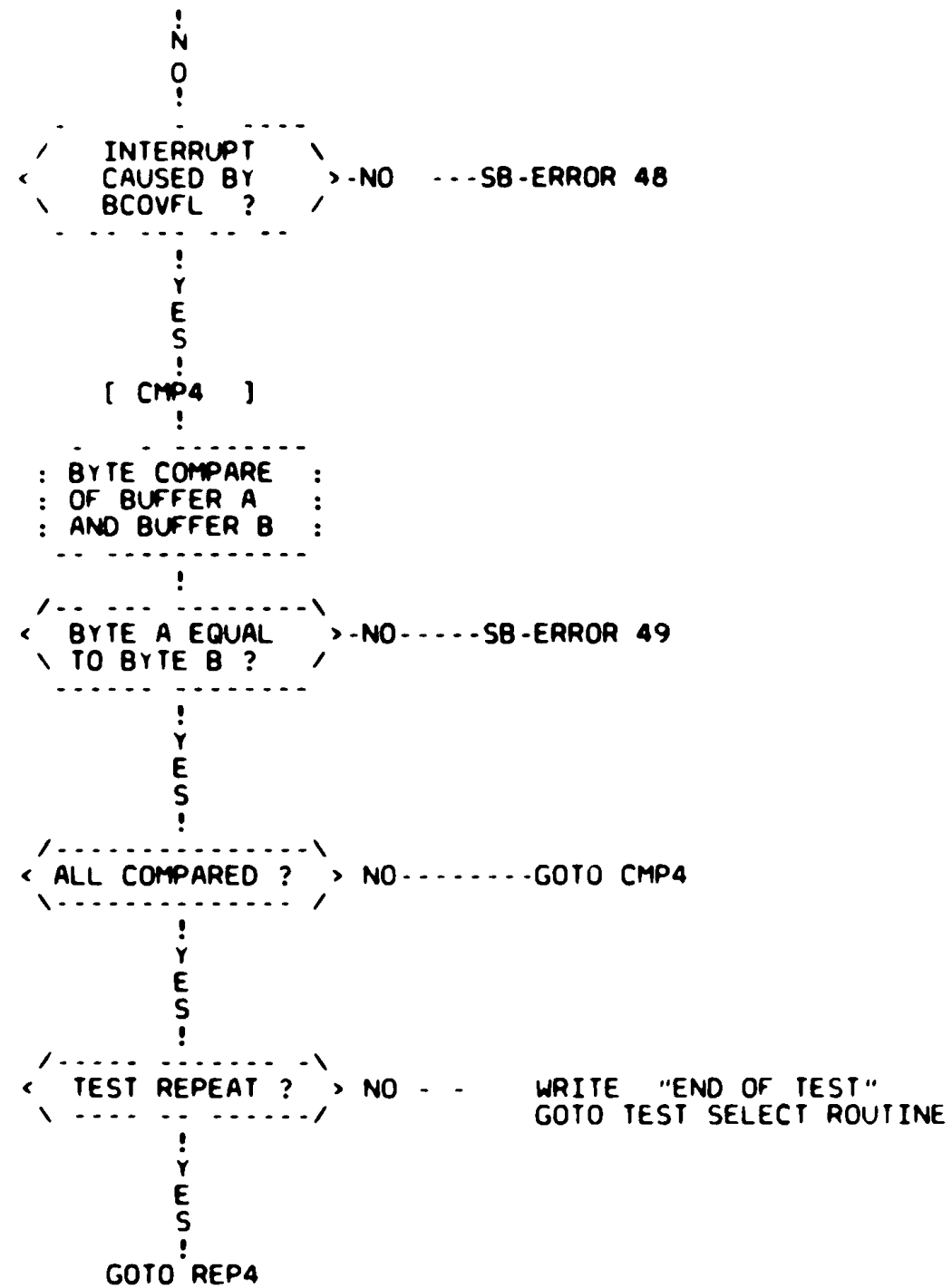
```

[ TEST4 ]
  !
  -----
: IDENTIFY TEST :
  -----
  !
/-----\
< TEST REPEAT ? > -NO      - -USE MAXIMAL BYTE COUNT
\-----/                GOTO REP4

```





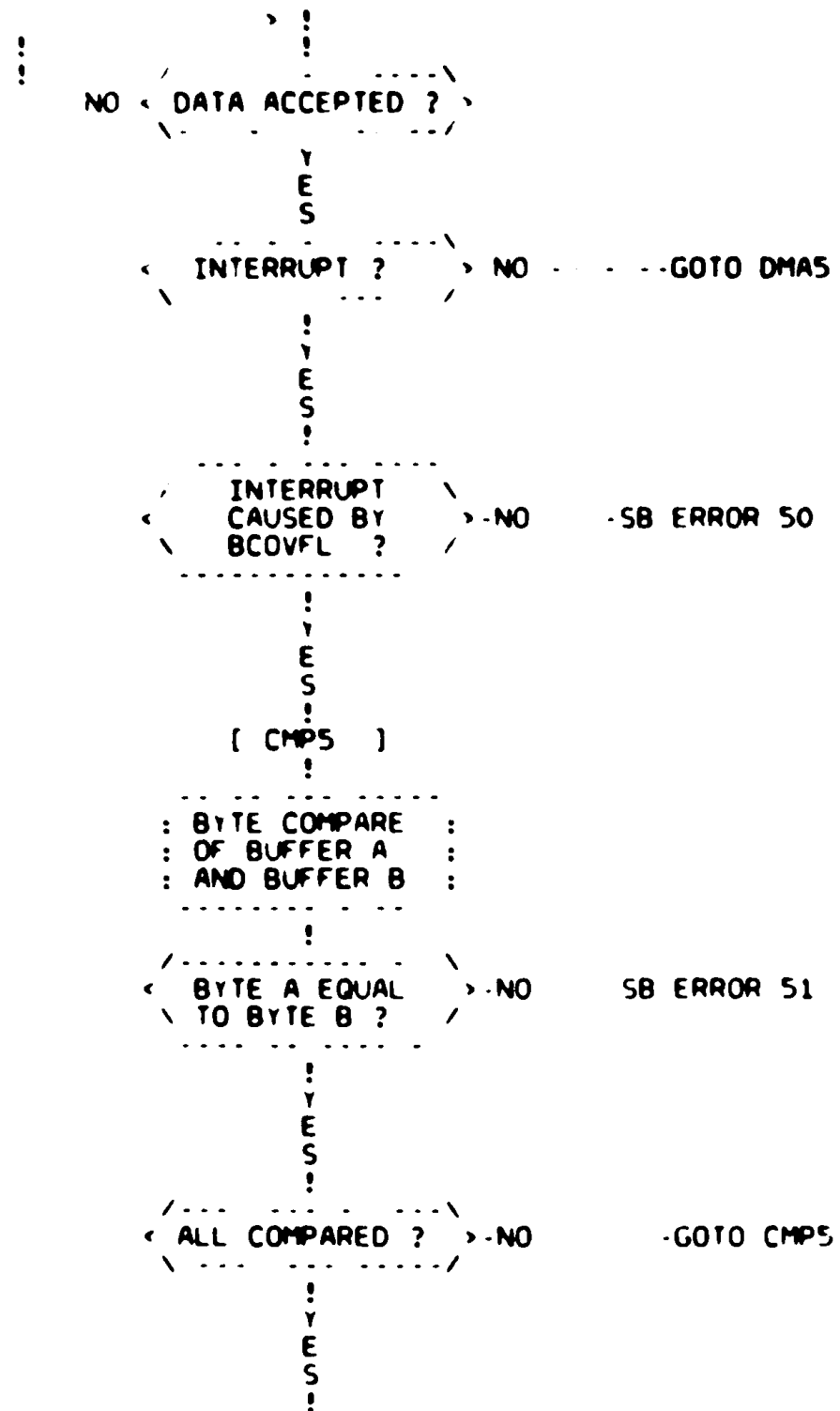


TEST 5 : DMA-TRANSFER FROM A TO B (B IS LISTENER)

```

      [ TEST5 ]
      !
      : IDENTIFY TEST :
      : -----
      : !
      < /-----\
      < TEST REPEAT ? > NO-----USE MAXIMAL BYTE COUNT
      < \-----/      GOTO REPS
      : Y
      : E
      : S
      [ BCINS ]
      !
      : ASK FOR
      : BYTE COUNT :
      : -----
      : !
      [ REPS ]
      !
      : PREPARE
      : BUFFER A AND B:
      : -----
      : !
      : MASTER CLEAR :
      : -----
      : SET A TO CACS :
      : SET A TO TACS :
      : SET B TO LACS :
      : GOTO STANDBY :
      : -----
      : !
      : PREPARE BAR AND:
      : BCR,ENABLE DMA :
      : AND INTERRUPT :
      : -----
      : !
      [ DMAS ]
      !
      : BYTE TO IOR
      : FROM BUFFER A :
      : -----

```



```

      /-----\
    < TEST REPEAT ? > NO
      \-----/
        |
        | YES
        |
        |
      GOTO REPS

```

```

      WRITE "END OF TEST"
      GOTO TEST SELECT ROUTINE

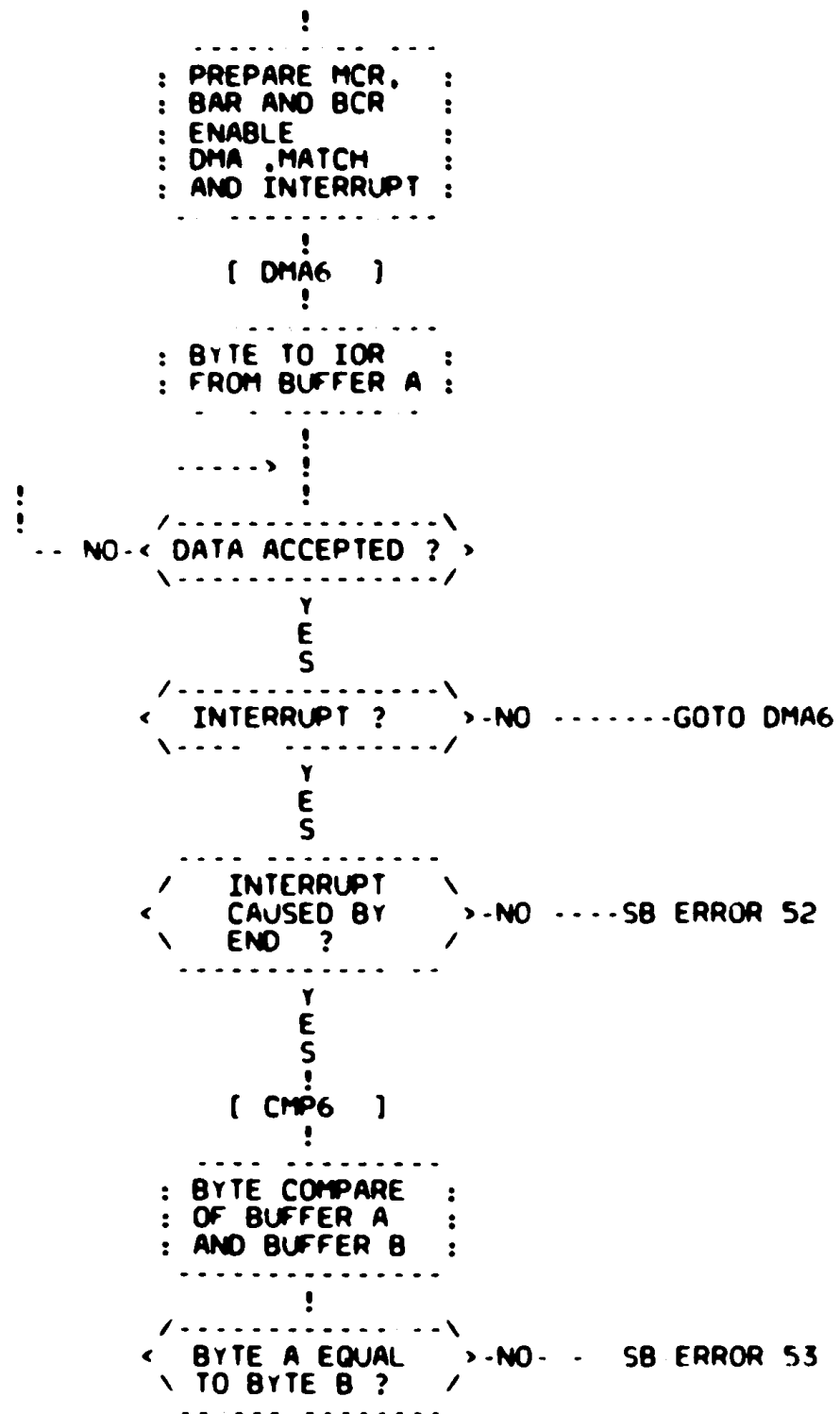
```

***** TEST 6 : MATCH CHARACTER REGISTER TEST (B IS LISTENER)

```

[ TEST6 ]
:
:
:
: IDENTIFY TEST :
:
:
: MASTER CLEAR :
:
: SET A TO CACS :
: SET A TO TACS :
: SET B TO LACS :
: GOTO STANDBY :
:
:
[ REP6 ]
:
:
: BUILD FIRST :
: MATCH COUNT :
: AND CHARACTER :
:
:
[ BEG6 ]
:
:
: PREPARE :
: BUFFER A WITH :
: ONE CHARACTER :
: LIMITED BY :
: MATCH COUNT :

```



```

      YES
      S
< ALL COMPARED ? > NO - GOTO CMP6
      YES
      S
< ALL
  CHARACTERS
  DONE ? > NO----- -GET NEXT CHARACTER
                                GOTO BEG6
      YES
      S
: GET FIRST :
: CHARACTER :
:           :
:           !
< ALL
  MATCH COUNTS > NO----- GET NEXT COUNT
                                GOTO BEG6
      YES
      S
< TEST REPEAT ? > NO--- ----WRITE "END OF TEST
                                GOTO TEST SELECT ROUTINE
      !
      YES
      S
      !
GOTO REP6

```

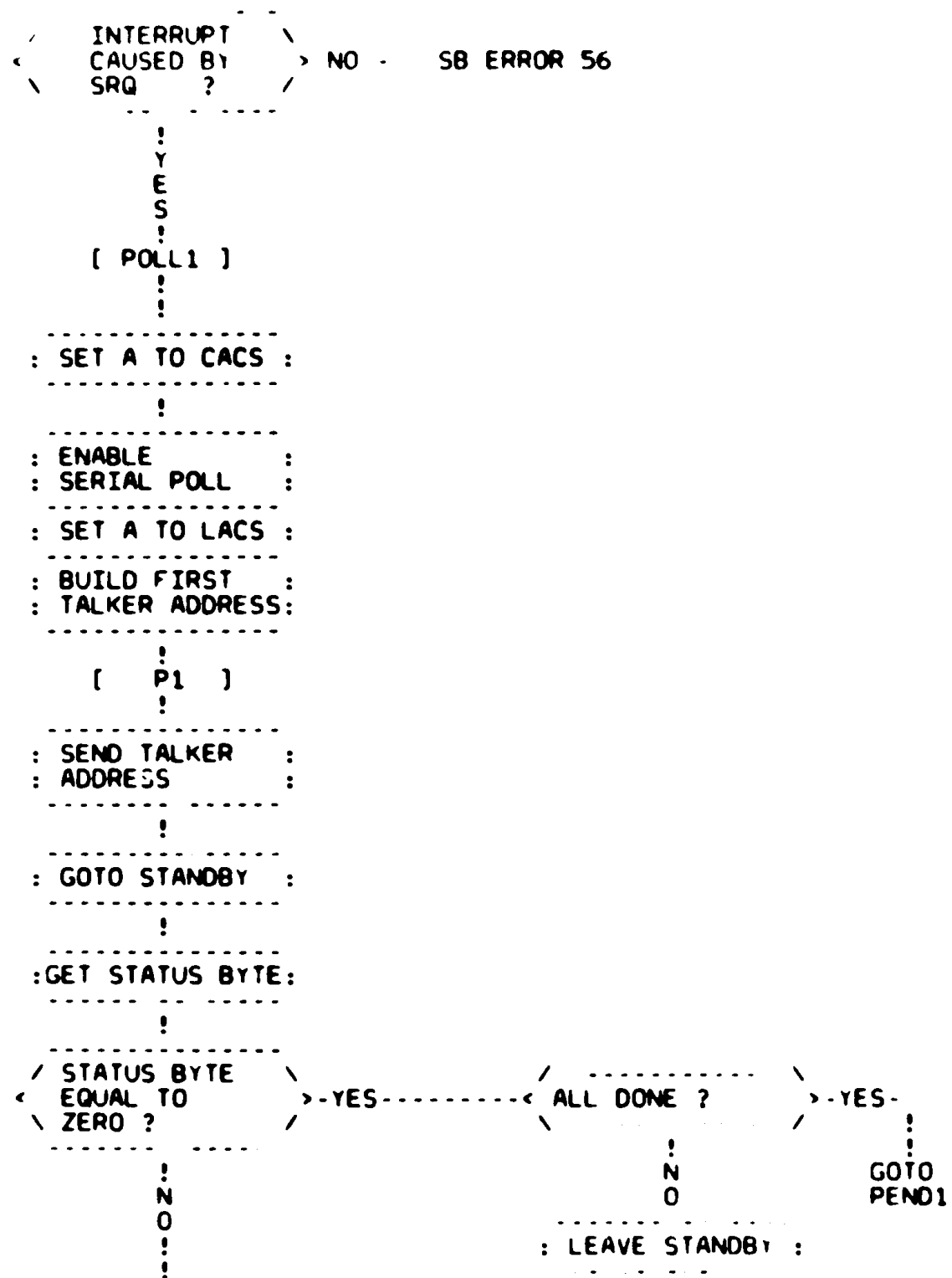
TEST 7 : SERIAL POLL PROCEDURE TEST

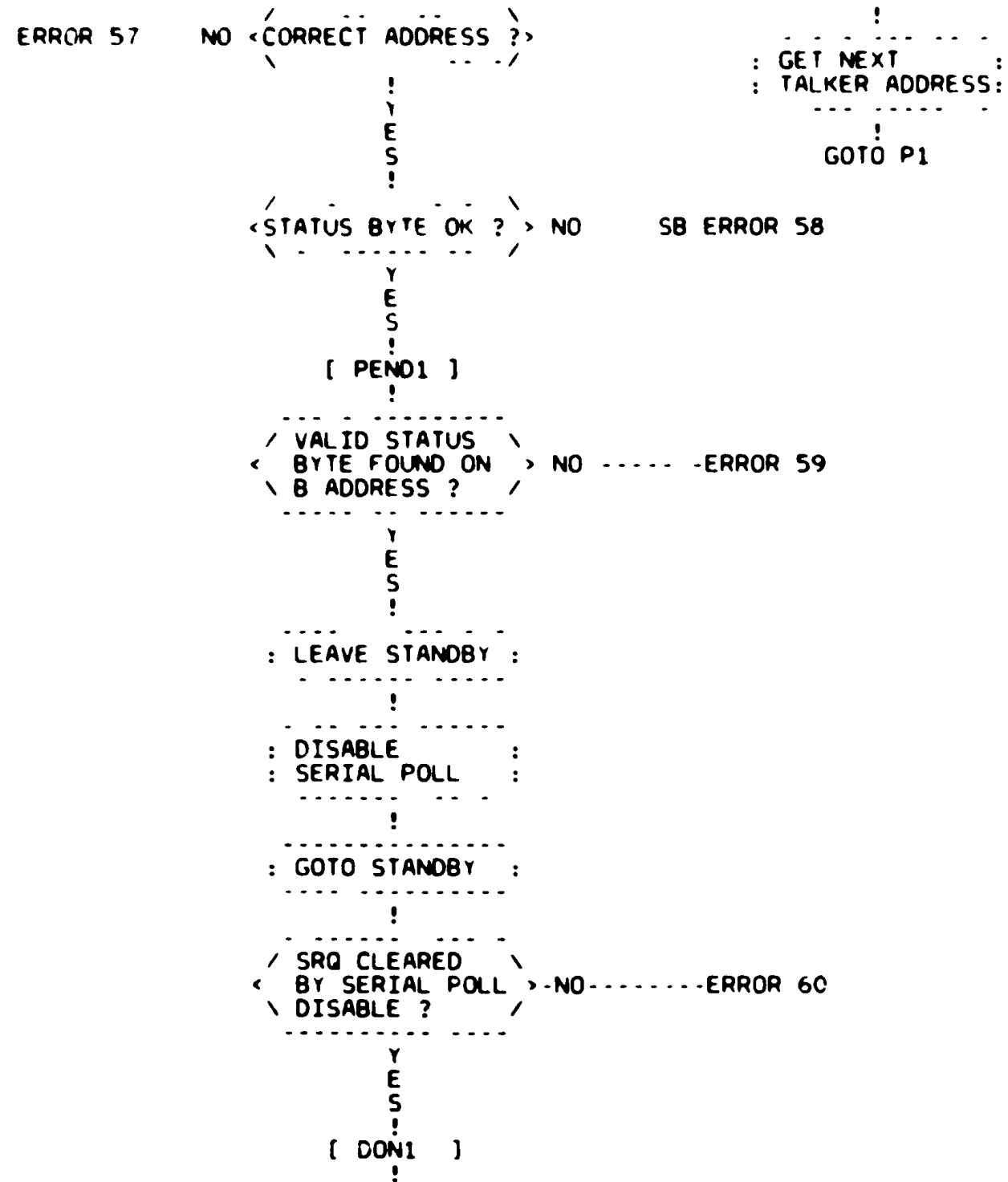
```
[ TEST7 ]
|
: IDENTIFY TEST :
|
[ REP7 ]
|
: MASTER CLEAR :
|
[ POLLO ]
|
: SET A TO CACS :
|
: ENABLE      :
: SERIAL POLL :
|
: SET A TO LACS :
|
: BUILD FIRST :
: TALKER ADDRESS:
|
[ PO ]
|
: SEND TALKER :
: ADDRESS     :
|
: GOTO STANDBY :
|
: GET STATUS BYTE:
|
```

```

/ STATUS BYTE \
< EQUAL TO    > NO-----ERROR 54
\ ZERO ?      /
-----
      Y
      E
      S
< ALL DONE ? > NO-----: LEAVE STANDBY :
\           /
      !
      Y
      E
      S
      !
: LEAVE STANDBY :
-----
      !
: DISABLE      :
: SERIAL POLL  :
-----
      !
: GOTO STANDBY :
-----
      !
[ DONO ]
      !
: REQUEST SERVICE:
-----
      !
: ENABLE      :
: INTERRUPT A  :
-----
      !
< INTERRUPT A ? > NO----- ERROR*55
\           /
      !
      Y
      E
      S
      !

```

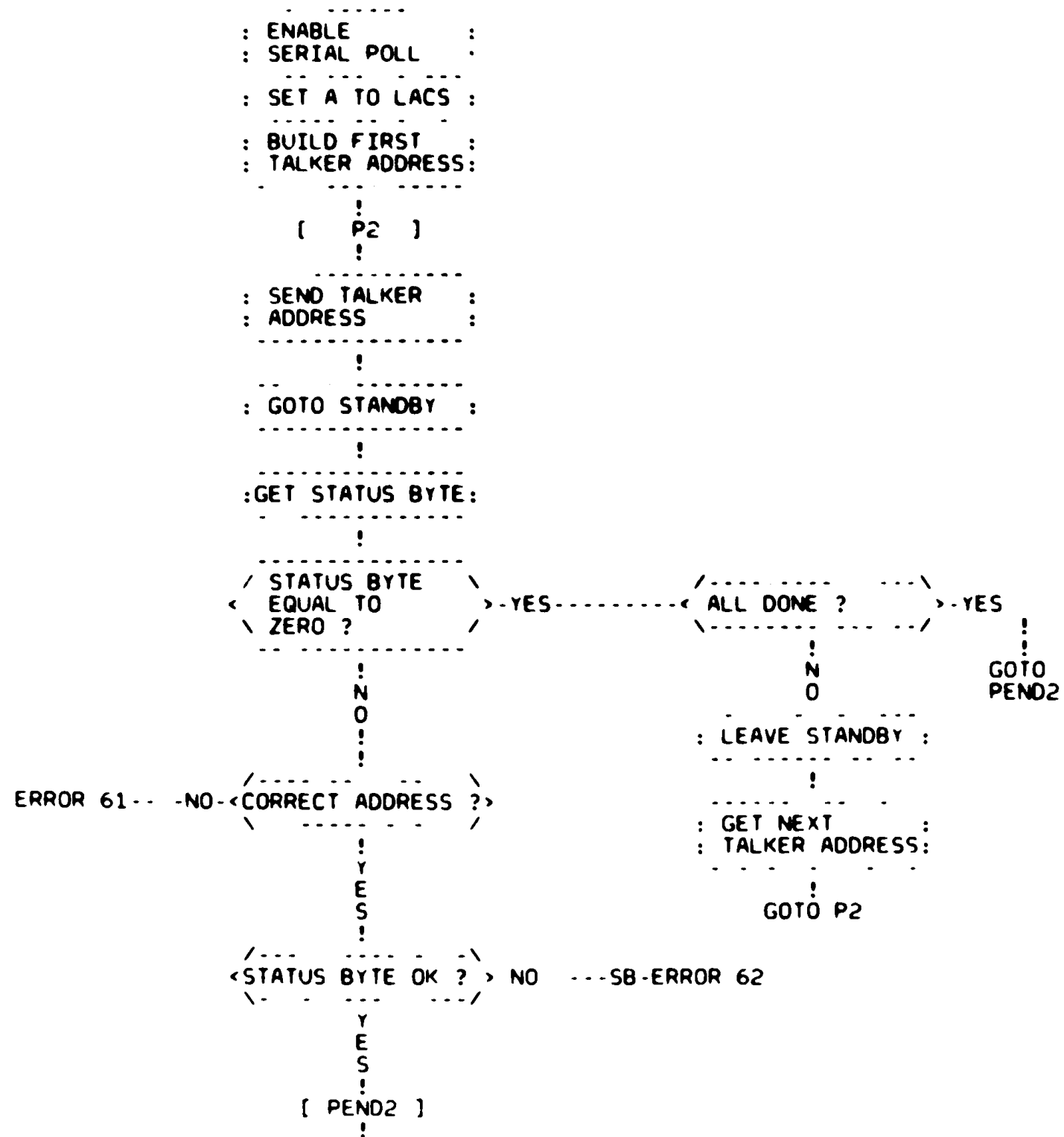




```

      : MASTER CLEAR :
      -----
      !
      : SET A TO CACS :
      -----
      : SET A TO TACS :
      : SET B TO LACS :
      -----
      !
      : GOTO STANDBY :
      -----
      [ SPOVFL ]
      |
      : ENABLE DMA :
      : AND INTERRUPT :
      : FOR BC OVFL :
      -----
      !
      : BYTE TO IOR :
      -----
      |
      - - - - -> |
                  |
      /-----\
-NO-< INTERRUPT ? >-
      \-----/
          Y
          E
          S
      : REQUEST SERVICE :
      -----
      [ POLL2 ]
      |
      : SET A TO CACS :
      -----
      !
      : UNTALK IEC11-A :
      -----

```



```

/ VALID STATUS \
< BYTE FOUND ON > -NO ..... -ERROR 63
\ B ADDRESS ? /
-----
Y
E
S
!
: LEAVE STANDBY :
-----
!
: DISABLE      :
: SERIAL POLL  :
-----
!
: GOTO STANDBY :
-----
!
[ DON2 ]
!
: MASTER CLEAR :
-----
!
: SET A TO CACS :
-----
: SET A TO TACS :
: SET B TO LACS :
-----
!
: GOTO STANDBY :
-----
!
[ SPEND ]
!
: ENABLE DMA    :
: AND INTERRUPT :
: FOR END      :
-----
!
: BYTE TO IOR   :
-----
!
```

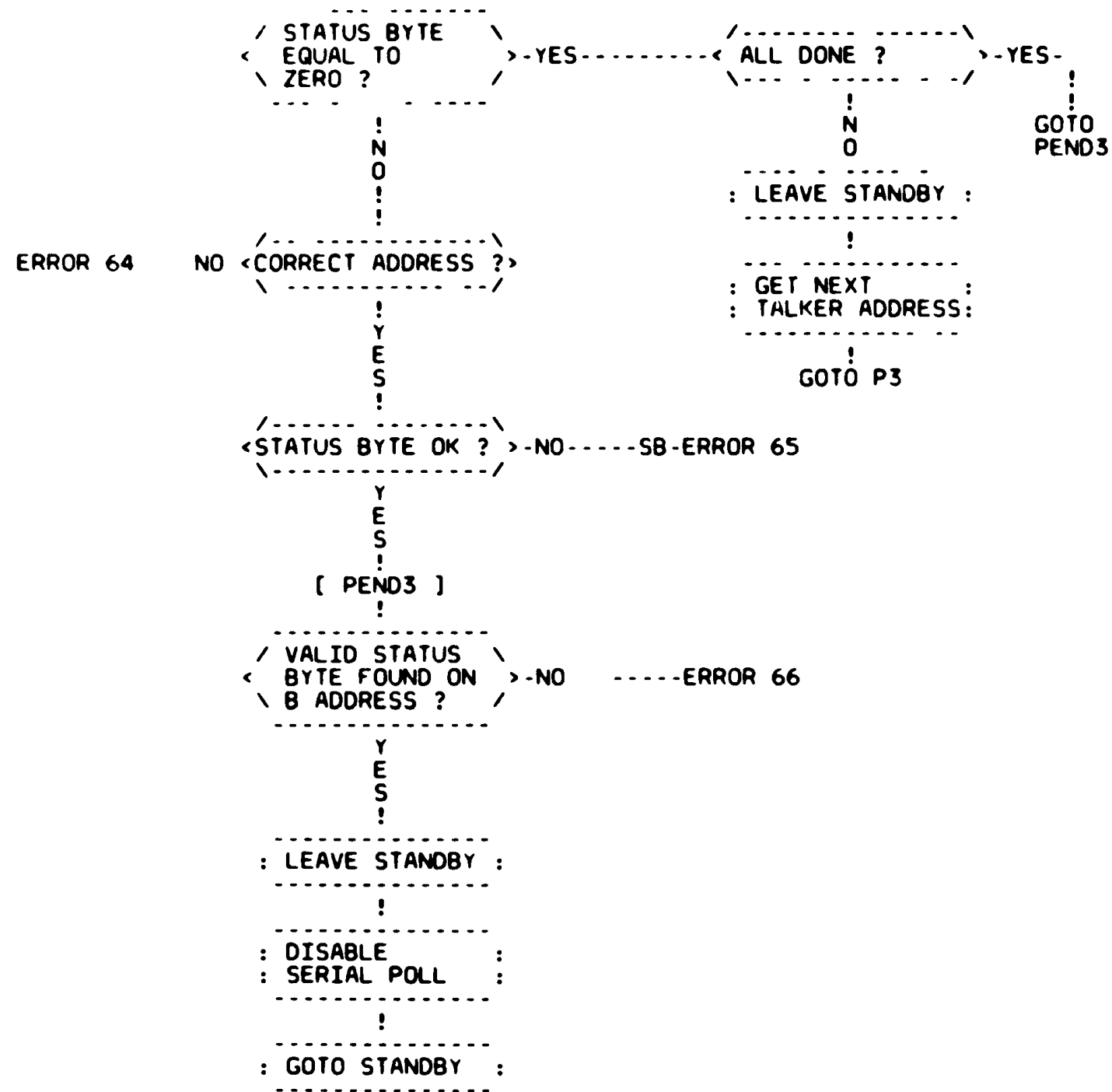
```

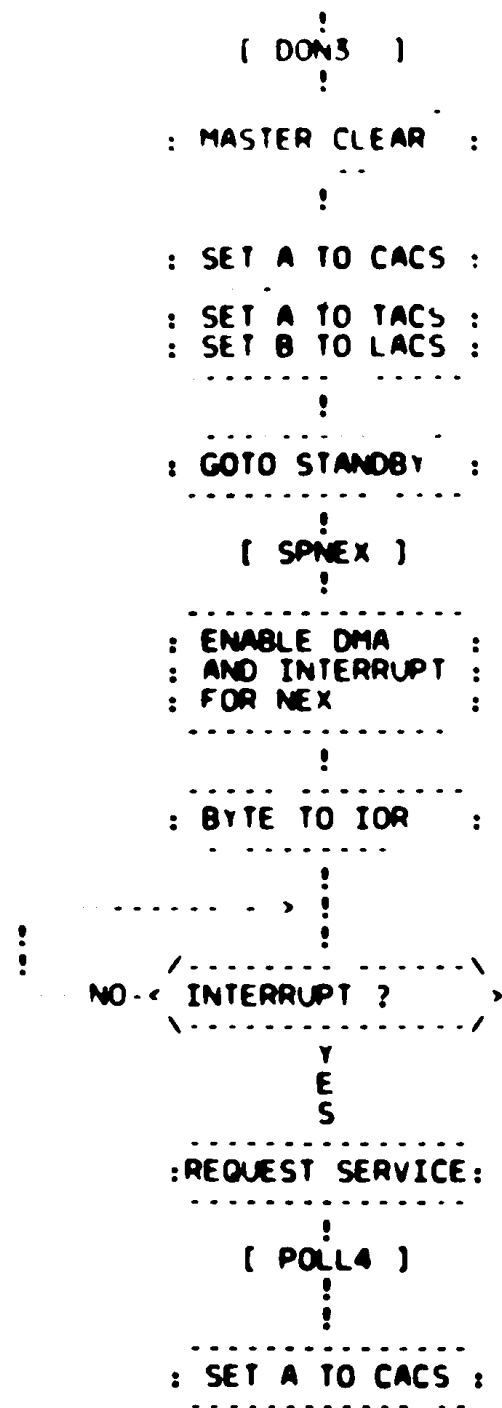
      > !
      | !
--NO < -.- INTERRUPT ? -.- >
      |
      E
      S
    : REQUEST SERVICE:
      |
      [ POLL3 ]
      |
    : SET A TO CACS :
      |
    : UNTALK IEC11-A:
      |
    : ENABLE          :
    : SERIAL POLL     :
    : SET A TO LACS   :
    : BUILD FIRST     :
    : TALKER ADDRESS:
      |
      [ P3 ]
      |
    : SEND TALKER     :
    : ADDRESS          :
      |
    : GOTO STANDBY    :
      |
    : GET STATUS BYTE:

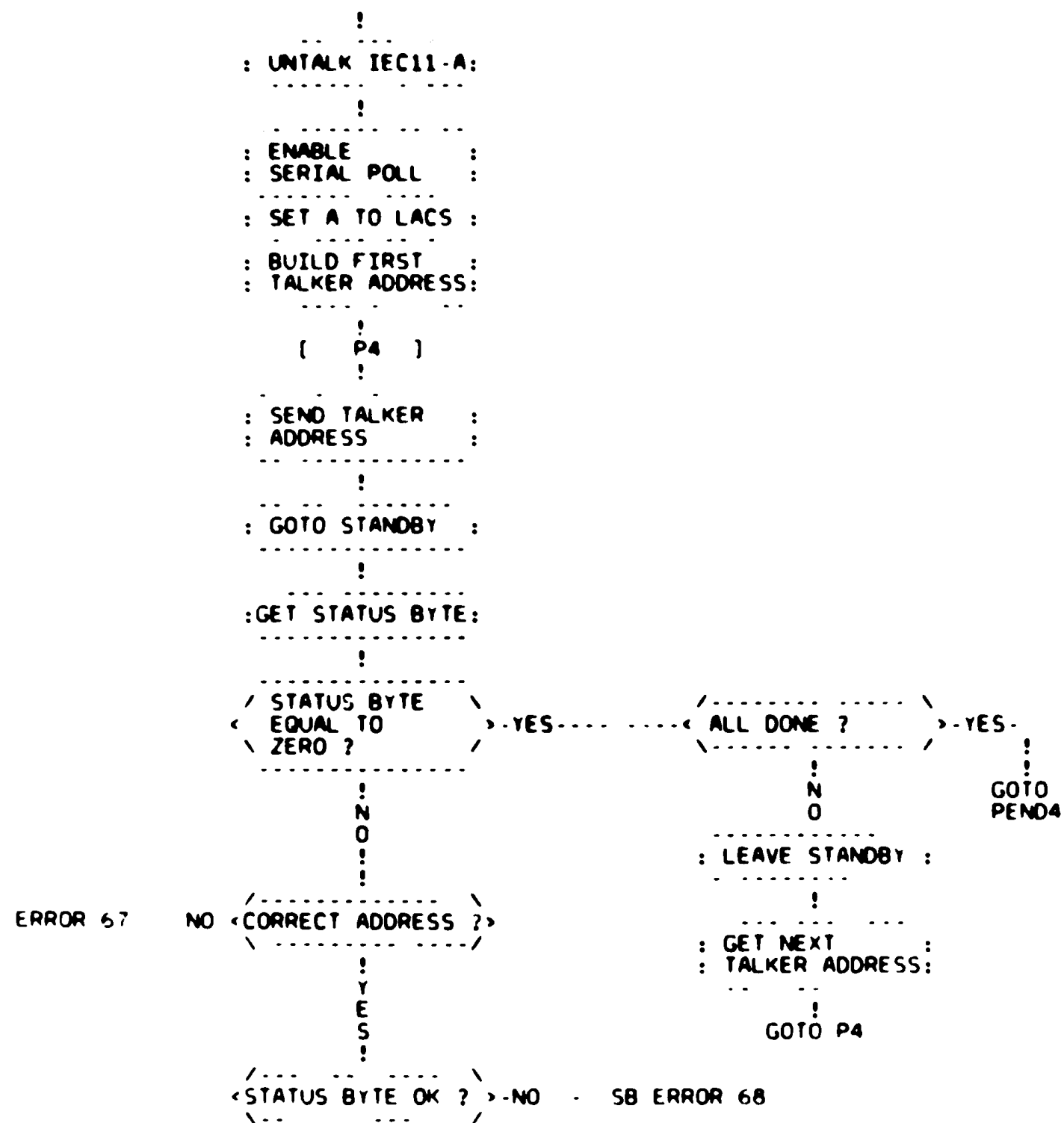
```

N3

SEQ 39



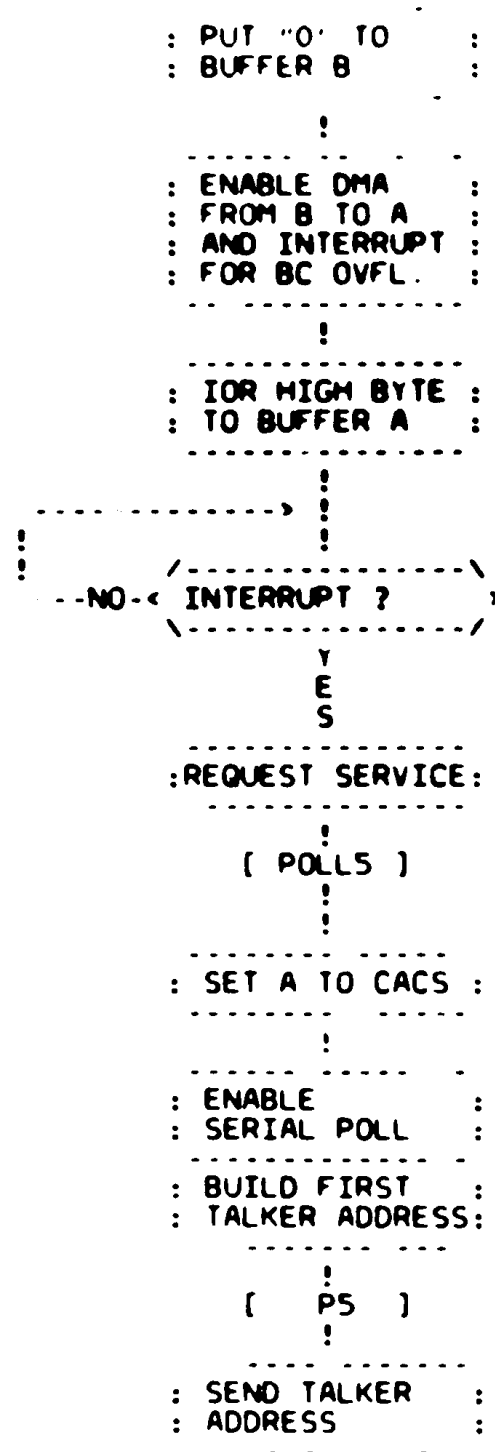


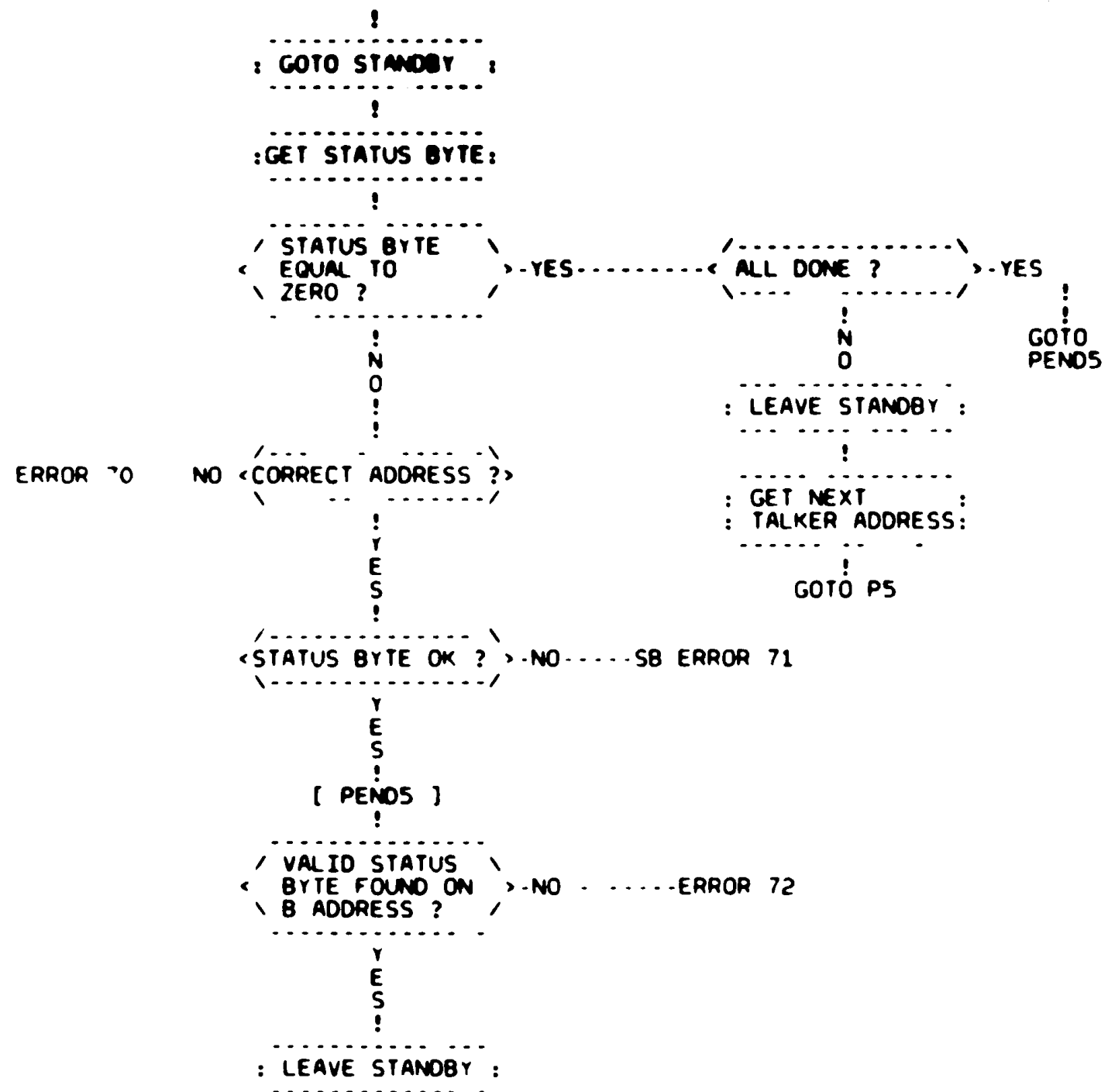


```

      Y
      E
      S
      !
    [ PEND4 ]
      !
      -
    / VALID STATUS \
    < BYTE FOUND ON > NO      ERROR 69
    \ B ADDRESS ? /
      -
      Y
      E
      S
      !
    : LEAVE STANDBY :
      -
      !
    : DISABLE      :
    : SERIAL POLL  :
    :             :
      -
      !
    : GOTO STANDBY :
      -
      !
    [ DON4 ]
      !
    : MASTER CLEAR :
      -
      !
    : SET A TO CACS :
    : SET B TO TACS :
    : SET A TO LACS :
      -
      !
    : GOTO STANDBY :
      -
      !
    [ SP8TOA ]
      !

```





```

      !
      :-----:
      : DISABLE :
      : SERIAL POLL :
      :-----:
      !
      :-----:
      : GOTO STANDBY :
      :-----:
      !
      [ DONS ]
      !
      !
      /-----\
      < TEST REPEAT ? > -NO -----WRITE "END OF TEST"
      \-----/          GOTO TEST SELECT ROUTINE
      !
      Y
      E
      S
      !
      GOTO REP7

```

10.0 DIAGNOSTIC MONITOR DESCRIPTION

LOADING PROCEDURE

BOOT XXDP MEDIA, THEN RUN DIAGNOSTIC WITH FOLLOWING
 COMMAND:

R ZIEC??

NOTE: ?? REFERS TO VERSION AND REV LEVEL.

STARTING PROCEDURE

A) USING A PROCESSOR WITH A SWITCH PANEL

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

PRESS "LOAD ADDRESS" SWITCH

PRESS "START" SWITCH

PROGRAM SHOULD BEGIN TO PRINT OUT THE IDENTIFICATION-PRINTOUT.

B) USING A LSI11

TYPE IN START ADDRESS, "G" INTO ONE LINE.

PRESS "RETURN" KEY.

PROGRAM STARTS WITH IDENTIFICATION PRINTOUT.

C) USING A PROCESSOR WITHOUT A CONSOLE

PRESS "CNTRL" AND "BOOT" BUTTONS. ODT CONSOLE

ROUTINE PRINTS A \$ SIGN.

TYPE IN "L 1000" AND A <CR>.

TYPE AFTER THE '\$' AN "S". PROGRAM SHOULD BEGIN TO PRINT IDENTIFICATION PRINTOUT.

OPERATING PROCEDURES

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

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

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

WHEN AN ERROR OCCURS, THE FOLLOWING TEXT IS PRINTED:

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

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

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

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

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

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

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

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

```
SYNTAX ERROR !
*****
```

YOU CAN NOW REPEAT THE INPUT.

ANY COMBINATION OF OPTIONS IS LEGAL, BUT PROBABLY NOT USEFUL.
WHEN YOU TYPE /SC/NS THE FIRST OPTION IS 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 INPUTED 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 PRINTOUT 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

AND A RETURN. THE MONITOR WILL PRINT AT THE BEGINNING OF
THE NEXT LINE THE CONTENT OF LOCATION 1000:

CDM>DMP/001000

012767

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

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

CDM>

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

CDM>SET/1000/000000

AND PRESS THE "RETURN" KEY. 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
INTERRUPTED 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

COM>
....

TYPE NOW ON THE SAME LINE

COM>ABO
....

AND THE PROGRAM WILL GO TO BEGIN OF TEST OR TO A
ROUTINE WHERE YOU CAN SELECT A SUBTEST .

11.0 LISTING
.....

2	4	LOW CORE
4	53	INITIALISATION
5	103	MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
5	215	\$.MAP MAPPING ROUTINE
7	267	\$.EMT
8	329	\$.TRP
9	399	\$.IOT
10	430	\$.PWR
10	482	\$.RSV
11	508	\$.RRS
12	536	\$.KBI
13	590	\$.INP
14	661	\$.IAY
15	712	\$.IAA
16	759	\$.IAD
18	873	\$.IAE
19-	979	\$.RED
20-	1162	\$.WRT
21	1230	\$.KBO
22-	1252	\$.PRO
23-	1285	\$.STX
24-	1317	\$.DMP
26	1418	\$.ADP
27-	1437	\$.TOT
28-	1463	\$.BUF
29-	2	LOCAL MACRO DEFINITIONS
30-	77	INPUT PROCEDURE
31	175	TEST SELECT ROUTINE
32-	209	TEST 1: REGISTER STATIC TEST
33-	268	TEST 2: TALKER AND LISTENER FUNCTION TEST
34-	379	TEST 3: GENERAL INTERRUPT AND DMA FUNCTION TEST
35-	651	TEST 4: DMA-TRANSFER FROM B TO A (B IS TALKER)
37	766	TEST 5: DMA-TRANSFER FROM A TO B (B IS LISTENER)
38	847	TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)
39	919	TEST 7: SERIAL POLL PROCEDURE TEST
40	1299	INTERRUPT SERVICE ROUTINES
41	1332	CONSTANTS, VARIABLES
44	1372	MESSAGES
45	1411	STANDARD CSS REGISTER TEST

1
2 000000
3
4
5
6 000000

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



```

47
48
49
50
51
52
53
54 001204 000005
55 001206 012706 001204
56 001212 012737 001242 000004
57 001220 005037 000006
58 001224 005037 177776
59 001230 000240
60 001232 012737 177777 007016
61 001240 000404
62 001242 005037 007016
63 001246
64 001252 012737 000006 000004
65 001260 012737 000004 000006
66 001266 012737 003424 000060
67 001274 012737 000340 000062
68 001302 012700 007026
69 001306 005020
70 001310 020027 007202
71 001314 001374
72 001316 012737 001000 007172
73
74 001324
75 001350
76 001360 012737 007026 007146
77 001366 004737 001516
78 001372 000414
79 001374
80 001374 012700 160000
81 001400 005001
82 001402 012737 001410 000004
83 001410 012706 001204
84 001414 005740
85 001416
86 001424
87 001424 010037 007022
88 001430 010137 007024
89 001434 006201
90 001436 006000
91 001440 006201
92 001442 006000
93 001444 000241
94 001446 006000
95 001450
96 001460
97 001470 052737 000100 177560
98 001476 012737 000042 000040
99 001504 005737 007152
100 001510 001375
101 001512 000137 010614

;
; MZ001
; *****
; AS VERY FIRST INSTRUCTION DO A "RESET" TO CLEAR
; ALL HARDWARE THAT IS IN AN UNDEFINED STATE
;
; .SBTTL INITIALISATION
START: RESET ;CLEAR THE WORLD ;MZ001
MOV #START,SP ;SETUP SP
MOV #8..TYP,4 ;PREP FOR TEST
CLR 6
CLR #0PS ;IS IT A PDP OR A LSI ?
NOP
MOV #1..8PTYP ;IT IS A PDP
BR TYP..E ;SKIP LSI PART
8..TYP: CLR 8PTYP ;INDICATE A LSI
STCKUP
TYP..E: MOV #6,4 ;REBUILD
MOV #4,6
MOV #8..KBI,#060 ;SETUP KEYBOARD VECTOR
MOV #340,#062
MOV #KBBUFF,R0 ;RESET ALL FLAGS
18: CLR (R0).
CMP R0,#PROMES
BNE 18
MOV #1000,SCOPAD ;IF NO SCOPE LABEL IS SET IN THE PROGRAM
;THEN GOTO START AFTER ERR-/SC
SET PS TO 0 ;MZ001
;SAY HELLO TO EVERYBODY
;SETUP INPUT-BUFFER-POINTER
;CALL MEM. TEST WITH MEM. MANAGEM. SETUP ;88001
;GO TO SAVE HIGHEST MEMORY ADDRESS ;88001
;HERE IF NO MEMORY MANAGEMENT FOUND ;88001
;SETUP MAXIMUM CORE-ADDRESS
;SETUP FOR MEMEND+2 FOR UNMAPPED PROC. ;88001
;SETUP TRAPHANDLING
;REBUILD STACKPOINTER
;TOP OF CORE?
;REBUILD TRAP TO 4 VECTOR
;HERE IF HIGHEST MEMORY ADDRESS FOUND ;88001
;SAVE HIGHEST MEMORY ADDR.
;SAVE EXTENSION BITS
;PUT ADDR16 INTO CARRY ;88001
;ROTATE ADDR16 INTO MSB OF R0 ;88001
;PUT ADDR17 INTO CARRY ;88001
;ROTATE ADDR17 INTO MSB OF R0 ;88001
;RESET CARRY ;88001
;SHIFT OUT THE LS-OCT DIGIT OF PHYS ADDR ;88001
;DUMP ADDRESS
;PRINT LS-OCT-DIGIT ALWAYS AS "6" ;88001
;ENABLE TTY INTERRUPTS
;PREP THIS LOCATION FOR USING XXDP
;SYNCHRONISE OUTPUT
;
; NOW WE ARE ON THE AIR

```

```

103 .SBTTL MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
104
105 SUBROUTINE: M..TST
106 .....
107
108 THIS ROUTINE CHECKS THE AVAILABILITY OF A MEMORY MANAGEMENT UNIT
109 IN PDP 11 PROCESSORS AND SETS KISAR0...KISAR5 TO THE LOW 24K WORDS
110 OF THE COMPUTER MEMORY.
111 KISAR7 IS USED TO MAP THE SO-CALLED I/O PAGE.
112 KISAR6 IS USED TO MAP THE HIGHEST MEMORY ADDRESS CHECK POINTER.
113
114 INPUT: (SP) *RETURN ADDRESS
115
116 OUTPUT: R0 *LOW 16 ADDRESS BITS OF HIGHEST MEMORY ADDRESS
117 R1 *BITS 0.1 ADDRESS BITS 16.17
118
119 AUTHOR: BERNHARD BAUDISCH CSS/DP MUNICH 19-DEC 78
120
121
122 M..TST:
123 001516 012737 001776 000250 MOV #M..TRP,#0250 ;REFERENCE LABEL
124 001524 005037 000252 CLR #0252 ;SETUP SEGMENTATION TRAP CATCHER
125 001530 012737 001776 000004 MOV #M..TRP,#04 ;SETUP TRAP-4 CATCHER
126 001536 005037 000006 CLR #06 ;...
127 001542 004737 002006 CALL M..MM ;SETUP DEFAULT MAPPING,ENABLE MM
128 001546 012737 007400 172354 MOV #7400,KISAR6 ;MAP WITH KISAR6 THROUGH MEMORY
129 001554 012737 000001 007020 MOV #1,MMTYP ;REMEMBER THAT WE HAVE MEMORY MANAGEMENT!
130 001562 012700 157776 MOV #157776,R0 ;GET HIGHEST ADDRESS WITHIN APR6 RANGE
131 001566 012737 001766 000004 MOV #M..MIM,#04 ;SET UP TRAP 4 CATCHER
132 001574 005710 201: TST #R0 ;SEE IF ADDRESS PRESENT
133 001576 103031 BCC M..DET ;CC IF ADDRESS FOUND
134 001600 162737 000200 172354 SUB #200,KISAR6 ;MAP TO NEXT LOWER 4K WORD BANK
135 001606 002401 BLT 301 ;THIS CASE SHOULD NEVER HAPPEN
136 001610 000771 BR 201 ;CHECK NEXT LOWER 4-K WORDS
137 001612 301: ;HERE ONLY IF MAIN PROCESSOR HARDWARE ERROR
138 001612 PSET #0 ;SET PRIORITY TO 0
139 001636 WRITE MEMDEF ;MEMORY MANAGEMENT DEFECT
140 001646 012716 001374 MOV #M..TSE,(SP) ;MODIFY RETURN ADDRESS TO UNMAPPED SYSTEM
141 001652 042737 000001 177572 BIC #BIT0,SRO ;DISABLE MM
142 001660 000422 BR M..RET ;AND GO AHEAD LIKE UNMAPPED SYSTEM
143 001662 M..DET: ;HIGHEST MEMORY ADDRESS FOUND
144 001662 PUSH KISAR6 ;GET MAP REGISTER CONTENTS
145 000005 .REPT 5 ;
146 ASL (SP) ;PUT ADDRESS BIT 17 INTO PSW CARRY
147 .ENDR ;
148 001700 103402 BCS 401 ;CS IF ADDRESS BIT 17 SET
149 001702 005001 CLR R1 ;HERE IF BIT17 NOT SET
150 001704 000402 BR 451 ;...
151 001706 012701 000001 401: MOV #BIT0,R1 ;FLAG BIT 17 FOUND
152 001712 006316 451: ASL (SP) ;GET ADDRESS BIT 16
153 001714 006101 ROL R1 ;INSERT BIT IN MEMORY EXTENSION MASK
154 001716 042700 160000 BIC #160000,R0 ;MASK OUT APR SELECTION BITS
155 001722 050016 BIS R0,(SP) ;BUILD 16 BIT BASE ADDRESS
156 001724 012600 MOV (SP),R0 ;>R0 FOR OUTPUT TO CALLER
157 001726 012737 000252 000250 M..RET: MOV #252,#0250 ;RESET SEGMENTATION TRAP CATCHER
158 001734 012737 000004 000252 MOV #4,#0252 ;...
159 001742 012737 000006 000004 MOV #6,#04 ;RESET TRAP 4 CATCHER

```

```

160 001750 012737 000004 000006      MOV    #4,B#6      ;...
161 001756 000207                    RTS    PC          ;RETURN TO CALLER
162 001760                    U..DET:  ;HERE IF MEMORY MANAGEMENT ADDRESS NOT FOUND
163 001760 012716 001374      MOV    #M..TSF,(SP)      ;MODIFY RETURN ADDRESS
164 001764 000760                    BR     M..RET      ;RETURN COMMON
165 001766                    M..MIM:  ;TRAP-4 AND SEGMENTATION TRAP CATCHER
166 001766 052766 000001 000002      BIS    #BIT0,2(SP) ;SET CARRY IN RETURN PSW SETUP
167 001774 000002                    RTI                    ;RETURN WITH CARRY SET INDICATOR
168
169 001776                    M..TRP:  ;
170 001776 022626                    CMP    (SP)+,(SP)+  ;CLEAN STACK OF INT. VALUES
171 002000 005726                    TST    (SP)+        ;CLEAN STACK FROM RETURN ADDRESS
172 002002 000137 001760                    JMP    U..DET  ;NO MEMORY MANAGEMENT UNIT AVAILABLE
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195 002006                    M..MM:   ;
196 002006                    ;
197 002014 012700 077406                    ;
198 002020 012701 000010                    ;
199 002024 012702 172300                    ;
200 002030 010022                    ;
201 002032 005301                    ;
202 002034 001375                    ;
203 002036 012702 172340                    ;
204 002042 005022                    ;
205 002044 012722 000200                    ;
206 002050 012722 000400                    ;
207 002054 012722 000600                    ;
208 002060 012722 001000                    ;
209 002064 012722 001200                    ;
210 002070 012722 001400                    ;
211 002074 012712 007600                    ;
212 002100 052737 000001 177572      BIS    #BIT0,SRO  ;ENABLE MEMORY MANAGEMENT
213 002106                    POP     R2,R1,R0          ;RESTORE REGS.
214 002114 000207      RETURN                    ;BACK TO CALLER
215
      .SBTTL  $.MAP MAPPING ROUTINE

```

SUBROUTINE M..MM

 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

```

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

```

```

; SUBROUTINE:  $.MAP
; .....
;
; THIS ROUTINE IS USED TO MAP WITH APR6 TO THE PHYSICAL MEMORY
; ABOVE 28. K AND COMPARE DMA READ AND WRITE DATA
;
; INPUT:  (SP)  =RETURN ADDRESS
;         2(SP) =100 OCTAL BYTE BLOCK OFFSET
;         4(SP) =MAPPING POINTER
;
; OUTPUT: KERNEL APR6 IS MAPPED TO BUFFER START ADDRESS
;         MAPPING POINTER IS SET TO BASE ADDRESS OF APR6=140000
;         CARRY IS SET IF NO MEMORY AVAILABLE
;         BREAKPOINT TRAP, IF YOU TRY TO MAP TO CDM (0 4.K)
;
; AUTHOR: ALBERT BREM CSS MUC 16-JAN 79
;
$.MAP: PUSH  R0,R1          ;SAVE REGISTERS
        IF 6(SP) GE #200 GOTO 10; ;DONT MAP TO SYSTEM AREA
        BPT              ;DONT MAP TO SYSTEM AREA
        JMP 2;           ;LEAVE
10;:    CALL M..MM        ;ENABLE MM,SETUP DEFAULT MAPPING
        MOV MEMEND,R0     ;SAVE HIGHEST MEMORY LOCATION
        .REPT 6
        ROR R0            ;COMPUTE 100 OCTAL BYTE BLOCKS
        .ENDR
        BIC #176000,R0    ;"
        MOV MEMEND+2,R1   ;SAVE MEMORY EXTENSION
        .REPT 10
        ASL R1            ;COMPUTE ADDITIONAL BLOCKS
        .ENDR
        ADD R0,R1         ;TOTAL BLOCKS OF 32.WORDS MEMORY
        BIC #BIT0,B#PS    ;CLEAR CARRY
        CMP R1,6(SP)      ;IS MEMORY AVAILABLE?
        BGE 1;           ;YES
        PREAD R0          ;SAVE CURRENT PSW
        BIS #BIT0,R0      ;SET CARRY FOR ERROR INDICATION
        PSET R0           ;"
        BR 2;            ;DONT TOUCH APR6
11;:    MOV 6(SP),KISAR6   ;MAP TO BUFFER AREA
21;:    POP R1,R0         ;RESTORE REGISTERS
        MOV (SP)+,(SP)    ;CLEAN STACK OF ARGUMENT
        RETURN           ;BACK TO CALLER
;
;
;

```

```

267          .SBTTL  $.EMT
268          ;
269          ; SUBROUTINE:  $.EMT
270          ; .....
271          ;
272          ; THIS ROUTINE HANDLES ALL EMT-TRAPS THAT MIGHT OCCUR
273          ; IN A DIAGNOSTIC PROGRAM, AND DIRECTS THE PROGRAM TO
274          ; CONTINUE AT THE APPROPRIATE FUNCTION SUBROUTINES.
275          ;
276          ; INPUT:      2(SP) = PS
277          ;              (SP) = PC
278          ; OUTPUT:     NONE
279          ;
280          ; AUTHOR:     BERT HUBER CSS/DP MUNICH 14 JULI 76
281          ;
282          ;
283          ; .ENABL LSB
284          $.EMT: MOV    @-2, -(SP)      ; REBUILD EMT-ADDR.
285          ADD     2(SP), (SP)          ;
286          MOV     @8(SP), (SP)         ; GET EMT-CODE
287          BIC     @177400, (SP)        ; EXTRACT EMT-NR.
288          ASL     (SP)                 ; CONVERT TO TABLE-OFFSET
289          CMP     (SP), @EMTTND-EMT    ; TBL-2; VALID EMT-NR.?
290          BLOS    1$                   ; IF LOS, YES
291          WRITE   MILEMT                ; ILLEGAL EMT
292          DUMP    OCT, 2(SP)           ;
293          JMP     @01000                ; RESTART PROGRAM
294          1$: ADD    @EMTTBL, (SP)      ; GET FUCTION CODE ADDR.
295          MOV     @8(SP), (SP)         ;
296          JMP     @8(SP)               ; ENTER FUCTION SUBROUTINE
297          EMTTBL: EMT0
298                  EMT1
299                  EMT2
300          EMTTND:
301          ;
302          ; SUBROUTINE TO SAVE SCOPE-LOOP ADDR.
303          ;
304          EMT0: MOV    (SP), SCOPAD     ; SAVE ADDR. OF SCOPE LOOP
305          RTI                          ; RETURN TO CALLER
306          ;
307          ; SUBROUTINE TO SETUP A VECTOR
308          ;
309          EMT1: MOV     @10(SP), -(SP)  ; GET POINTER TO VECTOR ADDR.
310          MOV     10(SP), @8(SP)       ; SETUP VECTOR
311          ADD     @2, (SP)              ; POINT TO NEW PS
312          MOV     6(SP), @8(SP)        ; SETUP NEW PS
313          MOV     (SP)+, 4(SP)          ; CLEAN UP STACK
314          MOV     (SP)+, 4(SP)          ;
315          TST     (SP)                 ;
316          RTI                          ; RETURN TO CALLER
317          ;
318          ; SUBROUTINE TO RESET A VECTOR
319          ;
320          EMT2: MOV     @4(SP), -(SP)   ; GET POINTER TO VECTOR A
321          MOV     (SP), -(SP)          ; DUPLICATE IT
322          ADD     @2, (SP)              ; BUILD ADDRESS OF VECTOR
323          MOV     (SP)+, @8(SP)        ; RESET VECTOR

```

G5

IEC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 7 1
\$..END

SEQ 58

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

```

329          .SBTTL 1..TRP
330
331      ; SUBROUTINE: 1..TRP
332      ; .....
333
334      ; THIS ROUTINE HANDLES ALL TRAPS TO LOC 34 THAT MIGHT
335      ; OCCUR IN A CSS DIAGNOSTIC PROGRAM.
336      ; AS THE TRAP INSTRUCTION IN THIS DIAGNOSTICS IS ONLY USED TO
337      ; INDICATE AN ERROR CONDITION, THIS TRAP HANDLER IS A (PSEUDO-)
338      ; SWITCH REGISTER CONTROLLED ERROR HANDLER. FOR THE DEFINITION
339      ; OF ALL AVAILABLE 'SWITCH'-OPTIONS, REFER TO SUBROUTINE "1..IAE"
340      ; LATER IN THIS LISTING.
341
342      ; INPUT:          2(SP)  *      PS
343      ;                  (SP)  *      PC
344      ; LOW ORDER BYTE OF TRAP INSTRUCTION * ERROR NUMBER
345
346      ; OUTPUT:         DEPENDING ON OPTIONS SELECTED
347      ; AUTHOR:         BERT HUBER, CSS/DP MUNICH 14 JUL 76
348
349      ; .PSECT CSSMON
350      ; ENABL  LSB
351      1..TRP: MOV  # 2, (SP)          ; REBUILD TRAP ADDRESS
352              ADD  2(SP), (SP)        ;
353              MOV  # 0(SP), (SP)      ; GET TRAP CODE
354              BIC  # 177400, (SP)     ; EXTRACT ERROR NUMBER
355              TST  1..ER              ; FIRST ERROR OCCURENCE ?
356              BEQ  2$                ; IF EQ, FIRST PASS
357              TST  PSDSWR              ; SCOPE LOOP ?
358              BPL  3$                ;
359              CMP  (SP), 1..ER        ; SAME SCOPE LOOP ?
360              BEQ  3$                ; IF EQ, YES
361              WRITE M1LSCP             ; ILLEGAL SCOPE LOOP
362              DUMP OCT, 2(SP)          ; DUMP ADDRESS OF ERROR *2
363              GOTO START              ; RESTART PROGRAM
364              MOV  (SP), 1..ER        ; STORE ERROR NUMBER
365              BIT  #BIT14, PSDSWR     ; INHIBIT PRINTOUT ?
366              BNE  4$                ; IF NE, YES
367              WRITE MERROR            ; PRINT ERROR MESSAGE PART 1
368              DUMP DEC, (SP)          ; DUMP ERROR NUMBER
369              WRITE MERRP2            ; PRINT ERROR MESSAGE PART 2
370              DUMP OCT, 2(SP)          ; DUMP ERROR PC *2
371              BIT  #BIT13, PSDSWR     ; COUNT ERRORS ?
372              BEQ  5$                ; IF EQ, NO
373              INC  B2(SP)             ; COUNT !
374              BNE  5$                ; IF EQ, OVERFLOW
375              WRITE MERRCOV           ; ERROR COUNTER OVERFLOW
376              DUMP DEC, (SP)          ; DUMP ERROR NUMBER
377              BIT  #BIT14, PSDSWR     ; PRINT ? ?
378              BNE  6$                ; IF NE, NO
379              BIT  #BIT15, PSDSWR     ; SCOPE LOOP ?
380              BNE  6$                ; IF SCOPE, NO PRINT
381              BIT  #BIT12, PSDSWR     ; HALT AFTER ERROR SELECTED
382              BNE  6$                ; IF NE, YES
383              WRITE MERRHLT           ; STOP MESSAGE
384              BIT  #BIT12, PSDSWR     ; HALT ?
385              BNE  7$                ; IF EQ, YES

```

IEC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 8 1
 8..TRP

SEQ 60

386	003004	032737	004000	007160		BIT	#BIT11,PSDSWR	;IMMEDIATE CONTINUE ?
387	003012	001770				BEQ	61	;IF NE,YES
388	003014	042737	004000	007160	71:	BIC	#BIT11,PSDSWR	;SCOPE LOOP ?
389	003022	005737	007160			TST	PSDSWR	;
390	003026	100005				BPL	81	;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		91:	CLR	1..ER	;RESET ERROR NUMBER
395	003046	005726				TST	(SP).	;READJUST STACK
396	003050	062'16	000002			ADD	#2,(SP)	;BUILD RETURN ADDRESS
397	003054	000002				RTI		;RETURN TO CALLER

```

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

```

```

422 003056
423 003066 162716 000002
424 003072
425 003102
426 003112
427 003124 000777

```

```

430          .SBTTL  $.PWR
431          ;
432          ; SUBROUTINE:  $.PWR
433          ; *****
434          ;
435          ; THIS ROUTINE IS DIRECTLY ENTERED AFTER EACH POWER-FAIL TO
436          ; HANDLE THE NECESSARY REGISTER SAVE OPERATIONS, AND IS USED
437          ; AFTER THE POWER-UP INTERRUPT TO RESTORE THE SAVED REGISTERS
438          ; AFTER A SUCCESSFUL POWER-FAIL RESTART, A SHORT MESSAGE IS PRINTED.
439          ;
440          ; AUTHOR:      BERT HUBER, CSS/DP MUNICH 27-JULY-76
441          ;             MZ001  SAVE THE OUTFLAG IN 3$ AND RESTORE IT
442          ;             ***** AFTER PRINTOUT OF THE POWERFAIL MESSAGE
443          ;
444          ;
445          ; .PSECT CSSMON
446          ; ENABL  LSB
447          $.PWR: PUSH  R0,R1,R2,R3,R4,R5
448          MOV      SP,(PC)+          ;SAVE STACK-POINTER
449          .WORD    0                  ;STACKPOINTER-SAVE-AREA
450          MOV      #2$,#24          ;SETUP VECTOR FOR RESTART
451          MOV      OUTFLG,(PC)+      ;SAVE PRINT FLAG          ;MZ001
452          .WORD    0                  ;SAVE FOR FLAG          ;MZ001
453          MOV      INFLAG,(PC)+     ;SAVE INPUT FLAG
454          .WORD    0                  ;SAVE WORD FOR INPUT FLAG
455          BR       .                  ;WAIT FOR POWER DOWN
456          CLR      R2                  ;DELAY
457          INC      R2                  ;TIME
458          CMP      R2,#177777        ;POWER UP START
459          BNE      . 6                  ;FOR 1SECOND
460          MOV      1$,SP              ;REBUILD STACKPOINTER
461          CLR      OUTFLG              ;ALLOW PRINTOUT OF PWRFL MES.  ;MZ001
462          CLR      INFLAG              ;
463          MOV      #1$, $.PWR,#24     ;REBUILD VECTOR SETUP
464          POP      R5,R4,R3,R2,R1,R0
465          MOV      1$, $.MSP,(PC)+    ;SAVE OLD MESSAGE POINTER
466          .WORD    0                  ;SAVE WORD FOR OLD MESSAGE POINTER
467          WRITE    MPWRFL              ;SEND POWER-FAIL-MESSAGE          ;MZ001
468          TST      OUTFLG              ;WAIT FOR OUTPUT COMPLETE          ;MZ001
469          BNE      4$                  ;
470          SET6     #4TKS                ;ENABLE FURTHER TTY-INTERRUPTS
471          MOV      3$,OUTFLG            ;RESTORE FLAG IF IT WAS SET          ;MZ001
472          TST      OUTFLG              ;IF AN OUTPUT BEFORE PWRFL WAS RUNNING
473          BEQ      7$                  ;
474          MOV      8$, $.MSP            ;THEN RESTORE OLD MESSAGE POINTER
475          SET6     #4TPS                ;AND ENABLE INT IN TPS
476          MOV8     #12,$#TPB            ;AND START INTERRUPT
477          TST      5$                  ;IF AN INP FLG BEFORE PWRFL WAS NOT SET
478          BEQ      6$                  ;GOTO 6$
479          WRITE    PROMES              ;ELSE WRITE >
480          MOV      5$,INFLAG            ;RESTORE INPUT FLAG IF IT WAS SET
481          RTI                          ;CONTINUE WITH PROGRAM
482          .DSABL  LSB
483          .SBTTL  $.RSV
484          ;
485          ; SUBROUTINE:  $.RSV
486          ; *****

```

```

487      ; THIS ROUTINE SAVES R0 R5 ONTO THE STACK
488      ;
489      ;
490      ; INPUT:      (SP)      *      CALLER PC
491      ;
492      ; OUTPUT:     (SP)      *      R5 CONTENT
493      ;             2(SP)     *      R4 CONTENT
494      ;             4(SP)     *      R3 CONTENT
495      ;             6(SP)     *      R2 CONTENT
496      ;             10(SP)    *      R1 CONTENT
497      ;             12(SP)    *      R0 CONTENT
498      ;             14(SP)    *      CALLER PC
499      ;
500      ; AUTHOR:      BERT HUBER, CSS/DP MUNICH   12-APR 76
501      ;
502      ; .PSECT CSSMON
503      ; .ENABL  LSB
504 003360 $.RSV: PUSH  R0,R1,R2,R3,R4,R5
505 003374 000176 000014 JMP  014(SP)      ;RETURN TO CALLER
506      ; .DSABL  LSB

```

```

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

```

```

536 .SBTTL $..KBI
537
538 ; SUBROUTINE: $..KBI
539 ; .....
540
541 ; THIS ROUTINE IS ENTERED IMMEDIATELY AFTER AN INTERRUPT
542 ; FROM TTY-KEYBOARD. ALL REGISTERS ARE SAVED ACROSS CALL.
543
544 ; INPUT: 2(SP)= PS
545 ; (SP) = PC
546
547 ; OUTPUT: NONE
548
549 ; AUTHOR: BERT HUBER, CSS/DP MUNICH 5 APR 76
550 ; -
551 ; .PSECT CSSMON
552 ; ENABL LSB
553 003424 005737 007152 $..KBI: TST OUTFLG ;;;OUTPUT RUNNING ?
554 003430 001422 BEQ 4$ ;;;IF NE, IGNORE INTERRUPT
555 003432 PUSH R0 ;;;SAVE
556 003434 005037 177560 CLR @TKS ;;;INHIBIT INTERRUPT
557 003440 013700 177562 MOV @TKB,R0 ;;;GET CHARACTER
558 003444 012737 000100 177560 MOV @100,@TKS ;;;RE ENABLE
559 003452 042700 000200 BIC @200,R0 ;;;MASK OUT ASCII
560 003456 022700 000003 CMP @3,R0 ;;; IS IT <CNTRL C> ?
561 003462 001003 BNE 13$ ;;;IF NOT, RETURN
562 003464 012737 007777 007170 MOV @7777,SPEFLG ;;;SET A SPECIAL FLAG
563 ;;;THIS FLAG IS USED TO FINISH
564 ;;;A RUNNING PRINTOUT AND THEN TO
565 ;;;REMEMBER THAT <CNTRL C> WAS HIT
566
567 003472 13$: POP R0 ;;;REBUILD
568 003474 000436 BR 3$ ;;; RETURN TO CALLER
569 003476 004737 003360 4$: CALL $..RSV ;;;SAVE ALL REGISTERS
570 003502 013700 177562 MOV @TKB,R0 ;;;GET INPUT BYTE
571 003506 042700 000200 BIC @200,R0 ;;;BUILD REAL ASCII
572 003512 122700 000003 CMPB @3,R0 ;;;<CNTRL C> FUNCTION ?
573 003516 001003 BNE 1$ ;;;IF NE, NO
574 003520 $$SPE: CALL $..PRO ;;;SEND PROMPT STRING
575 003520 004737 006136 BR 2$ ;;;RETURN
576 003524 000420 BR 2$ ;;;RETURN
577 003526 005737 007150 1$: TST INFLAG ;;;INPUT FLAG SET ?
578 003532 001415 BEQ 2$ ;;;IF EQ, NO
579 003534 004737 003574 5$: CALL $..INP ;;;PROCESS INPUT
580 003540 020027 000015 CMP R0,@15 ;;;WAS IT A <CR> ?
581 003544 001010 BNE 2$ ;;;IF NE, NO
582 003546 005037 007150 CLR INFLAG ;;;RESET INPUT MODE FLAG
583 003552 012701 007026 MOV @KBBUFF,R1 ;;;REBUILD BUFFER POINTER
584 003556 010137 007146 MOV R1,KBBPNT ;;;
585 003562 004737 004010 CALL $..IAY ;;;ANALYZE INPUT
586 003566 004737 003400 2$: CALL $..RRS ;;;RESTORE REGISTERS
587 003572 000002 3$: RTI ;;;RETURN TO CALLER
588 .DSABL LSB

```

```

590 .SBTTL 1..INP
591
592 SUBROUTINE: 1..INP
593 .....
594
595 THIS SUBROUTINE IS THE INPUT-HANDLER FOR ALL TERMINALS
596 USED IN ALL CSS-MUNICH DIAGNOSTIC SOFTWARE GENERATED NOW
597 AND LATER.
598 THE SPECIAL FUCTIONS HANDLED BY THIS ROUTINE ARE:
599 <CNTRL U> FOR LINE RUBOUT
600 <DELETE> FOR CHARACTER RUBOUT
601 ALL OTHER INPUT IS HANDLED AS NORMAL ASCII TEXT AND STORED
602 IN THE KEYBOARD INPUT BUFFER.
603
604 INPUT: (SP) - CALLER PC
605 RO - CURRENT INPUT CHARACTER
606
607 OUTPUT: NONE
608
609 AUTHOR: BERT HUBER CSS/DP MUNICH
610
611 .IDENT /V1.0/
612 .PSECT CSSMON
613 .ENABL LSH
614 003574 013701 007146 1..INP: MOV KBBPNT,R1 ;;;GET CURRENT BUFFER POINTER
615 003600 012702 006122 MOV #1,KBO,R2 ;;;POINT TO KEYBOARD-ECHO
616 003604 022701 007146 CMP #KBBEND,R1 ;;;INPUT-BUFFER FULL?
617 003610 001015 BNE 1$ ;;;IF NE, NO
618 003612 022700 000025 CMP #25,RO ;;;<CNTRL U>?
619 003616 001415 BEQ 2$ ;;;IF EQ, YES.
620 003620 122700 000177 CMPB #177,RO ;;;<DELETE>?
621 003624 001407 BEQ 1$ ;;;IF EQ, YES.
622 003626 005037 007150 CLR INFLAG ;;;RESET INPUTFLAG
623 003632 WRITE MKBOVF ;;;PRINT OVERFLOW-MESS.
624 003642 000451 BR 7$ ;;;PROMPT FOR INPUT
625 003644 122700 000025 1$: CMPB #25,RO ;;;<CNTRL U>?
626 003650 001436 BEQ 6$ ;;;IF NE, NO
627 003652 122700 000177 2$: CMPB #177,RO ;;;<DELETE>?
628 003656 001016 BNE 4$ ;;;IF NE, NO
629 003660 005737 007162 TST RUBFLG ;;;FIRST RUBOUT?
630 003664 001005 BNE 3$ ;;;IF NE, NO
631 003666 010637 007162 MOV SP,RUBFLG ;;;FLAG RUBOUT MODE
632 003672 012700 000057 MOV #'/,RO ;;;PRINT "/"
633 003676 004712 CALL BR2 ;;;
634 003700 020127 007026 3$: CMP R1,#KBBUFF ;;;FULL LINE DELETED?
635 003704 001434 BEQ INP..E ;;;IF EQ, YES
636 003706 114100 MOVB (R1),RO ;;;NO, REMOVE CHARACTER
637 003710 004712 CALL BR2 ;;;AND ECHO IT
638 003712 000431 BR INP..E ;;;WAIT FOR NEXT INPUT
639 003714 005737 007162 4$: TST RUBFLG ;;;RUBOUT MODE?
640 003720 001407 BEQ 5$ ;;;IF EQ, NO
641 003722 PUSH RO ;;;YES, SAVE CURRENT INPUT
642 003724 012700 000057 MOV #'/,RO ;;;END OF RUBOUT
643 003730 004712 CALL BR2 ;;;ECHO "/"
644 003732 005037 007162 CLR RUBFLG ;;;RESET RUBOUT MODE FLAG
645 003736 POP RO ;;;GET CURRENT INPUT AGAIN
646 003740 004712 5$: CALL BR2 ;;;ECHO IT

```

647	003742	110021		MOVB	R0,(R1).	SAVE IN INPUT BUFFER
648	003744	000414		BR	INP..E	WAIT FOR NEXT INPUT
649	003746	012700	000136	61:	MOV	#136,R0
650	003752	004712			CALL	SR2
651	003754	012700	000125		MOV	#'U,R0
652	003760	004712			CALL	SR2
653	003762	005037	007162		CLR	RUBFLG
654	003766	012701	007026	71:	MOV	#KBBUFF,R1
655	003772	004737	006136		CALL	S..PRO
656	003776	010137	007146	INP..E:	MOV	R1,KBBPNT
657	004002	116100	177777		MOVB	1(R1),R0
658	004006	000207			RETURN	
659					.DSABL	LSB

```

661          .SBTTL  $..IAY
662          ;
663          ;SUBROUTINE:  $..IAY
664          ;.....
665          ;
666          ;LEGAL INPUT TYPES ARE:
667          ;
668          ;      ERR=          FOR ERROR-HANDLING
669          ;      DMP=          FOR CORE DUMP OPTIONS
670          ;      SET=          FOR DEBUGGING OPTIONS
671          ;      NONFORMATTED INPUT FOR SPECIAL REQ.
672          ;
673          ;INPUT: (SP)=  RETURN PC
674          ;OUTPUT:      NONE
675          ;
676          ;AUTHOR:        BERT MUBER, CSS/DP MUNICH      28 JUN 76
677          ;
678          ;.PSECT CSSMON
679          ;.ENABL  LSB
680 004010 122711 000015  $..IAY: CMPB  #15,(R1)      ;;;<CR>?
681 004014 001440          BEQ   IAY..E      ;;;IF EQ., SKIP ANALIZATION
682 004016 005737 007174 3$:   TST   $,ASF      ;;;DONT ANALYZE IF ASCII INPUT RUNNING
683 004022 001035          BNE   IAY..E      ;;;IF NE, YES.
684 004024 122711 000104      CMPB  #'D,(R1)    ;;;'DMP' REQUESTED?
685 004030 001003          BNE   1$          ;;;IF NE, NO
686 004032 004737 004210      CALL  $..IAD      ;;;ANALYZE DUMP REQUEST
687 004036 000427          BR     IAY..E      ;;;RETURN TO CALLER
688 004040 122711 000123 1$:   CMPB  #'S,BR1    ;;;'SET' REQUESTED?
689 004044 001003          BNE   2$          ;;;IF NE, NO
690 004046 004737 004402      CALL  $..IAS      ;;;ANALYZE SET REQUEST
691 004052 000421          BR     IAY..E      ;;;RETURN TO CALLER
692 004054 122711 000105 2$:   CMPB  #'E,BR1    ;;;'ERR' REQUEST?
693 004060 001003          BNE   4$          ;;;IF NE, NO
694 004062 004737 004472      CALL  $..IAE      ;;;ANALYZE ERROR REQUEST
695 004066 000413          BR     IAY..E      ;;;RETURN TO CALLER
696 004070 122711 000101 4$:   CMPB  #'A,BR1    ;;;'ABO' REQUESTED ?
697 004074 001003          BNE   5$          ;;;IF NE, NO
698 004076 004737 004120      CALL  $..IAA      ;;;ANALYZE ABO-REQUEST
699 004102 000405          BR     IAY..E      ;;;RETURN TO CALLER
700 004104 005737 007156 5$:   TST   INPREQ    ;;;NO SYNTAX ERROR IF SPECIAL INPUT RUNNING
701 004110 001002          BNE   IAY..E      ;;;
702 004112 004737 006262      CALL  $..STX      ;;;SYNTAX-ERROR
703 004116 000207          IAY..E: RETURN      ;;;RETURN TO CALLER
704          .DSABL  LSB

```

```

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

```

```

754          .PSECT CSSMON
755          ;
756          ;
757          ;
758          ;
759          .SBTTL $.IAD
760          ;
761          SUBROUTINE      :      $.IAD
762          :               :      .....
763          ;
764          MZ001
765          .....
766          MODIFICATION TO ALLOW THE <CR> AFTER INPUT
767          TO COMPLETE.
768          ;
769          ;
770          THIS SUBROUTINE IS ENTERED WHEN A DUMP-REQUEST IS ANALYZED.
771          DEPENDING ON HOW THE REQUEST IS TERMINATED, THERE ARE TWO
772          WAYS FOR RETURN:
773          ;
774          A)      WHEN TERMINATED BY <CR> THE CONTENT OF A LOCATION IS
775          PRINTED AND THE RETURN IS PERFORMED
776          B)      WHEN INPUT IS TERMINATED BY A "'/'" THE MONITOR WAITS
777          FOR INPUT AND LOADS THIS INPUT INTO THE EXAMINED LOCATION.
778          ;
779          INPUT :      (SP)      =RETURN ADDRESS
780          OUTPUT :      MODIFIED LOCATION IF REQUESTED
781          ;
782          AUTHOR:      M. ZILLER, SS/DP MUNICH          OCTOBER 1976
783          ;
784          ;
785          .PSECT CSSMON
786          .ENABL  LSB
787          $.IAD: INC      R1          ;POINT TO "M
788          CMPB      (R1),0,M        ;IS IT A "M"
789          BNE       77$             ;NO IF NE
790          CMPB      (R1),0,P        ;IS IT A "P"
791          BNE       77$             ;NO IF NE
792          CMPB      (R1),0,/        ;IS IT A "/"
793          BNE       77$             ;NO IF NE
794          CALL      $.NOR           ;RETRIEFE INPUT
795          BCS       77$             ;IS CS, INPUT FORMAT ERROR
796          11$: TSTB      @TPS        ;IS THE <CR> FINISHED ? ;MZ001
797          BPL       11$             ;MZ001
798          MOV       @12,@TPB        ;PRINT A <LF
799          12$: TSTB      @TPS        ;DONE ?
800          BPL       12$             ;WAIT FOR COMLETION
801          DUMP      OCT,(R0)         ;AND PRINT
802          BR        IAD..E          ;GO TO END
803          77$: CALL      $.STX       ;WRITE SYNTAX ERROR MESSAGE
804          IAD..E: RETURN            ;RETURN TO MAIN
805          ;

```

787	004210	005201		
788	004212	122127	000115	
789	004216	001027		
790	004220	122127	000120	
791	004224	001024		
792	004226	122127	000057	
793	004232	001021		
794	004234	004737	004304	
795	004240	103416		
796	004242	105737	177564	
797	004246	100375		
798	004250	012737	000012	177566
799	004256	105737	177564	
800	004262	100375		
801	004264			
802	004274	000402		
803	004276	004737	006262	
804	004302	000207		
805				

```

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

;
; THIS IS THE ROUTINE GET THE OCTAL VALUE OF THE ADDRESS
;
;
;..NOR: CLR R0 ;CLEAR STORAGE OF NUMBER
; MOV #6,R2 ;CLEAR DIGIT COUNTER
18: CMPB BR1,#15 ;IS IT A TERMINATOR ?
; BEQ NOR..E ;IF EQ. YES
;
;
; BICB #60,BR1 ;BUILD REAL BINARY
; BMI NOR..E ;IF HI, IT WAS LOWER THAN 60
; CMPB BR1,#7 ;IS IT REAL OCTAL ?
; BMI NOR..E ;IF HI,NO REAL OCTAL
; ASL R0 ;MULTIPLY
; ASL R0 ;WITH 10 IN
; ASL R0 ;OCTAL
; BISB (R1),R0 ;SET THE TRUE BITS
; DEC R2 ;COUNT DIGITS BUILT
; BNE 18 ;ALL DONE?
; CMPB BR1,#15 ;IS THE LAST + 1 BYTE A TERMINATOR?
; BEQ NOR..E ;IF YES THEN SYNTAX OK
; CMPB BR1,#7 ;TERMINATOR IN SET COMMAND?
; BNE NOR..E ;IF NOT THEN SYNTAX ERROR
; NOR..E: CLC ;INDICATE SUCCESSFUL OPERATION
; RETURN ;
; NOR..E: SEC ;INDICATE FAILURE
; RETURN ;

;
; SUBROUTINE: $..IAS
; ..IAS:
;
; THIS SUBROUTINE IS ENTERED IF THE INPUT ANALYZER DETECTS A SET-COMMAND
;
; INPUT: (SP) - RETURN ADDRESS
; R1 - BEGIN OF INPUT STRING
;
; OUTPUT: - MODIFIED LOCATION
;
;
;..IAS: INC R1 ;POINT TO "E"
; CMPB (R1),#E ;CORRECT
; BNE 76$ ;NO IF NE
; CMPB (R1),#T ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CMPB (R1),#7 ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CALL $..NOR ;BUILD ADDRESS TO MODIFY
; BCS 76$ ;IF CS, FORMAT ERROR
; MOV R0,2$ ;SAVE
; CMPB (R1),#7 ;CORRECT SYNTAX ?
; BNE 76$ ;NO IF NE
; CALL $..NOR ;BUILD DATA TO INSERT
; BCS 76$ ;IF CS, FORMAT ERROR
; MOV R0,2$ ;AND LOAD

```

H6

REC11 R TEST MACRO M1200 30 MAR 84 16:11 PAGE 17 1
...IAD

SEQ 72

864	004460	000402		BR	IAS..E		GO TO END
865	004462	004737	006262	768:	CALL	8..SIX	WRITE ERROR MESSAGE
866	004466	000207		IAS..E:	RETURN		RETURN
867	004470	000000		28:	.WORD	0	SORAGE FOR ADDRESS TO MODIFY
868					.DSABL	LSB	

```

870      ;
871      ;
872      ;
873      ;       .SBTTL  8..IAE
874      ;
875      ;SUBROUTINE:  8..IAE
876      ;.....
877      ;
878      ;THIS ROUTINE CHECKS THE CORRECT SYNTAX OF
879      ;AN "ERR" INPUT STRING AND PROCESSES THE
880      ;REQUESTED SWITCH REGISTER-OPTIONS .
881      ;
882      ;LEGAL SWITCH OPTIONS ARE:
883      ;
884      ;       /MT      *      HALT AFTER ERROR (DEFAULT)
885      ;       /NM      *      NO HALT AFTER ERROR
886      ;       /PR      *      PRINT ERROR MESSAGE (DEFAULT)
887      ;       /NP      *      INHIBIT ERROR PRINTOUT
888      ;       /CT      *      COUNT ERROR
889      ;       /NC      *      NO ERROR COUNTING (DEFAULT)
890      ;       /SC      *      SCOPE LOOP ON THIS ERROR
891      ;       /NS      *      NO SCOPE LOOP (DEFAULT)
892      ;       /CONT    *      IMMEDIATE CONTINUE SWITCH
893      ;
894      ;
895      ;INPUT:         R1      *START ADDR. OF TEXT STRING
896      ;OUTPUT:        BIT SETUP IN SOFTWARE-SWITCH-REGISTER
897      ;
898      ;AUTHOR:        BERT HUBER, CSS/DP MUNICH  28 JULY 76
899      ;
900      ;.PSECT CSSMON
901      ;.ENABL  LSB
902      8..IAE: MOV     SP,R2          ;;;FLAG FOR VERY FIRST CHARACTER
903      MOV     @PSDSWR,R3          ;;;GET SWR" ADDRESS
904      CLR     R4                  ;;;USE AS 'CHARACTER 1'
905      ADD     @3,R1               ;;;SKIP IDENTIFIER
906      CMPB    (R1),0              ;;;CORRECT S'NTAX?
907      BNE     77$                ;;;BRANCH IF NOT
908      1$: CMPB    (R1),0 /        ;;;IS NEXT CHAR. A  /
909      BEQ     2$                  ;;;IF EQ. YES
910      TST     R2                  ;;;VERY FIRST?
911      BNE     77$                ;;;IF YES: SYNTAX ERROR
912      BR      11$                 ;;;ASSUME END OF STRING
913      2$: CLR     R2              ;;;NO LONGER VERY FIRST
914      INC     R4                  ;;;SET FOR 'C1'-FLAG
915      CMPB    (R1),0 N           ;;;IS IT THE '/N' OPTION?
916      BNE     3$                  ;;;BRANCH IF NOT
917      INC     R1                  ;;;ADVANCE TO NEXT CHARACTER
918      CLR     R4                  ;;;RESET 'C1' FLAG
919      3$: CMPB    (R1),0 S         ;;;IS IT A 'S'?
920      BEQ     5$                  ;;;BRANCH IF YES
921      CMPB    (R1),0 C           ;;;IS IT A 'C'?
922      BEQ     7$                  ;;;BRANCH IF YES
923      CMPB    (R1),0 M           ;;;IS IT A 'M'?
924      BEQ     9$                  ;;;BRANCH IF YES
925      CMPB    (R1),0 P           ;;;IS IT A 'P'?
926      BNE     77$                ;;;IF NO: SYNTAX ERROR

```

927	004576	005704				TST	R4	IS IT A 'P' ?
928	004600	001004				BNE	48	BRANCH IF YES
929	004602	052713	040000			BIS	#BIT14,(R3)	SET SR-BIT FOR /NP
930	004606	005201			30:	INC	R1	ADVANCE TO NEXT CHARACTER
931	004610	000741				BR	18	CHECK FOR NEXT SWITCH
932	004612	005201			4:	INC	R1	NEXT CHARACTER
933	004614	122127	000122			CMPS	(R1),#R	IS IT A '/PR' ?
934	004620	001074				BNE	77	IF NO: SYNTAX ERROR
935	004622	042713	040000			BIC	#BIT14,(R3)	SET '/PR'-OPTION
936	004626	000732				BR	18	CHECK NEXT SWITCH
937	004630	005704			5:	TST	R4	IS IT 'S' ?
938	004632	001003				BNE	68	BRANCH IF YES
939	004634	042713	100000			BIC	#BIT15,(R3)	SETUP FOR '/NS' OPTION
940	004640	000762				BR	30	TRY NEXT
941	004642	005201			6:	INC	R1	NEXT CHARACTER
942	004644	122127	000103			CMPS	(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	18	CHECK NEXT SWITCH
946	004660	126127	000001	000117	7:	CMPS	1(R1),#O	CHECK FOR '/CONT'
947	004666	001016				BNE	70	IF NE, OTHER
948	004670	126127	000002	000116		CMPS	2(R1),#N	CHECK FOR '/CONT'
949	004676	001045				BNE	77	IF NO: SYNTAX ERROR
950	004700	126127	000003	000124		CMPS	3(R1),#T	CHECK FOR '/CONT'
951	004706	001041				BNE	77	IF NO: SYNTAX ERROR
952	004710	052737	004000	007160		BIS	#BIT11,PSDSWR	SET 'IMMEDIATE CONTINUE
953	004716	062701	000005			ADD	#5,R1	ADJUST POINTER
954	004722	000430				BR	11	END THIS SCAN
955	004724	005704			70:	TST	R4	IS IT A '/CT' ?
956	004726	001003				BNE	88	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			CMPS	(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	18	CHECK NEXT SWITCH
964	004754	005704			9:	TST	R4	IS IT '/HT' ?
965	004756	001003				BNE	108	BRANCH IF YES
966	004760	052713	010000			BIS	#BIT12,(R3)	SETUP FOR '/NP' OPTION
967	004764	000710				BR	30	TRY NEXT SWITCH
968	004766	005201			10:	INC	R1	ADVANCE TO NEXT CHARACTER
969	004770	122127	000124			CMPS	(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	18	TRY NEXT SWITCH
973	005004	124127	000015		11:	CMPS	(R1),#15	WAS LAST BYTE A <CR>?
974	005010	001402				BEQ	IAE..E	BRANCH IF YES
975	005012	004737	006262		77:	CALL	S..STX	SYNTAX ERROR
976	005016	000207			IAE..E:	RETURN		RETURN TO CALLER
977						.DSABL	LSB	

```

979      .SBTTL  $..RED
980
981      ; SUBROUTINE  $..RED
982      ; .....
983      ;
984      ;
985      ; **** AB00
986      ;      THE CARRY BIT IS SET IF A ZERO OCTAL INPUT (ONLY CR,NO DATA)
987      ;      HAPPENS
988      ;
989      ;
990      ; THIS SUBROUTINE HANDLES ALL TYPES OF SPECIAL INPUT REQUESTS
991      ; SETUP BY THE PROGRAMMER THROUGH A ' READ ' MACRO.
992      ; LEGAL INPUT TYPES THAT CAN BE HANDLED ARE:
993      ;
994      ; BINARY FORMAT (EX: 1001101101010001)
995      ; .....
996      ; INPUT CAN BE UP TO 16 DIGITS OF THE FORM 0 OR 1.
997      ; LESS THAN 16 DIGIT INPUT IS FILLED WITH LEADING ZEROES
998      ; INPUT IS RETURNED IN R0.
999      ; THE PREVIOUS CONTENT OF R0 WILL BE OVERWRITTEN.
1000      ;
1001      ; OCTAL FORMAT (EX:176523)
1002      ; .....
1003      ; INPUT CAN BE UP TO 6 DIGITS OF THE FORM 0 7.
1004      ; IN CASE OF A 6-DIGIT-WIDE INPUT,THE LEFTMOST(HIGH ORDER)
1005      ; DIGIT IS TRUNCTATED TO THE RIGHTMOST (LOW ORDER)BIT.
1006      ; IN CASE OF LESS THAN 6 DIGITS INPUT, LEADING ZERO DIGITS
1007      ; ARE INSERTED.
1008      ; THE RESULT IS RETURNED IN R0,HEREBY OVERWRITING
1009      ; THE PREVIOUS CONTENT OF R0.
1010      ;
1011      ; DECIMAL FORMAT (EX: 96003)
1012      ; .....
1013      ; INPUT CAN BE UP TO 5 DIGITS OF THE FORM 0 9.
1014      ; LESS THEN FIVE DIGITS INPUT ARE FILLED UP WITH
1015      ; LEADING ZEROE DIGITS.
1016      ; MAXIMUM NUMBER ALLOWED IS 32656.
1017      ; INPUT IS RETURNED IN R0, HEREBY OVERWRITING THE
1018      ; PREVIOUS CONTENT OF R0.
1019      ;
1020      ; ASCII FORMAT (EX: B6T#&L)
1021      ; .....
1022      ;
1023      ; INPUT CAN BE ANY LEGAL ASCII CHARACTER.
1024      ; IF INPUT INTO A BUFFER IS REQUESTED, UP TO 80
1025      ; CHARACTERS CAN BE INPUT.
1026      ; IF INPUT RETURN IN R0 IS REQUESTED,ONLY THE FIRST
1027      ; CHARACTER WILL BE RETURNED IN THE LOW ORDER BYTE
1028      ; OF R0.
1029      ; OLD CONTENT OF R0 IS OVERWRITTEN IN THIS CASE.
1030      ; IF THE ONLY INPUT IS <CR>, ZERO WILL BE RETURNED IN R0
1031      ; AS WELL AS IN THE FIRST BYTE OF THE SPECIFIED BUFFER.
1032      ;
1033      ; AUTHOR:          BERT HUBER, CSS/DP MUNICH          29 JUN 76
1034      ;
1035      $..RED: CALL      $..PRO          ;PRINT PROMPT STRING

```

```

1035 005020 004737 006136

```

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

1093	005316	101412				BLOS	4:		;IF LOS, NO
1094	005320				3:	WRITE		MILDEC	;ILLEGAL DECIMAL INPUT
1095	005330					POP		R5,R4,R3,R2,R1	;REST. REGS
1096	005342	000626				BR		\$.RED	;RESTART INPUT
1097	005344	005704			4:	TST		R4	;ZERO DIGIT COUNT ?
1098	005346	001527				BEQ		RED..E	;IF EQ, YES
1099	005350	006304				ASL		R4	;BUILD TABLE OFFSET
1100	005352	016402	005360			MOV		5:(R4),R2	;SETUP CORRECT PARAMETER
1101	005356	000406				BR		6:	;SKIP TABLE
1102	005360	000000	000001	000012	5:	.WORD		0,1,10.,100.,1000.,10000.	
	005366	000144	001750	023420					
1103	005374	006204			6:	ASR		R4	;REBUILD R4 CONTENT
1104	005376	005003				CLR		R3	;RESET SAVE BUFFER
1105	005400	112103				MOVB		(R1),R3	;GET FIRST DIGIT
1106	005402	020327	000060			CMP		R3,#60	;CHECK LOWER DIGIT RANGE
1107	005406	103744				BLO		3:	;IF LO, ERROR
1108	005410	020327	000071			CMP		R3,#71	;CHECK UPPER DIGIT RANGE
1109	005414	101341				BHI		3:	;IF HI, ERROR
1110	005416	042703	177760			BIC		#177760,R3	;CONVERT TO OCTAL
1111	005422	005703				TST		R3	;ZERO DIGIT?
1112	005424	001405				BEQ		13:	;IF EQ, YES
1113	005426	000241			7:	CLC			;RESET CARRY
1114	005430	060200				ADD		R2,R0	;BUILD OCTAL NUMBER
1115	005432	103732				BCS		3:	;ERROR IF OVERFLOW
1116	005434	005303				DEC		R3	;ALL DONE?
1117	005436	001373				BNE		7:	;IF NE, NO
1118	005440	005304			13:	DEC		R4	;NEXT DIGIT
1119	005442	000740				BR		4:	
1120	005444	122711	000015			\$.BUF: CMPB		#15,(R1)	;ZERO DIGIT ?
1121	005450	001001				BNE		1:	;SKIP IF NOT
1122	005452	005011				CLR		(R1)	;RESET FIRST INPUT WORD
1123	005454	010100			1:	MOV		R1,R0	;TRANSFER BUFFER ADDRESS
1124	005456	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		INC	6\$	;COUNT INPUTS
1151	005612	000742		BR	1\$	;CHECK NEXT
1152	005614	121127	4\$:	CMPB	BR1,#15	;END OF INPUT ?
1153	005620	001343		BNE	5\$	;NO. ILLEGAL CHARACTER
1154	005522	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		MOV	(SP),2(SP)	;READJUST STACK RETURN ADDRESS
1158	005644	005726		TST	(SP)+	;
1159	005646	005037		CLR	INPREQ	;RESET SPECIAL REQUEST POINTER
1160	005652	000207		RETURN		;RETURN TO CALLER

```

1162          .SBTTL 0..WRT
1163
1164          SUBROUTINE: 0..WRT
1165          .....
1166
1167          THIS ROUTINE IS CALLED TO SETUP A MESSAGE PRINTOUT ON CONSOLE
1168          ALL PRINTING IS CONTROLLED BY INTERRUPT EXCEPT <CR>, <LF>.
1169
1170          INPUT:      2(SP)      ADDRESS OF MESSAGE TEXT
1171                     (SP)      CALLER PC
1172
1173          OUTPUT:      MESSAGE ON TERMINAL
1174
1175          AUTHOR:      BERT HUBER CSS/DP MUNICH      28 JUN 76
1176
1177          MODIFIED BY: JOHN LEVESQUE  CSS ISG  23 JAN 83
1178
1179          MODIFICATION REQUIRED TO HANDLE XON AND XOFF
1180
1181
1182          .PSECT CSSMON
1183          .ENABL  LSB
1184 005654 005737 007152 0..WRT: TST OUTFLG      ;OUTPUT RUNNING ?
1185 005660 001375      BNE 0..WRT      ;IF NE, YES - WAIT
1186 005662 005737 007150      TST INFLAG      ;INPUT RUNNING ?
1187 005666 001372      BNE 0..WRT      ;IF NE, YES - WAIT
1188 005670 010637 007152      MOV SP,OUTFLG      ;SET OUTPUTFLAG
1189 005674 016627 000002      MOV 2(SP),(PC).      ;GET MESSAGE POINTER
1190 005700 000000      0..MSP: .WORD 0      ;MESSAGE POINTER
1191 005702 012737 005736 000064      MOV 010,0064      ;SETUP VECTOR
1192 005710 012737 000340 000066      MOV 0340,0066      ;
1193 005716 012616      MOV (SP),(SP,      ;UPDATE STACK
1194 005720 052737 000100 177564      BIS 0100,00TPS      ;ENABLE INTERRUPTS
1195 005726 005737 007152      TST OUTFLG      ;OUTPUT RUNNING?      ;**GS0001**
1196 005732 001375      BNE -4      ;IF NE, YES - WAIT      ;**GS0001**
1197 005734 000207      RETURN      ;RETURN TO CALLER
1198 005736 122737 000023 177562 10: CMPB 023,00TKB      ;CHECK FOR XOFF
1199 005744 001401      BEQ 30      ;XOFF CONDITION WAIT TO CLEAR
1200 005746 000404      BR 200      ;NO XOFF THEN CONTINUE
1201 005750 122737 000021 177562 30: CMPB 021,00TKB      ;CHECK FOR XON
1202 005756 001367      BNE 10      ;NO XON GO WAIT FOR IT
1203 005760
1204 005760 127727 177714 000135 200: CMPB 010,MSP,0' )      ;END OF MESSAGE ?
1205 005766 001434      BEQ WRT..E      ;IF EQ, YES
1206 005770 127727 177704 000133      CMPB 010,MSP,0' [      ;IF NE, NO
1207 005776 001022      BNE 300      ;DISABLE INTERRUPTS
1208 006000 042737 000100 177564      BIC 0100,00TPS      ;SEND <CR>
1209 006006 012737 000015 177566      MOV 015,00TPB      ;WAIT FOR DONE
1210 006014 105737 177564      TSTB 00TPS      ;
1211 006020 100375      BPL -4      ;POINT TO NEXT CHARACTER
1212 006022 005237 005700      INC 0..MSP      ;SEND <LF>
1213 006026 012737 000012 177566      MOV 012,00TPB      ;RE-ENABLE INTERRUPT
1214 006034 052737 000100 177564      BIS 0100,00TPS      ;RETURN TO MAIN PROGRAM
1215 006042 000002      RTI      ;SEND NEXT CHARACTER
1216 006044 117737 177630 177566 300: MOVB 010,MSP,00TPB      ;POINT TO NEXT
1217 006052 005237 005700      INC 0..MSP      ;RETURN TO CALLER
1218 006056 000002      RTI

```

1219	006060	042737	000100	177564	WRT..E: BIC	#100.8#TPS	DISABLE OUTPUT INTERRUPT
1220	006066	005037	007152		CLR	OUTFLG	RESET OUTPUT FLAG
1221	006072	005737	007170		TST	SPEFLG	THIS FLAG IS SET ONLY
1222	006076	001410			BEQ	41	WHEN DURING A PRINTOUT
1223	006100	005037	007170		CLR	SPEFLG	THE <CNTRL C> COMMAND WAS
1224	006104	005037	007156		CLR	INPREQ	NECESSARY TO INTERRUPT THE
1225	006110	004737	003360		CALL	8..RSV	RUNNING PRINTOUT
1226	006114	000137	003520		JMP	11SPE	NOW NEW SPEC. INPUT IS PROCESSED
1227	006120	000002		41:	RTI		RETURN TO MAIN PROGRAM
1228					.DSABL	LSB	

```

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

```

```

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

```

```

1285          .SBTTL $..STX
1286          ;
1287          ;SUBROUTINE:  $..STX
1288          ;.....
1289          ;
1290          ;THIS ROUTINE PRINTS THE 'SYNTAX ERROR' MESSAGE
1291          ;
1292          ;INPUTS:          (SP)=  CALLER PC
1293          ;
1294          ;OUTPUT:         MESSAGE ON TERMINAL
1295          ;
1296          ;AUTHOR:         BERT HUBER CSS/DP MUNICH      28 JUN 76
1297          ;
1298          ;
1299          ;      MZ001
1300          ;      *****
1301          ;      THE USE OF THIS MACRO'S ALLOWS TO RUN ONE VERSION OF CDM
1302          ;      ON LSI AND PDP.
1303          ;
1304          ;-
1305          ;.PSECT CSSMON
1306          .ENABL  LSB
1307 006262 000240  $..STX: NOP          ;;;DUMMY
1308 006264          PREAD  STATUS  ;SAVE STATUS      ;MZ001
1309 006310          PSET   00      ;CLEAR PS        ;MZ001
1310 006334          WRITE  MSYERR  ;WRITE SYNTAX ERROR MESSAGE
1311 006344 005737 007152 1$:  TST   OUTFLG  ;WAIT FOR OUTPUT COMPLETE
1312 006350 001375          BNE    1$
1313 006352          PSET   STATUS  ;RELOAD PS      ;MZ001
1314 006376 000207          RETURN      ;;;RETURN TO CALLER
1315          .DSABL  LSB

```

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

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

```

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

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

```

1388	006574	005366	000002		DEC	2(SP)	;COUNT NUMBER OF CONF. DIGITS
1389	006600	001357			BNE	18	;ALL DONE
1390	006602	012737	006616	006614	MOV	@TABLE, @POINT	;REBUILD FOR NEXT USER
1391	006610	022626			CMF	(SP), (SP)	;CORRECT STACK FOR RETURN
1392	006612	000430			BR	DMP..E	;END OF DEC. DUMP
1393	006614	006616					
1394	006616	023420			POINT: TABLE		
1395	006620	001750			TABLE: 10000.		
1396	006622	000144			1000.		
1397	006624	000012			100.		
1398	006626	000001			10.		
1399					1.		
1400					.DSABL	LSB	
1401	006630	012766	000020	000004	.ENABL	LSB	
1402	006636	000241			S..DBN: MOV	@16., 4(SP)	;SETUP PRINT POINTER
1403	006640	006166	000006		18:	CLC	;RESET CARRY
1404	006644	103410				ROL	;SELECT FIRST CHARACTER
1405	006646	012700	000060			BCS	;IF " 1 " BRANCH
1406	006652	004737	006122			MOV	;SETUP A "0" CHARACTER
1407	006656	005366	000004		28:	CALL	;DUMP CHARACTER
1408	006662	001365				DEC	;COUNT CHARACTERS DUMPED
1409	006664	000403				BNE	;IF NE, CONTINUE
1410	006666	012700	000061			BR	;END OF DUMP
1411	006672	000767			38:	MOV	;SETUP A ' 1 " CHARACTER
1412						BR	;DUMP IT
1413	006674					.DSABL	
1414	006676	062706	000006		DMP..E: POP	RO	;RESTORE RO
1415	006702	000176	177772			ADD	;READJUST STACK
1416						JMP	;RETURN TO CALLER
						.DSABL	LSB

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

```

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

```

```

1437          .SBTTL  $..TOT
1438          ;
1439          ; THIS ROUTINE IS ENTERED AFTER THE MACRO "ADRTST" HAS BEEN EVOCED
1440          ; AND A BUS TIMEOUT HAS HAPPENED
1441          ;
1442          ; INPUT: 4(SP) = PC FOR PROGRAM CONTINUATION
1443          ;          R0   = REGISTER ADDRESS (E.G. 164000) WHICH SHOULD BE TESTED
1444          ;
1445          ; OUTPUT:      THE ERROR MESSAGE:
1446          ;              THIS ADDRESS IS NOT AVAILABLE ON UNIBUS: 164000
1447          ;
1448          ; AUTHOR:      ALBERT BREM, CSS MUC      FEB 1979
1449          ;
1450          ;
1451          $..TOT: ADD    #4, SP          ; CORRECT SP
1452                   WRITE $MNOAV        ; SAY ADDRESS NOT AVAILABLE ON UNIBUS
1453                   DUMP  OCT, R0        ; DUMP REGISTER ADDRESS
1454                   TST   OUTFLG         ; SYNCHRONIZE PRINTOUT
1455                   BNE   . 4            ; "
1456                   MOV   #6, @#4        ; REBUILD TRAP CATCHER
1457                   JMP   @ (SP)+         ; JUMP TO CONTINUATION LABEL
1458          ;
1459          ;
1460          ;
1461          ;

```

```

1463          .SBTTL $.BUF
1464          ;*
1465          ;SUBSECTION TO CONTAIN ALL CDM BUFFERS, CONSTANTS AND
1466          ; NOT MACRO DEFINED' PARAMETERS.
1467          ;
1468          ;AUTHOR:          BERT HUBER CSS/DP MUNICH
1469          ;
1470          ;.PSECT CSSMON
1471          .ENABL  LSB
1472          ;MEMORY MANAGEMENT DEFINITIONS
1473          172360 KDSAR0=172360
1474          172320 KDSDR0=172320
1475          172340 KISAR0=172340
1476          172352 KISAR5=172352
1477          172354 KISAR6=172354
1478          172356 KISAR7=172356
1479          172300 KISDR0=172300
1480          172314 KISDR6=172314
1481          172316 KISDR7=172316
1482          172200 SISDR0=172200
1483          177660 UDSAR0=177660
1484          177620 UDSDR0=177620
1485          177640 UISAR0=177640
1486          177650 UISAR4=177650
1487          177652 UISAR5=177652
1488          177654 UISAR6=177654
1489          177656 UISAR7=177656
1490          177600 UISDR0=177600
1491          177610 UISDR4=177610
1492          177612 UISDR5=177612
1493          177614 UISDR6=177614
1494          177616 UISDR7=177616
1495          170200 UBMPR=170200
1496          140000 CMODE=140000
1497          030000 PMODE=30000
1498          177572 SRO=177572
1499          172516 SR3=172516
1500 007016 000000  $$PTYP: .WORD 0          ; IF 0:=LSI; IF -1:=PDP
1501 007020 000000  $$MTYP: .WORD 0          ; IF 0:NO MM; IF 1:WITH MM
1502 007022 000000 000000 MEMEND: .WORD 0,0 ; TOP OF CORE AND CORE EXTENSION
1503                                     ; STARTING FROM HERE TILL PROMES
1504                                     ; THE CONTENTS WILL BE CLEARED
1505                                     ; WITH "ABO"
1506 007026          KBBUFF: .BLKB 80.          ; TERMINAL INPUT BUFFER
1507 007146          KBBEND:          ; REFERENCE LABEL
1508 007146 007026   KBBPNT: .WORD KBBUFF ; INPUT BUFFER POINTER
1509 007150 000000   INFLAG: .WORD 0        ; INPUT 'RUNNING' FLAG
1510 007152 000000   OUTFLG: .WORD 0        ; OUTPUT 'RUNNING' FLAG
1511 007154 000000   ERRFLG: .WORD 0        ; 'ERROR HANDLING REQ.' FLAG
1512 007156 000000   INPREQ: .WORD 0        ; 'WAIT FOR SPECIAL INPUT' FLAG
1513 007160 000000   PSDSWR: .WORD 0        ; SOFTWARE SWITCH REGISTER
1514 007162 000000   RUBFLG: .WORD 0        ; RUBOUTMODE-INDICATOR FLAG
1515 007164 000000   STATUS: .WORD 0        ; BUFFER PS-CONTENT IF SAVED
1516 007166 000000   $.ER: .WORD 0         ; HOLDS CURRENT ERROR NUMBER
1517 007170 000000   SPEFLG: .WORD 0        ; USED IN $.KBI AND $.WRT
1518 007172 001000   SCOPAD: 1000          ; SAVE AREA FOR SCOPE LOOP ADDRESS
1519 007174 000000   $.ASF: .WORD 0        ; SPECIAL ASCII INPUT FLAG

```

Address	Hex	ASCII	Word	Byte	Comment
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	MBOVF: .ASCII /{KEYBOARD BUFFER OVERFLOW! REENTER}/
1526	007256	133	111	114	MIOCT: .ASCII /{ILLEGAL OCTAL INPUT !! REENTER}/
1527	007316	133	111	114	MILDEC: .ASCII /{ILLEGAL DECIMAL INPUT !! REENTER}/
1528	007360	133	111	114	MILBIN: .ASCII /{ILLEGAL BINARY INPUT !! REENTER}/
1529	007421	133	111	114	MILEMT: .ASCII /{ILLEGAL EMT-NUMBER}/
1530	007445	133	102	131	MILADR: .ASCII /{BYTE ADDRESS IS NOT ALLOWED }/
1531	007503	133	123	131	MSYERR: .ASCII /{SYNTAX ERROR !}/
1532	007524	133	123	103	MILSCP: .ASCII /{SCOPE LOOP SKIPPED TO A SUBSEQUENT ERROR !!{/
1533	007601	133	123	125	.ASCII /{SUSPECT AN INTERMITTEND HARDWARE ERROR(}/
1534	007652	133	105	122	MERROR: .ASCII /{ERROR }/
1535	007662	040	040	040	MERRP2: .ASCII / AT LOCATION : }/
1536	007704	133	120	122	MPWRFL: .ASCII /{PROGRAM MADE POWER FAIL (}/
1537	007737	133	105	122	MERCOV: .ASCII /{ERROR COUNTER OVERFLOW ON ERROR: }/
1538	010002	133	056	056	MERHLT: .ASCII /{...HALT AFTER ERROR...}/
1539	010032	133	117	120	MNOSUB: .ASCII /{OPTION NOT SUPPORTED IN THIS CDM VERSION}/
1540	010104	133	111	114	MILTR1: .ASCII /{ILLEGAL INTERRUPT OR TRAP TO : }/
1541	010144	040	040	040	MILTR2: .ASCII / FROM: }/
1542	010156	133	133	133	MVERSN: .ASCII /{[[[CDM V3.5 ---CSS DIAGNOSTIC MONITOR- -{[[{/
1543	010252	110	111	107	.ASCII /{HIGHEST MEMORY ADDRESS : }/
1544	010304	133	040	103	MER1: .ASCII /{ CONTENT SHOULD BE : }/
1545	010333	040	040	040	MER2: .ASCII / BUT WAS : }/
1546	010351	133	133	124	\$MNOAV: .ASCII /{[[THIS ADDRESS IS NOT AVAILABLE ON UNIBUS: }/
1547	010426	133	115	105	MEMDEF: .ASCII /{MEMORY MANAGEMENT UNIT IS DEFECT!!!/
1548	010472	133	111	114	MAPER: .ASCII /{ILLEGAL MAPPING ADDRESS SPECIFIED!!!!}/
1549	010542	133	127	105	.ASCII /{WE WILL TRY WITHOUT MEMORY MANAGEMENT(}/
1550	010612	066	135		N06: .ASCII /6}/
1551					.EVEN
1552					.LIST BEX
1553					.DSABL LSB
1554	010614				CDM..E;

```

1      .TITLE      IEC11-B TEST
2      .SBTTL     LOCAL MACRO DEFINITIONS
3
4      :
5      : THIS SECTION CONTAINS ONLY MACROS WHICH ARE
6      : USED SEPARATE FROM THE MACRO LIBRARY (NEWMAC).
7      :
8      :
9      :
10     :
11     :
12     :
13     :
14     :
15     :
16     :
17     :

```

```

18     .MACRO      MCLEAR
19     MOV         #BIT5,@CIR          ; CLEAR REGISTERS IN IEC11-A
20     MOV         #BIT3,@CSR          ; CLEAR REGISTERS IN IEC11 B
21     .ENDM      MCLEAR
22
23     :
24     :
25     :

```

```

26     .MACRO      DATACC
27     BIT         #BIT15,@CIR          ; DATA ACCEPTED ?
28     BEQ         . 6                 ; IF YES,CONTINUE
29     BIC         #BIT15,@CIR          ; CLEAR "DATA ACC" IN CIR
30     .ENDM      DATACC
31
32     :
33     :
34     :

```

```

35     .MACRO      STALL
36     NOP
37     NOP
38     NOP
39     NOP
40     NOP
41     NOP
42     NOP
43     NOP
44     NOP
45     NOP
46     NOP
47     NOP
48     .ENDM
49
50     :
51     :
52     :

```

```

53     : THE MACRO "CACS" SETS THE IEC11 A
54     : TO CONTROLLER ACTIVE STATE (CONTROLLER IN CHARGE)
55     :
56
57

```

```

58     .MACRO      CACS,?A,?B
59     BIS         #BIT0,@CIR          ; SET SACS

```

IFC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 29 1
LOCAL MACRO DEFINITIONS

SEQ 92

58		BIS	#B116,BSMR	SET SIC
59	A:	BIT	#B113,BSMR	IS SIAS STILL SET ?
60		BNE	A	IF NOT,TEST AGAIN
61	B:	BIT	#B118,BSMR	IS CACS SET ?
62		BEQ	B	IF NOT,WAIT
63		BIC	#177400,BCIR	
64		.ENDM	CACS	
65				
66				
67				
68				
69				
70				
71				
72				
73				
74				
75				

```

77      .SBTTL      INPUT PROCEDURE
78 010614 032737 000003 007024 BUFPRP: BIT      #3, MEMEND-2      ; MORE THAN 32 K ?
79 010622 001012      BNE      18      ;
80 010624      IF MEMEND HI #157776 GOTO 18      ;
81 010640 013737 007022 026772      MOV      MEMEND, MEMSAV      ;
82 010646 000403      BR      28      ;
83 010650 012737 157776 026772 18:      MOV      #157776, MEMSAV      ; USE 28K FOR BUFFER
84 010656 013700 026772      MOV      MEMSAV, RO      ;
85 010662 162700 000277      SUB      #277, RO      ; SAVE ABSOLUTE LOADER AREA
86 010666 010037 027004      MOV      RO, BUFBE      ; LAST BYTE ADDRESS OF BUFFER B
87 010672 012737 031624 026774      MOV      #PRGEND, BUFAB      ; FIRST BYTE ADDRESS OF BUFFER A
88 010700 013700 027004      MOV      BUFBE, RO      ; BUILD LENGTH
89 010704 162700 031623      SUB      #PRGEND 1, RO      ; IN BYTES
90 010710 006200      ASR      RO      ; OF
91 010712 042700 100000      BIC      #BIT15, RO      ;
92 010716 010037 027000      MOV      RO, BUFFL      ; BUFFER A OR B
93 010722 062700 031624      ADD      #PRGEND, RO      ; BUILD FIRST BYTE ADDRESS OF
94 010726 010037 027002      MOV      RO, BUFBB      ; BUFFER B
95 010732 162700 000001      SUB      #1, RO      ; BUILD LAST BYTE ADDRESS
96 010736 010037 026776      MOV      RO, BUFAE      ; OF BUFFER A
97 010742      WRITE      MID      ; IDENTIFY TEST
98 010752      RGAIN: WRITE      MES1      ; ENTER FIRST REG OF IEC11 A
99 010762      READ      OCT, RO      ; INPUT REQUEST
100 010770      IF RO NE #0 GOTO 18      ;
101 011000 012700 160010      MOV      #160010, RO      ; SAVE DEFAULT VALUE
102 011004 032700 000007      18: BIT      #7, RO      ; CHECK THREE LSB OF INPUT
103 011010 001360      BNE      RGAIN      ; IF NOT CLEAR, REQUEST AGAIN
104 011012      IF RO LO #160010 GOTO RGAIN      ;
105 011022      IF RO HI #177776 GOTO RGAIN      ;
106 011032      ADRTST      RGAIN      ;
107 011056 010037 026732      MOV      RO, CIR      ; SAVE CIR ADDRESS
108 011062 062700 000002      ADD      #2, RO      ; BUILD NEXT ADDRESS
109 011066      ADRTST      RGAIN      ;
110 011112 010037 026734      MOV      RO, SMR      ; SAVE SMR REG ADDRESS
111 011116 062700 000002      ADD      #2, RO      ; BUILD NEXT ADDRESS
112 011122      ADRTST      RGAIN      ;
113 011146 010037 026736      MOV      RO, IOR      ; SAVE IOR REG ADDRESS
114 011152 062700 000001      ADD      #1, RO      ; BUILD HIGH-BYTE ADDRESS
115 011156 010037 026740      MOV      RO, IORMB      ; OF IOR
116 011162 062700 000001      ADD      #1, RO      ; BUILD NEXT ADDRESS
117 011166      ADRTST      RGAIN      ;
118 011212 010037 026742      MOV      RO, VSR      ; SAVE VSR REG ADDRESS
119 011216      VECAIN: WRITE      MES2      ; REQUEST VLCTOR ADDRESS OF IEC11-A
120 011226      READ      OCT, RO      ; INPUT REQUEST
121 011234      IF RO NE #0 GOTO 28      ;
122 011244 012700 000270      MOV      #270, RO      ; SAVE DEFAULT VALUE
123 011250      28: IF RO LO #40 GOTO VECAIN      ;
124 011260      IF RO HI #774 GOTO VECAIN      ;
125 011270 020077 015446      CMP      RO, BVSR      ; IS INPUT EQUAL TO VSR ?
126 011274 001406      BEQ      SAV      ; IF YES, SAVE INPUT AND CONTINUE
127 011276      WRITE      MES6      ; IF NOT, WRITE MESSAGE
128 011306 000137 011216      JMP      VECAIN      ; REENTER
129 011312 010037 026756      SAV: MOV      RO, VECA      ; SAVE IEC11 A VECTOR ADDRESS
130 011316      ADDRIN: WRITE      MES5      ; REQUEST IEC11 A BUS ADDRESS
131 011326      READ      OCT, RO      ; INPUT
132 011334      IF RO HI #36 GOTO ADDRIN      ;
133 011344      IF RO EQ #0 GOTO DEFA      ;

```

```

134 011354          IF RO EQ #35 GOTO DEFA
135 011366 010037 026762      MOV      RO,IECAAD
136 011372 000403          BR        RGBIN
137 011374 012737 000035 026762  DEFA:  MOV      #35,IECAAD
138 011402          RGBIN:  WRITE     MES3
139 011412          READ     OCT,RO
140 011420          IF RO NE #0 GOTO 41
141 011430 012700 160020      MOV      #160020,RO
142 011434 032700 000007      41:      BIT      #7,RO
143 011440 001360          BNE      RGBIN
144 011442          IF RO LO #160020 GOTO RGBIN
145 011452          IF RO HI #177776 GOTO RGBIN
146 011462          ADRTST   RGBIN
147 011506 010037 026744      MOV      RO,CSR
148 011512 062700 000002      ADD      #2,RO
149 011516          ADRTST   RGBIN
150 011542 010037 026746      MOV      RO,BCR
151 011546 062700 000002      ADD      #2,RO
152 011552          ADRTST   RGBIN
153 011576 010037 026750      MOV      RO,BAR
154 011602 062700 000002      ADD      #2,RO
155 011606          ADRTST   RGBIN
156 011632 010037 026752      MOV      RO,MCR
157 011636 062700 000001      ADD      #1,RO
158 011642 010037 026754      MOV      RO,MCRMB
159 011646          VECBIN:  WRITE     MES4
160 011656          READ     OCT,RO
161 011664          IF RO NE #0 GOTO 51
162 011674 012700 000274      MOV      #274,RO
163 011700          51:      IF RO LO #40 GOTO VECBIN
164 011710          IF RO HI #774 GOTO VECBIN
165 011720 010037 026760      MOV      RO,VECB
166 011724          BDDRIN:  WRITE     MES7
167 011734          READ     OCT,RO
168 011742          IF RO HI #36 GOTO BDDRIN
169 011752          IF RO EQ #0 GOTO DEFB
170 011762          IF RO EQ #36 GOTO DEFB
171 011774 010037 026764      MOV      RO,IECBAD
172 012000 000403          BR        $..SPC
173 012002 012737 000036 026764  DEFB:  MOV      #36,IECBAD

```

```

;
; SAVE IEC11-A BUS ADDRESS
; GOTO GET NEXT REGISTER ADDRESS
; SAVE DEFAULT VALUE
; ENTER FIRST REG OF IEC11 B
; INPUT REQUEST
;
; SAVE DEFAULT VALUE
; CHECK THREE LSB OF INPUT
; ARE ZERO
;
;
; SAVE CSR REG ADDRESS
; BUILD NEXT REG ADDRESS
;
; SAVE BCR REG ADDRESS
; BUILD NEXT REG ADDRESS
;
; SAVE BAR REG ADDRESS
; BUILD NEXT REG ADDRESS
;
; SAVE MCR REG ADDRESS
; BUILD MCR-HIGHBYTE
;
; REQUEST IEC11 B VECTOR
; INPUT REQUEST
;
; SAVE DEFAULT VALUE
;
;
; SAVE IEC11 B VECTOR ADDRESS
; REQUEST IEC11 B BUS ADDRESS
; INPUT
;
;
; SAVE IEC11 B BUS ADDRESS
; GOTO TEST SELECT
; SAVE DEFAULT VALUE

```

175				.SBTTL	TEST SELECT ROUTINE	
176	012010			\$...SPC: REMVEC	#VECA	:
177	012016			REMVEC	#VECB	:
178	012024	005737	007152	TST	OUTFLG	: WAIT FOR ANY OUTPUT COMPLETE
179	012030	001375		BNE	--4	:
180	012032	000005		RESET		: CLEAR THE WORLD
181	012034			SET6	#TKS	: RE-ENABLE TTY
182	012042			LAB11: WRITE	MESNEX	: REQUEST NEXT TEST TO RUN
183	012052			READ	OCT,RO	: WAIT FOR INPUT
184	012060			IF RO HI #7 GOTO LAB11		:
185	012070			IF RO EQ #0 GOTO TESTAL		:
186	012100	012737	177777	MOV	#177777,TSTREP	: LOOP ON SINGLE TESTS
187	012106	006300		ASL	RO	: BUILD WORD OFFSET INTO TABLE
188	012110	010027		MOV	RC,(PC).	: AND SAVE
189	012112	000000		OFFSET: .WORD	0	: HOLDS CURRENT TEST NUMBER
190	012114	013737	012112	SINGLE: MOV	OFFSET,OFFCNT	:
191	012122	062737	012220	ADD	#TT,OFFCNT	:
192	012130	017137	000104	MOV	#OFFCNT,OFFCNT	:
193	012136	004777	000076	JSR	PC,#OFFCNT	: GOTO SELECTED TEST
194	012142	000764		BR	SINGLE	: REPEAT TEST
195	012144	012737	000000	TESTAL: MOV	#0,TSTREP	: NO LOOP ON SINGLE TESTS
196	012152	012727	012222	MOV	#TT+2,(PC).	: GET ADDRESS OF FIRST TEST
197	012156	000000		SEQPNT: .WORD	0	: POINTER FOR TEST IN PROGRESS
198	012160	013737	012156	SEQ: MOV	SEQPNT,OFFCNT	: LOAD ADDRESS OF NEXT TEST
199	012166	017737	000046	MOV	#OFFCNT,OFFCNT	:
200	012174	004777	000040	JSR	PC,#OFFCNT	: GOTO EXECUTE TEST
201	012200	023727	012156	CMP	SEQPNT,#TTE-4	: ALL DONE ?
202	012206	001156		BEQ	TESTAL	: IF EQ, YES
203	012210	062737	0002	ADD	#2,SEQPNT	: POINT TO NEXT TEST
204	012216	000760		BR	SEQ	: AND DO NEXT
205	012220	000000	012242	TT: .WORD	0,TEST1,TEST2,TEST3,TEST4,TEST5,TEST6,TEST7	
	012226	014352	017350			
	012234	021350	022206			
206	012240	000000		OFFCNT: .WORD	0	:
207	012242			TTE:		:

209						.SBTTL	TEST 1: REGISTER STATIC TEST	
210	012242					WRITE	ME=8	IDENTIFY TEST 1
211	012252	104000				SCOPE		
212	012254					MCLEAR		
213	012270	052777	000161	014446		BIS	#161,BCSR	SET ALL R/W BITS IN CSR
214	012276	052777	177777	014442		BIS	#177777,BCR	SET ALL R/W BITS IN BCR
215	012304	052777	177777	014436		BIS	#177777,BAR	SET ALL R/W BITS IN BAR
216	012312	052777	137777	014432		BIS	#137777,MCR	SET ALL R/W BITS IN MCR
217	012320					MCLEAR		
218	012334	022777	000000	014370		CMP	#0,BCIR	ALL BITS RESET IN CIR ?
219	012342	001417				BEQ	1:	IF YES,CONTINUE
220	012344					ADP	#0,BCIR	
221	012376					ERROR	1.	
222	012402	022777	000000	014324	1:	CMP	#0,BSMR	ALL BITS RESET IN SMR ?
223	012410	001417				BEQ	2:	IF YES,CONTINUE
224	012412					ADP	#0,BSMR	
225	012444					ERROR	2.	
226	012450	022777	000000	014260	2:	CMP	#0,BIOR	ALL BITS RESET IN IOR ?
227	012456	001417				BEQ	3:	IF YES,CONTINUE
228	012460					ADP	#0,BIOR	
229	012512					ERROR	3.	
230	012516	022777	000000	014220	3:	CMP	#0,BCSR	ALL BITS RESET IN CSR ?
231	012524	001417				BEQ	4:	IF YES,CONTINUE
232	012526					ADP	#0,BCSR	
233	012560					ERROR	4.	
234	012564	022777	000000	014154	4:	CMP	#0,BCR	ALL BITS RESET IN BCR ?
235	012572	001417				BEQ	5:	IF YES,CONTINUE
236	012574					ADP	#0,BCR	
237	012626					ERROR	5.	
238	012632	022777	000000	014110	5:	CMP	#0,BAR	ALL BITS RESET IN BAR ?
239	012640	001417				BEQ	6:	IF YES,CONTINUE
240	012642					ADP	#0,BAR	
241	012674					ERROR	6.	
242	012700	022777	000000	014044	6:	CMP	#0,MCIR	ALL BITS RESET IN MCR ?
243	012706	001417				BEQ	7:	IF YES,CONTINUE
244	012710					ADP	#0,MCIR	
245	012742					ERROR	7.	
246	012746	012746	000060		7:	MOV	#60,-(SP)	SAVE BIT MASK
247	012752	012746	026744			MOV	#CSR,(SP)	SAVE REGISTER
248	012756	004737	031100			CALL	REGTST	TEST R/W BITS
249	012762	012746	000100			MOV	#100,-(SP)	SAVE BIT MASK
250	012766	012746	026744			MOV	#CSR,-(SP)	SAVE REGISTER
251	012772	004737	031100			CALL	REGTST	TEST R/W BITS
252	012776	012777	000010	013740		MOV	#BIT3,BCSR	CLEAR REGISTERS IN IEC11 B
253	013004	012746	177777			MOV	#177777,(SP)	SAVE BIT MASK
254	013010	012746	026746			MOV	#BCR,-(SP)	SAVE REGISTER
255	013014	004737	031100			CALL	REGTST	TEST R/W BITS
256	013020	012746	177777			MOV	#177777,-(SP)	SAVE BIT MASK
257	013024	012746	026750			MOV	#BAR,(SP)	SAVE REGISTER
258	013030	004737	031100			CALL	REGTST	TEST R/W BITS
259	013034	012746	137777			MOV	#137777,(SP)	SAVE BIT MASK
260	013040	012746	026752			MOV	#MCR,-(SP)	SAVE REGISTER
261	013044	004737	031100			CALL	REGTST	TEST R/W BITS
262	013050	005737	027024			TST	TSTREP	TEST LOOP SELECTED ?
263	013054	001402				BEQ	8:	IF NOT,LEAVE THE TEST
264	013056	000137	012252			JMP	REP1	
265	013062				8:	WRITE	MESEND	END OF TEST

G8

TEST 1: REGISTER STATIC TEST

MACRO M1200

30 MAR 84 16:11 PAGE 32 1

SEQ 97

266 013072 000207

RETURN

1

268					TEST2:	SBTTL	TEST 2: TALKER AND LISTENER FUNCTION TEST	
269	013074				WRITE	MESS9	IDENTIFY TEST 2	
270	013104	104000			SCOPE		LOOP FOR ERROR 3	
271	013106				MCLEAR		TEST LOOP ENTRY	
272	013122	052777	000001	013602	BIS	#BIT0,BCIR	SET SACS	
273	013130	052777	000100	013576	BIS	#BIT6,BSMR	SET SIC	
274	013136	005037	027006		CLR	CNT1		
275	013142	032777	020000	013564	11:	BIT	#BIT13,BSMR	IS SIAS STILL SET ?
276	013150	001405			BEQ	21	IF NOT,CONTINUE	
277	013152	005237	027006		INC	CNT1	IF YES,WAIT A WHILE	
278	013156	001371			BNE	11		
279	013160				ERROR	9.		
280	013164	032777	000400	013542	21:	BIT	#BIT8,BSMR	IS CACS SET ?
281	013172	001002			BNE	31		
282	013174				ERROR	9.		
283	013200	042777	177400	013524	31:	BIC	#177400,BCIR	
284	013206	013700	026766		MOV	TAIND,RO	BUILD MTA B	
285	013212	063700	026764		ADD	IECBAD,RO	"	
286	013216	110077	013514		MOVB	RO,BIOR	SET B TO TADS	
287	013222	005037	027006		CLR	CNT1		
288	013226	032777	100000	013476	41:	BIT	#BIT15,BCIR	DATA ACCEPTED ?
289	013234	001007			BNE	51	IF YES,CONTINUE	
290	013236	005237	027006		INC	CNT1	IF NOT,WAIT A WHILE	
291	013242	005737	027006		TST	CNT1	WAITLOOP FINISHED ?	
292	013246	001367			BNE	41		
293	013250				ERROR	10.		
294	013254	042777	100000	013450	51:	BIC	#BIT15,BCIR	CLEAR "DATA ACC" IN CIR
295	013262	052777	000004	013444	BIS	#BIT2,BSMR	GOTO STANDBY STATE,B TO TACS	
296	013270	022777	001204	013446	CHP	#1204,BCSR	IS TACS,STATE CH AND INT IN B SET?	
297	013276	001420			BEQ	LAB1	IF YES CONTINUE	
298	013300				ADP	#1204,BCSR		
299	013332				ERROR	11.		
300	013336	104000			SCOPE		LOOP FOR ERROR 4	
301	013340	042777	001000	013376	LAB1:	BIC	#BIT9,BCSR	CLEAR STATE CHANGE IN CSR
302	013346	022777	000004	013370	CHP	#4,BCSR	TACS SET,STATE CH AND INT CLEAR?	
303	013354	001420			BEQ	LAB2	IF YES,CONTINUE	
304	013356				ADP	#4,BCSR		
305	013410				ERROR	12.		
306	013414	104000			SCOPE		LOOP FOR ERROR 5	
307	013416	052777	000002	013310	LAB2:	BIS	#BIT1,BSMR	LEAVE STANDBY STATE
308	013424	022777	001200	013312	CHP	#1200,BCSR	STATE CH AND INT SET,TACS CLEAR?	
309	013432	001420			BEQ	LAB3	IF YES CONTINUE	
310	013434				ADP	#1200,BCSR		
311	013466				ERROR	13.		
312	013472	104000			SCOPE			
313	013474	042777	001000	013242	LAB3:	BIC	#BIT9,BCSR	CLEAR STATE CHANGE IN CSR
314	013502	013700	026766		MOV	TAIND,RO	BUILD MTA A	
315	013506	063700	026762		ADD	IECAAD,RO	"	
316	013512	110077	013220		MOVB	RO,BIOR	A SETS ITSELF TO TACS,UNADDR. OTHER	
317	013516				DATAACC			
318	013534	052777	000004	013172	BIS	#BIT2,BSMR	GO TO STANDBY STATE	
319	013542	022777	000000	013174	CHP	#0,BCSR	TACS CLEAR ?	
320	013550	001420			BEQ	LAB4	IF YES,CONTINUE	
321	013552				ADP	#0,BCSR		
322	013604				ERROR	14.		
323	013610	104000			SCOPE		LOOP FOR ERROR 7	
324	013612	052777	000002	013114	LAB4:	BIS	#BIT1,BSMR	LEAVE STANDBY STATE

325	013620	013700	026766		MOV	TAINO,RO
326	013624	063700	026764		ADD	IECBAD,RO
327	013630	110077	013102		MOVB	RO,BIOR
328	013634				DATACC	
329	013652	013700	026770		MOV	LIIND,RO
330	013656	063700	026764		ADD	IECBAD,RO
331	013662	110077	013050		MOVB	RO,BIOR
332	013666				DATACC	
333	013704	052777	000004	013022	BIS	#BIT2,BSMR
334	013712	022777	001202	013024	CMP	#1202,BCSR
335	013720	001417			BEQ	LAB5
336	013722				ADP	#1202,BCSR
337	013754				ERROR	15.
338	013760	104000			SCOPE	
339	013762	042777	001000	012754	BIC	#BIT9,BCSR
340	013770	022777	000002	012746	CMP	#2,BCSR
341	013776	001417			BEQ	LAB6
342	014000				ADP	#2,BCSR
343	014032				ERROR	16.
344	014036	104000			SCOPE	
345	014040	052777	000002	012666	BIS	#BIT1,BSMR
346	014046	022777	001200	012670	CMP	#1200,BCSR
347	014054	001417			BEQ	LAB7
348	014056				ADP	#1200,BCSR
349	014110				ERROR	17.
350	014114	042777	001000	012622	BIC	#BIT9,BCSR
351	014122	104000			SCOPE	
352	014124	012700	000077		MOV	#77,RO
353	014130	110077	012602		MOVB	RO,BIOR
354	014134				DATACC	
355	014152	052777	000004	012554	BIS	#BIT2,BSMR
356	014160	032777	000002	012556	BIT	#BIT1,BCSR
357	014166	001403			BEQ	LAB8
358	014170				ERROR	18.
359	014174	104000			SCOPE	
360	014176	052777	000002	012530	BIS	#BIT1,BSMR
361	014204	013700	026770		MOV	LIIND,RO
362	014210	063700	026764		ADD	IECBAD,RO
363	014214	110077	012516		MOVB	RO,BIOR
364	014220				DATACC	
365	014236	013700	026766		MOV	TAINO,RO
366	014242	063700	026764		ADD	IECBAD,RO
367	014246	110077	012464		MOVB	RO,BIOR
368	014252	052777	000004	012454	BIS	#BIT2,BSMR
369	014260	022777	001204	012456	CMP	#1204,BCSR
370	014266	001417			BEQ	LAB9
371	014270				ADP	#1204,BCSR
372	014322				ERROR	19.
373	014326	005737	027024		TST	TSTREP
374	014332	001402			BEQ	11
375	014334	000137	013106		JMP	REP2
376	014340				WRITE	MESND
377	014350	000207			RETURN	

```

: BUILD MTA B
:
: SET B TO TADS
:
: BUILD MLA B
:
: SET B TO LADS
:
: GOTO STANDBY STATE,B TO LACS
: LACS,STATE CHG AND INT SE* ?
: IF YES,CONTINUE
: ELSE ERROR
:
: LOOP FOR ERROR 8
: CLEAR STATE CHANGE IN CSR
: LACS SET,STATE CHANGE AND INT RESET :
: IF TRUE,CONTINUE
:
:
: LOOP FOR ERROR 9
: LEAVE STANDBY STATE
: STATE CHGE AND INT SET,LACS CLEAR ?
: IF YES,CONTINUE
:
:
: CLEAR STATE CHGE AND INT
: LOOP FOR ERROR 10
:
: UNLISTEN IEC11 B
:
: GOTO STANDBY STATE
: IS LACS RESET ?
: IF YES,CONTINUE
:
: LOOP FOR ERROR 11
: LEAVE STANDBY STATE
: BUILD MLA B
:
: SET B TO LADS
:
: BUILD MTA B
:
: SET B TO TADS
: GOTO STANDBY STATE,SET B TO TACS
: TACS,STATE CHGE AND INT SET ?
: IF YES,CONTINUE
:
:
: TEST LOOP SELECTED ?
: IF NOT,LEAVE THE TEST
:
: END OF TEST

```

```

379 .SBTTL TEST 3: GENERAL INTERRUPT AND DMA FUNCTION TEST
380 014352 TEST3: WRITE MES10
381 014362 005737 007152 TST OUTFLG ; WAIT FOR ANY OUTPUT COMPLETE
382 014366 001375 BNE .-4 ; ...
383 014370 104000 SCOPE
384 014372 REP3: MCLEAR
385 014406 CACS
386 014450 SETVEC #VECB,#INTTST,#340
387 014466 013700 026770 MOV LIIND,RO ; BUILD MLA B
388 014472 063700 026764 ADD IECDAD,RO ;
389 014476 110077 012234 MOVB RO,#IOR ; SET B TO LADS
390 014502 DATACC
391 014520 013700 026766 MOV TAINO,RO ; BUILD MTA A
392 014524 063700 026762 ADD IECAAD,RO ;
393 014530 110077 012202 MOVB RO,#IOR ; SET A TO TADS
394 014534 DATACC
395 014552 052777 000004 012154 BIS #BIT2,#SMR ; GOTO STANDBY ,SET A TO TACS,B TO LACS
396 014560 005037 027022 CLR INTFLG ;
397 014564 052777 000100 012152 STCHGE: BIS #BIT6,#CSR ; ENABLE INTERRUPT
398 014572 012737 177000 027006 MOV #177000,CNT1 ; PREPARE DELAY
399 014600 005737 027022 1$: TST INTFLG ; HAS INTERRUPT OCCURRED ?
400 014604 001010 BNE 2$ ; IF YES,CONTINUE
401 014606 005237 027006 INC CNT1 ;
402 014612 001372 BNE 1$ ;
403 014614 042777 000100 012122 BIC #BIT6,#CSR ; DISABLE INTERRUPTS
404 014622 ERROR 20. ;
405 014626 032777 000100 012110 2$: BIT #100,#CSR ; IS INTERRUPT ENABLE BIT RESET ?
406 014634 001402 BEQ 3$ ; IF YES,CONTINUE
407 014636 ERROR 21. ;
408 014642 022777 001202 012074 3$: CMP #1202,#CSR ; CAUSED STATE CHGE THE INTERRUPT ?
409 014650 001420 BEQ NEXBIT ; IF YES,CONTINUE
410 014652 ADP #1202,#CSR ;
411 014704 ERROR 22. ;
412 014710 104000 SCOPE ;
413 014712 NEXBIT: SETVEC #VECB,#INTSRV,#340 ;
414 014730 005037 027016 CLR NEXFLG ;
415 014734 042777 177400 012002 BIC #177400,#CSR ; CLEAR INTERRUPT CAUSING BITS
416 014742 022737 157776 007022 CMP #157776,MEMEND ;
417 014750 001121 BNE NEX1 ;
418 014752 022737 000003 007024 CMP #3,MEMEND-2 ; 128K MEMORY ?
419 014760 001115 BNE NEX1 ;
420 014762 012737 026722 000004 MOV #NEXMEM,#04 ; SETUP VECTOR
421 014770 052777 000060 011746 BIS #60,#CSR ; ENABLE ADDRESSING OVER 32K
422 014776 012700 160000 MOVB #160000,RO ; TRY FIRST ADDRESS IN I/O PAGE
423 015002 005710 1$: TST (RO) ; NON EXISTANT ADDRESS?
424 015004 005737 027016 TST NEXFLG ;
425 015010 001003 BNE 2$ ;
426 015012 062700 000002 ADD #2,RO ; IF NOT,TRY NEXT ONE
427 015016 000771 BR 1$ ;
428 015020 012737 000006 000004 2$: MOV #6,#04 ; REBUILD VECTOR
429 015026 010077 011716 MOV RO,#BAR ; NEX-ADDRESS TO BAR
430 015032 012777 000001 011706 MOV #1,#BCR ; PREPARE BCR
431 015040 052777 000101 011676 BIS #101,#CSR ; ENABLE INTERRUPT AND FUNCTION (DMA)
432 015046 112777 000252 011662 MOVB #252,#IOR ; DUMMY VALUE TO OUTPUT REGISTER
433 015054 012737 177000 027006 MOV #177000,CNT1 ; PREPARE DELAY
434 015062 032777 100000 011642 6$: BIT #BIT15,#CIR ; DATA ACCEPTED ?
435 015070 001007 BNE 3$ ; IF YES,CONTINUE

```

436	015072	005237	027006		INC	CNT1		; IF NOT, WAIT A WHILE
437	015076	005737	027006		TST	CNT1		; WAITLOOP FINISHED ?
438	015102	001367			BNE	6:		
439	015104				ERROR	23.		
440	015110	032777	000100	011626	3:	BIT	#BIT6, @CSR	; HAS INTERRUPT OCCURRED ?
441	015116	001410			BEQ	4:		; IF YES, CONTINUE
442	015120	005237	027006		INC	CNT1		
443	015124	001371			BNE	3:		
444	015126	042777	000100	011610	BIC	#BIT6, @CSR		; DISABLE INTERRUPTS
445	015134				ERROR	24.		
446	015140	022777	100262	011576	4:	CMP	#100262, @CSR	; CAUSED NXM THE INTERRUPT ?
447	015146	001417			BEQ	5:		; IF YES, CONTINUE
448	015150				ADP	#100262, @CSR		
449	015202				ERROR	25.		
450	015206	000137	016216		5:	JMP	BCOVFL	
451	015212	104000			SCOPE			
452	015214	042777	177760	011522	NEX1:	BIC	#177760, @CSR	; CLEAR INTERRUPT CAUSING BITS
453	015222	032737	000002	007024		BIT	#2, MEMEND+2	; MEMORY EXISTS ?
454	015230	001075			BNE	NEX2		
455	015232	052777	000040	011504	BIS	#40, @CSR		; SELECT BUS ADDRESS 400000
456	015240	012777	000000	011502	MOV	#0, @BAR		
457	015246	012777	000001	011472	MOV	#1, @BCR		; PREPARE BCR
458	015254	052777	000101	011462	BIS	#101, @CSR		; ENABLE INTERRUPT AND FUNCTION (DMA)
459	015262	112777	000252	011446	MOVB	#252, @IOR		; DUMMY VALUE TO OUTPUT REGISTER
460	015270	012737	177000	027006	MOV	#177000, CNT1		; PREPARE DELAY
461	015276	032777	100000	011426	5:	BIT	#BIT15, @CIR	; DATA ACCEPTED ?
462	015304	001007			BNE	1:		; IF YES, CONTINUE
463	015306	005237	027006		INC	CNT1		; IF NOT, WAIT A WHILE
464	015312	005737	027006		TST	CNT1		; WAITLOOP FINISHED ?
465	015316	001367			BNE	5:		
466	015320				ERROR	26.		
467	015324	032777	000100	011412	1:	BIT	#BIT6, @CSR	; HAS INTERRUPT OCCURRED ?
468	015332	001410			BEQ	2:		; IF YES, CONTINUE
469	015334	005237	027006		INC	CNT1		
470	015340	001371			BNE	1:		
471	015342	042777	000100	011374	BIC	#BIT6, @CSR		; DISABLE INTERRUPTS
472	015350				ERROR	27.		
473	015354	022777	100242	011362	2:	CMP	#100242, @CSR	; CAUSED NXM THE INTERRUPT ?
474	015362	001420			BEQ	NEX2		; IF YES, CONTINUE
475	015364				ADP	#100242, @CSR		
476	015416				ERROR	28.		
477	015422	104000			SCOPE			
478	015424	042777	177760	011312	NEX2:	BIC	#177760, @CSR	
479	015432	032737	000001	007024		BIT	#1, MEMEND+2	; MEMORY EXISTS ?
480	015440	001072			BNE	NEX3		
481	015442	052777	000020	011274	BIS	#20, @CSR		; SELECT BUS ADDRESS 200000
482	015450	012777	000000	011272	MOV	#0, @BAR		
483	015456	052777	000101	011260	BIS	#101, @CSR		; ENABLE INTERRUPT AND FUNCTION (DMA)
484	015464	112777	000252	011244	MOVB	#252, @IOR		; DUMMY VALUE TO OUTPUT REGISTER
485	015472	012737	177000	027006	MOV	#177000, CNT1		; PREPARE DELAY
486	015500	032777	100000	011224	5:	BIT	#BIT15, @CIR	; DATA ACCEPTED ?
487	015506	001007			BNE	2:		; IF YES, CONTINUE
488	015510	005237	027006		INC	CNT1		; IF NOT, WAIT A WHILE
489	015514	005737	027006		TST	CNT1		; WAITLOOP FINISHED ?
490	015520	001367			BNE	5:		
491	015522				ERROR	29.		
492	015526	032777	000100	011210	2:	BIT	#BIT6, @CSR	; HAS INTERRUPT OCCURRED ?

493	015534	001410			BEQ	3:			; IF YES,CONTINUE
494	015536	005237	027006		INC	CNT1			
495	015542	001371			BNE	2:			
496	015544	042777	000100	011172	BIC	#BIT6,BCSR			; DISABLE INTERRUPTS
497	015552				ERROR	30.			
498	015556	022777	100222	011160	3:	CMP	#100222,BCSR		; CAUSED NXM THE INTERRUPT ?
499	015564	001420			BEQ	NEX3			; IF YES,CONTINUE
500	015566				ADP	#100222,BCSR			
501	015620				ERROR	31.			
502	015624	104000			SCOPE				
503	015626	042777	177760	011110	NEX3:	BIC	#177760,BCSR		
504	015634	032737	100000	007022		BIT	#BIT15,MEMEND		; MEMORY EXISTS ?
505	015642	001067			BNE	NEX4			
506	015644	012777	100000	011076		MOV	#100000,BCAR		; SELECT BUS ADDRESS 100000
507	015652	052777	000101	011064		BIS	#101,BCSR		; ENABLE INTERRUPT AND FUNCTION (DMA)
508	015660	112777	000252	011050		MOVB	#252,BIOR		; DUMMY VALUE TO OUTPUT REGISTER
509	015666	012737	177000	027006		MOV	#177000,CNT1		; PREPARE DELAY
510	015674	032777	100000	011030	5:	BIT	#BIT15,BCIR		; DATA ACCEPTED ?
511	015702	001007			BNE	2:			; IF YES,CONTINUE
512	015704	005237	027006		INC	CNT1			; IF NOT,WAIT A WHILE
513	015710	005737	027006		TST	CNT1			; WAITLOOP FINISHED ?
514	015714	001367			BNE	5:			
515	015716				ERROR	32.			
516	015722	032777	000100	011014	2:	BIT	#BIT6,BCSR		; HAS INTERRUPT OCCURRED ?
517	015730	001410			BEQ	3:			; IF YES,CONTINUE
518	015732	005237	027006		INC	CNT1			
519	015736	001371			BNE	2:			
520	015740	042777	000100	010776		BIC	#BIT6,BCSR		; DISABLE INTERRUPTS
521	015746				ERROR	33.			
522	015752	022777	100202	010764	3:	CMP	#100202,BCSR		; CAUSED NXM THE INTERRUPT ?
523	015760	001420			BEQ	NEX4			; IF YES,CONTINUE
524	015762				ADP	#100202,BCSR			
525	016014				ERROR	34.			
526	016020	104000			SCOPE				
527	016022	042777	177400	010714	NEX4:	BIC	#177400,BCSR		; CLEAR INTERRUPT CAUSING BITS
528	016030	032737	040000	007022		BIT	#BIT14,MEMEND		; MEMORY EXISTS ?
529	016036	001067			BNE	BCOVFL			
530	016040	012777	040000	010702		MOV	#40000,BCAR		; SELECT BUS ADDRESS 40000
531	016046	052777	000101	010670		BIS	#101,BCSR		; ENABLE INTERRUPT AND FUNCTION (DMA)
532	016054	112777	000252	010654		MOVB	#252,BIOR		; DUMMY VALUE TO OUTPUT REGISTER
533	016062	012737	177000	027006		MOV	#177000,CNT1		; PREPARE DELAY
534	016070	032777	100000	010634	5:	BIT	#BIT15,BCIR		; DATA ACCEPTED ?
535	016076	001007			BNE	1:			; IF YES,CONTINUE
536	016100	005237	027006		INC	CNT1			; IF NOT,WAIT A WHILE
537	016104	005737	027006		TST	CNT1			; WAITLOOP FINISHED ?
538	016110	001367			BNE	5:			
539	016112				ERROR	35.			
540	016116	032777	000100	010620	1:	BIT	#BIT6,BCSR		; HAS INTERRUPT OCCURRED ?
541	016124	001410			BEQ	2:			; IF YES,CONTINUE
542	016126	005237	027006		INC	CNT1			
543	016132	001371			BNE	1:			
544	016134	042777	000100	010602		BIC	#BIT6,BCSR		; DISABLE INTERRUPTS
545	016142				ERROR	36.			
546	016146	022777	100202	010570	2:	CMP	#100202,BCSR		; CAUSED NXM THE INTERRUPT ?
547	016154	001420			BEQ	BCOVFL			; IF YES,CONTINUE
548	016156				ADP	#100202,BCSR			
549	016210				ERROR	37.			

550	016214	104000			SCOPE				
551	016216	042777	177760	010520	BCOVFL:	BIC	#177760,BCSR		; CLEAR INTERRUPT CAUSING BITS
552	016224	012777	177777	010514		MOV	#177777,BCR		; PREPARE BCR WITH HIGHEST VALUE
553	016232	013777	027002	010510		MOV	BUFBB,ABAR		; FIRST BUFFER ADDRESS
554	016240	052777	000101	010476		BIS	#101,BCSR		; SET INTERRUPT ENABLE AND FUNCTION BIT
555	016246	112777	000252	010462		MOVB	#252,BIOR		; DUMMY VALUE TO I/O REGISTER
556	016254	012737	177000	027006		MOV	#177000,CNT1		; PREPARE DELAY
557	016262	032777	100000	010442	5:	BIT	#BIT15,BCIR		; DATA ACCEPTED ?
558	016270	001007				BNE	1:		; IF YES,CONTINUE
559	016272	005237	027006			INC	CNT1		; IF NOT,WAIT A WHILE
560	016276	005737	027006			TST	CNT1		; WAITLOOP FINISHED ?
561	016302	001367				BNE	5:		
562	016304					ERROR	38.		
563	016310	032777	000100	010426	1:	BIT	#BIT6,BCSR		; HAS INTERRUPT OCCURRED ?
564	016316	001410				BEQ	2:		; IF YES,CONTINUE
565	016320	005237	027006			INC	CNT1		
566	016324	001371				BNE	1:		
567	016326	042777	000100	010410		BIC	#BIT6,BCSR		; DISABLE INTERRUPTS
568	016334					ERROR	39.		
569	016340	022777	004202	010376	2:	CMP	#4202,BCSR		; CAUSED BC OVFL THE INTERRUPT ?
570	016346	001420				BEQ	ENDBIT		; IF YES,CONTINUE
571	016350					ADP	#4202,BCSR		
572	016402					ERROR	40.		
573	016406	104000			SCOPE				
574	016410	042777	177400	010326	ENDBIT:	BIC	#177400,BCSR		; CLEAR INTERRUPT CAUSING BITS
575	016416	013777	027002	010324		MOV	BUFBB,ABAR		; FIRST BUFFER ADDRESS
576	016424	012777	000001	010314		MOV	#1,BCR		; PREPARE BYTE COUNT REGISTER
577	016432	012777	100777	010312		MOV	#100777,BCMR		; PREPARE MATCH CHARACTER
578	016440	052777	000101	010276		BIS	#101,BCSR		; ENABLE FUNCTION (DMA) AND INTERRUPT
579	016446	112777	000377	010262		MOVB	#377,BIOR		; FIRST BYTE IN I/O REGISTER
580	016454	012737	177000	027006		MOV	#177000,CNT1		; PREPARE DELAY
581	016462	032777	100000	010242	5:	BIT	#BIT15,BCIR		; DATA ACCEPTED ?
582	016470	001007				BNE	1:		; IF YES,CONTINUE
583	016472	005237	027006			INC	CNT1		; IF NOT,WAIT A WHILE
584	016476	005737	027006			TST	CNT1		; WAITLOOP FINISHED ?
585	016502	001367				BNE	5:		
586	016504					ERROR	41.		
587	016510	032777	000100	010226	1:	BIT	#BIT6,BCSR		; HAS INTERRUPT OCCURRED ?
588	016516	001410				BEQ	2:		; IF YES,CONTINUE
589	016520	005237	027006			INC	CNT1		
590	016524	001371				BNE	1:		
591	016526	042777	000100	010210		BIC	#BIT6,BCSR		; DISABLE INTERRUPTS
592	016534					ERROR	42.		
593	016540	022777	002202	010176	2:	CMP	#2202,BCSR		; CAUSED "END" THE INTERRUPT ?
594	016546	001417				BEQ	3:		; IF YES,CONTINUE
595	016550					ADP	#2202,BCSR		
596	016602					ERROR	43.		
597	016606	005737	027024		3:	TST	TSTREP		; TEST LOOP SELECTED ?
598	016612	001403				BEQ	PR		; IF NOT,DO THE REST OF TEST
599	016614	000137	014372			JMP	REP3		
600	016620	104000			SCOPE				
601	016622				PR:	REMEVC	#VECB		
602	016630					MCLEAR			
603	016644					CACS			
604	016706	013700	026766			MOV	TAIND,RO		; BUILD MTA A
605	016712	063700	026762			ADD	IECAAD,RO		
606	016716	110077	010014			MOVB	RO,BIOR		; SET B TO TADS

607 016722				DATAACC			
608 016740	013700	026770		MOV	LIIND,RO		; BUILD MLA B
609 016744	063700	026764		ADD	IECBAD,RO		; "
610 016750	1.0077	007762		MOVB	RO,@IOR		; SET A TO LADS
611 016754				DATAACC			;
612 016772	052777	000004	007734	BIS	@BIT2,@SMR		; GOTO STANDBY STATE
613 017000				SETVEC	@VECB,@PRIOR,@340		;
614 017016				PSET	@340		; SETUP PSW
615 017042	052777	000100	007674	BIS	@BIT6,@CSR		; ENABLE INTERRUPT
616 017050	000240			NOP			;
617 017052	000240			NOP			;
618 017054	012703	017112		MOV	@PRI7,R3		; GET PRINTOUT ADDR. BR7
619 017060				PSET	@300		; ALLOW BR7
620 017104	000240			NOP			;
621 017106	000240			NOP			;
622 017110	000406			BR	PR2		; GOTO PRIORITY 6
623 017112				WRITE	MES13		;
624 017122	000137	017330		JMP	PRIEX		; LEAVE TEST 3
625 017126	012703	017164		MOV	@PRI6,R3		; GET PRINTOUT ADDR. BR6
626 017132				PSET	@240		; ALLOW BR6
627 017156	000240			NOP			;
628 017160	000240			NOP			;
629 017162	000406			BR	PR3		; GOTO PRIORITY 5
630 017164				WRITE	MES14		;
631 017174	000137	017330		JMP	PRIEX		; LEAVE TEST 3
632 017200	012703	017236		MOV	@PRI5,R3		; GET PRINTOUT ADDR. BR5
633 017204				PSET	@200		; ALLOW BR5
634 017230	000240			NOP			;
635 017232	000240			NOP			;
636 017234	000406			BR	PR4		; GOTO PRIORITY 4
637 017236				WRITE	MES15		;
638 017246	000137	017330		JMP	PRIEX		; LEAVE TEST 3
639 017252	012703	017310		MOV	@PRI4,R3		; GET PRINTOUT ADDR. BR4
640 017256				PSET	@140		; ALLOW BR4
641 017302	000240			NOP			;
642 017304	000240			NOP			;
643 017306	000406			BR	PRERR		; GOTO ERROR
644 017310				WRITE	MES16		;
645 017320	000137	017330		JMP	PRIEX		; LEAVE TEST 3
646 017324				ERROR	44.		; NO INTERRUPT OCCURED
647 017330				REMVEC	@VECB		;
648 017336				WRITE	MESEND		; NO OF TEST
649 017346	000207			RETURN			;

```

651 .SBTTL TEST 4: DMA TRANSFER FROM B TO A (B IS TALKER)
652 017350 TEST4: WRITE MES11
653 017360 005737 027024 TST TSTREP ; TEST LOOP SELECTED ?
654 017364 001004 BNE BCIN4 ;
655 017366 013737 027000 027020 MOV BUFFL,BCINP ; USE MAXIMAL BUFFER LENGTH
656 017374 000425 BR REP4 ;
657 017376 BCIN4: WRITE MES17 ; REQUEST BYTE COUNT
658 017406 DUMP OCT,BUFFL ; MAXIMAL INPUT
659 017420 READ OCT,RO ;
660 017426 IF RO HI BUFFL GOTO BCIN4 ;
661 017436 IF RO EQ #0 GOTO BCIN4 ;
662 017444 010037 027020 MOV RO,BCINP ; SAVE BYTE COUNT
663 017450 005737 007152 REP4: TST OUTFLG ; WAIT FOR ANY OUTPUT COMPLETE
664 017454 001375 BNE . 4 ; ...
665 017456 104000 SCOPE ;
666 017460 REMVEC @VECB ;
667 017466 005037 027006 CLR CNT1 ;
668 017472 013700 027002 MOV BUFFB,RO ; PREPARE
669 017476 005001 11: CLR R1 ; BUFFER B
670 017500 110120 21: MOVB R1,(RO). ; WITH
671 017502 005237 027006 INC CNT1 ;
672 017506 023737 027006 027000 CMP CNT1,BUFFL ; CONSECUTIVE
673 017514 001405 BEQ 31 ; VALUES

```

675	017516	005201				INC	R1		FROM
676	017520	022701	000400			CMP	#400,R1		O 377
677	017524	001365				BNE	28		OCTAL
678	017526	000763				BR	18		
679	017530	013700	026774		38:	MOV	BUFAB,R0		SET ALL
680	017534	112720	000377		48:	MOVB	#377,(R0).		BITS
681	017540	020037	027002			CMP	R0,BUFBB		IN
682	017544	001373				BNE	48		BUFFER A
683	017546					MCLEAR			
684	017562					CACS			
685	017624	013700	026766			MOV	YAINO,R0		BUILD MTA B
686	017630	063700	026764			ADD	IECBAD,R0		
687	017634	110077	007076			MOVB	R0,BIOR		SET B TO TADS
688	017640					DATAAC			
689	017656	013700	026770			MOV	LIIND,R0		BUILD MLA A
690	017662	063700	026762			ADD	IECAAD,R0		
691	017666	110077	007044			MOVB	R0,BIOR		SET A TO LADS
692	017672					DATAAC			
693	017710	052777	000004	007016		BIS	#BIT2,BSMR		GOTO STANDBY STATE
694	017716	104000				SCOPE			
695	017720	042777	177400	007016		BIC	#177400,BCSR		CLEAR INTERRUPT CAUSING BITS
696	017726	013777	027002	007014		MOV	BUFBB,SBAR		FIRST BUFFER ADDRESS
697	017734	013700	027020			MOV	BCINP,R0		PREPARE BYTE COUNT REGISTER
698	017740	005400				NEG	R0		
699	017742	010077	007000			MOV	R0,SBAR		
700	017746	013701	026774			MOV	BUFAB,R1		POINTER TO BUFFER A
701	017752					SETVEC	#VEC8,#INTSRV,#340		
702	017770	052777	000101	006746		BIS	#101,BCSR		ENABLE FUNCTION (DMA) AND INTERRUPT
703	017776	005037	027010			CLR	CNT2		
704	020002	005037	027014			CLR	DONE		
705	020006	005037	027006			CLR	CNT1		
706	020012	032777	100000	006712	58:	BIT	#BIT15,BCIR		DATA ACCEPTED ?
707	020020	001013				BNE	DMA4		IF YES,CONTINUE
708	020022	005237	027006			INC	CNT1		IF NOT,WAIT A WHILE
709	020026	005737	027006			TST	CNT1		WAITLOOP FINISHED ?
710	020032	001367				BNE	58		
711	020034					ERROR	45.		
712	020040	032777	100000	006664	ACC4:	BIT	#BIT15,BCIR		DATA ACCEPTED ?
713	020046	001774				BEQ	ACC4		IF YES,CONTINUE
714	020050	032777	000100	006666	DMA4:	BIT	#100,BCSR		HAS INTERRUPT OCCURRED ?
715	020056	001410				BEQ	INTB4		IF YES,GET LAST BYTE
716	020060	042777	100000	006644		BIC	#BIT15,BCIR		
717	020066	117721	006646			MOVB	BIORMB,(R1).		DATA TO BUFFER A
718	020072	005237	027010			INC	CNT2		BYTE COUNTER
719	020076	000760				BR	ACC4		
720	020100	012737	150000	027006	INTB4:	MOV	#150000,CNT1		PREPARE DELAY AFTER INTERRUPT B
721	020106	005237	027006		LAST4:	INC	CNT1		
722	020112	005737	027006			TST	CNT1		
723	020116	001426				BEQ	108		
724	020120	032777	100000	006604		BIT	#BIT15,BCIR		DATA ACCEPTED ?
725	020126	001767				BEQ	LAST4		IF NOT,WAIT
726	020130	005237	027010		98:	INC	CNT2		IF YES,CHECK FOR
727	020134	005737	027014			TST	DONE		CORRECT BYTE COUNT
728	020140	001023				BNE	118		
729	020142	042777	100000	006562		BIC	#BIT15,BCIR		
730	020150	117721	006564			MOVB	BIORMB,(R1).		BYTE TO BUFFER A
731	020154	023737	027010	027020		CMP	CNT2,BCINP		ALL BYTES DONE ?

Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

766 .SBTTL TEST 5: DMA TRANSFER FROM A TO B (B IS LISTENER)
767 020442 TEST5: WRITE MES12 ; TEST LOOP SELECTED ?
768 020452 005737 027024 TST TSTREP ;
769 020456 001004 BNE BCINS ; USE MAXIMAL BUFFER LENGTH
770 020460 013737 027000 027020 MOV BUFL,BCINP ;
771 020466 000425 BR REP5 ; REQUEST BYTE COUNT
772 020470 BCINS: WRITE MES17 ; MAXIMAL INPUT
773 020500 DUMP OCT,BUFL ;
774 020512 READ OCT,RO ;
775 020520 IF RO HI BUFL GOTO BCINS ;
776 020530 IF RO EQ #0 GOTO BCINS ;
777 020536 010037 027020 MOV RO,BCINP ; SAVE BYTE COUNT
778 020542 005737 007152 REP5: TST OUTFLG ; WAIT FOR ANY OUTPUT COMPLETE
779 020546 001375 BNE .-4 ; ...
780 020550 104000 SCOPE ;
781 020552 REMVEC @VECB ;
782 020560 013700 026774 MOV BUFB,RO ; PREPARE
783 020564 005001 1: CLR R1 ; BUFFER A
784 020566 110120 2: MOVB R1,(RO) ; WITH
785 020570 020037 027002 CMP RO,BUFB ; CONSECUTIVE
786 020574 001405 BEQ 3: ; VALUES
787 020576 005201 INC R1 ; FROM
788 020600 022701 000400 CMP #400,R1 ; 0 377
789 020604 001370 BNE 2: ; OCTAL
790 020606 000766 BR 1: ;
791 020610 005037 027006 3: CLR CNT1 ; SET
792 020614 013700 027002 MOV BUFB,RO ; ALL
793 020620 112720 000377 4: MOVB #377,(RO) ; BITS
794 020624 005237 027006 INC CNT1 ; IN
795 020630 023737 027000 027006 CMP BUFL,CNT1 ; BUFFER
796 020636 001370 BNE 4: ; A
797 020640 MCLEAR ;
798 020654 CACS ;
799 020716 013700 026766 MOV TAIN,RO ; BUILD MTA A
800 020722 063700 026762 ADD IECA,RO ;
801 020726 110077 006004 MOVB RO,@IOR ; SET A TO TADS
802 020732 DATACC ;
803 020750 013700 026770 MOV LIIND,RO ; BUILD MLA B
804 020754 063700 026764 ADD IECB,RO ;
805 020760 110077 005752 MOVB RO,@IOR ; SET B TO LADS
806 020764 DATACC ;
807 021002 052777 000004 005724 BIS @BIT2,@SMR ; GOTO STANDBY STATE
808 021010 104000 SCOPE ;
809 021012 042777 177400 005724 BIC #177400,@CSR ; CLEAR INTERRUPT CAUSING BITS
810 021020 013777 027002 005722 MOV BUFB,@BAR ; FIRST BUFFER ADDRESS
811 021026 013700 027020 MOV BCINP,RO ; PREPARE BYTE COUNT REGISTER
812 021032 005400 NEG RO ;
813 021034 010077 005706 MOV RO,@BCR ;
814 021040 013701 026774 MOV BUFB,R1 ; POINTER TO BUFFER A
815 021044 SETVEC @VECB,@INTSRV,#340 ;
816 021062 052777 000101 005654 BIS #101,@CSR ; ENABLE FUNCTION (DMA) AND INTERRUPT
817 021070 112177 005642 5: MOVB (R1),@IOR ; DATA BYTE TO I/O REGISTER
818 021074 DATACC ;
819 021112 032777 000100 005624 BIT #100,@CSR ; HAS INTERRUPT OCCURRED ?
820 021120 001363 BNE 5: ; IF NOT,CONTINUE DMA
821 021122 022777 004202 005614 CMP #4202,@CSR ; CAUSED 'BC OVFL' THE INTERRUPT
822 021130 001417 BEQ 6: ; IF YES,CONTINUE

```

IEC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 37 1
 TEST 5: DMA TRANSFER FROM A TO B (B IS LISTENER)

SEQ 109

823	021132				ADP	04202,BCSR	
824	021164				ERROR	50.	
825	021170	013700	026774	6:	MOV	BUFAB,R0	; PROVIDE FIRST BYTE OF BUFFER A
826	021174	013701	027002		MOV	BUFBB,R1	; PROVIDE FIRST BYTE OF BUFFER B
827	021200	005037	027006		CLR	CNT1	
828	021204	005237	027006	CMP5:	INC	CNT1	; BYTE COUNT
829	021210	122021			CMPB	(R0),.(R1),	; BUFFER A EQUAL TO BUFFER B
830	021212	001440			BEQ	7:	; IF YES,CONTINUE
831	021214	114037	027026		MOVB	-(R0),ASAVE	; IF NOT,SAVE DATA
832	021220	114137	027030		MOVB	-(R1),BSAVE	
833	021224	032737	040000	007160	BIT	#BIT14,PSDSWR	; PRINTOUT ALLOWED ?
834	021232	001030			BNE	7:	
835	021234				WRITE	MES18	
836	021244				DUMP	OCT,CNT1	; WRITE BYTE COUNT
837	021256				ADP	ASAVE,BSAVE	; DUMP NUMBER
838	021310				ERROR	51.	
839	021314	023737	027006	027020	7:	CMP	CNT1,BCINP
840	021322	001330			BNE	CMP5	; ALL BYTES COMPARED ?
841	021324	005737	027024		TST	TSTREP	; IF NOT,GET NEXT ONE
842	021330	001402			BEQ	8:	; TEST LOOP SELECTED ?
843	021332	000137	020542		JMP	REPS	; IF NOT,LEAVE THE TEST
844	021336			8:	WRITE	MESEND	
845	021346	000207			RETURN		; END OF TEST

847						.SBTTL	TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)	
848	021350					TEST6: WRITE	MES19	
849	021360	005737	007152			TST	OUTFLG	; WAIT FOR ANY OUTPUT COMPLETE
850	021364	001375				BNE	--4	; ...
851	021366					MCLEAR		
852	021402					CACS		
853	021444	013700	026766			MOV	TAIND,RO	; BUILD MTA A
854	021450	063700	026762			ADD	IECAAD,RO	; "
855	021454	110077	005256			MOVB	RO,BIOR	; SET A TO TADS
856	021460					DATAAC		
857	021476	013700	026770			MOV	LIIND,RO	; BUILD MLA B
858	021502	063700	026764			ADD	IECBAD,RO	; "
859	021506	110077	005224			MOVB	RO,BIOR	; SET B TO LADS
860	021512					DATAAC		
861	021530	052777	000004	005176		BIS	#BIT2,BSMR	; GOTO STANDBY STATE
862	021536	042777	177400	005200	18:	BIC	#177400,BCSR	; CLEAR HIGH BYTE IN CSR
863	021544					SETVEC	#VECB,#INTSRV,#340	
864	021562	104000				SCOPE		
865	021564	012737	000001	027010	REP6:	MOV	#1,CNT2	; FIRST COUNT OF MCR
866	021572	005037	027012			CLR	CHAR	; FIRST CHARACTER OF MCR
867	021576	042777	177400	005140	BEG6:	BIC	#177400,BCSR	; CLEAR INTERRUPT CAUSING BITS
868	021604	005001				CLR	R1	
869	021606	013700	026774			MOV	BUFAB,RO	; POINTER TO BUFFER A
870	021612	113720	027012		18:	MOVB	CHAR,(R0).	; FILL BUFFER A
871	021616	105201				INCB	R1	; WITH ONE CHARACTER
872	021620	123701	027010			CMPS	CNT2,R1	; LIMITED BY MATCH COUNT
873	021624	001372				BNE	18	
874	021626	013701	026774			MOV	BUFAB,R1	; PROVIDE FIRST BUFFER ADDRESS A
875	021632	013777	027002	005110		MOV	BUFB8,BBAR	; PREPARE BAR
876	021640	012777	000001	005100		MOV	#1,BBCR	; DUMMY VALUE TO BCR
877	021646	113777	027010	005100		MOVB	CNT2,BMCRMB	; COUNT TO REGISTER
878	021654	113777	027012	005070		MOVB	CHAR,BMCR	; CHARACTER TO REGISTER
879	021662	152777	000200	005064		BISB	#BIT7,BMCRMB	; ENABLE MATCH
880	021670	052777	000101	005046		BIS	#101,BCSR	; ENABLE INTERRUPT AND DMA
881	021676	112177	005034		DMA6:	MOVB	(R1),BIOR	; ONE BYTE TO IOR
882	021702					DATAAC		
883	021720	032777	000100	005016		BIT	#100,BCSR	; HAS INTERRUPT OCCURRED ?
884	021726	001363				BNE	DMA6	; IF NOT,CONTINUE DMA
885	021730	022777	002202	005006		CMPS	#2202,BCSR	; CAUSED "END" THE INTERRUPT
886	021736	001417				BEQ	38	; IF YES,CONTINUE
887	021740					ADP	#2202,BCSR	
888	021772					ERROR	52.	
889	021776	013700	026774		38:	MOV	BUFAB,RO	; PROVIDE FIRST BYTE OF BUFFER A
890	022002	013701	027002			MOV	BUFB8,R1	; PROVIDE FIRST BYTE OF BUFFER B
891	022006	005002				CLR	R2	
892	022010	020237	027010		CMP6:	CMPS	R2,CNT2	; END OF BYTE COUNT ?
893	022014	001442				BEQ	68	
894	022016	005202				INC	R2	; BYTE COUNT
895	022020	122021				CMPS	(R0), (R1).	; BUFFER A EQUAL TO BUFFER B
896	022022	001772				REQ	CMP6	; IF YES,CONTINUE
897	022024	114037	027026			MOVB	-(R0),ASAVE	; IF NOT,SAVE DATA
898	022030	114137	027030			MOVB	-(R1),BSAVE	
899	022034	032737	040000	007160		BIT	#BIT14,PSDSWR	; PRINTOUT ALLOWED ?
900	022042	001010				BNE	58	
901	022044					WRITE	MES18	; WRITE BYTE COUNT
902	022054					DUMP	OCT,R2	; DUMP NUMBER
903	022054				58:	ADP	ASAVE,BSAVE	

IEC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 38 1
 TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)

SEQ 111

904	022116				ERROR	53.	
905	022122	005237	027012	64:	INC	CHAR	; NEXT MATCH CHARACTER
906	022126	122737	000400	027012	CMPS	@400,CHAR	; LAST CHARACTER ?
907	022134	001220			BNE	BEG6	
908	022136	005037	027012		CLR	CHAR	; FIRST CHARACTER
909	022142	005237	027010		INC	CNT2	; NEXT MATCH COUNT
910	022146	122737	000100	027010	CMPS	@100,CNT2	; ALL COUNTS DONE ?
911	022154	001402			BEQ	78	
912	022156	000137	021576		JMP	BEG6	
913	022162	005737	027024	78:	TST	TSTREP	; TEST LOOP SELECTED ?
914	022166	001402			BEQ	88	; IF NOT, LEAVE THE TEST
915	022170	000137	021564		JMP	HEP6	
916	022174			88:	WRITE	HESEND	; END OF TEST
917	022204	000207			RETURN		

```

919 .SBTTL TEST 7: SERIAL POLL PROCEDURE TEST
920 022206 TEST7: WRITE MES20
921 022216 005737 007152 TST OUTFLG ; WAIT FOR ANY OUTPUT COMPLETE
922 022222 001375 BNE .4 ; ...
923 022224 SETVEC @VECB,@INTSRV,@340 ;
924 022242 REP7: @VECA,@INTA,@340 ;
925 022260 MCLEAR ;
926 022274 CACS ;
927 022336 104000 SCOPE ;
928 022340 052777 000002 004366 POLLO: BIS @BIT1,@SMR ; GOTO CACS
929 022346 112777 000030 004362 MOVB @30,@IOR ; SERIAL POLL ENABLE
930 022354 013700 026770 MOV LIIND,RO ; BUILD MLA A
931 022360 063700 026762 ADD IECAAC,P ;
932 022364 110077 004346 MOVB RO,@IOR ; SET A TO LADS
933 022370 DATACC ;
934 022406 012701 000000 MOV @0,R1 ; BUILD FIRST TALKER ADDRESS
935 022412 013700 026766 PO: MOV TAINO,RO ;
936 022416 060100 ADD R1,RO ; BUILD TALKER ADDRESS
937 022420 110077 004312 MOVB RO,@IOR ;
938 022424 DATACC ;
939 022442 052777 000004 004264 BIS @BIT2,@SMR ; GOTO STANDBY STATE,A TO LACS
940 022450 STALL ;
941 022500 122777 000000 004232 CMPB @0,@IORMB ; STATUS BYTE IS ZERO ?
942 022506 001416 BEQ 51 ; IF YES,TRY THE NEXT ONE
943 022510 032737 040000 007160 BIT @BIT14,PSDSMR ; PRINTOUT SELECTED ?
944 022516 001012 BNE 51 ;
945 022520 WRITE MES21 ; GIVE INFORMATION ABOUT
946 022530 DUMP OCT,R1 ; ILLEGAL IEC BUS ADDRESS
947 022540 ERROR 54. ;
948 022544 005201 51: INC R1 ; GET NEXT TALKER ADDRESS
949 022546 022701 000037 CMP @37,R1 ; ALL DONE ?
950 022552 001405 BEQ 71 ;
951 022554 052777 000002 004152 BIS @BIT1,@SMR ; IF NOT,GOTO CACS
952 022562 000137 022412 JMP PO ; AND SELECT THE NEXT ONE
953 022566 052777 000002 004140 71: BIS @BIT1,@SMR ; GOTO CACS
954 022574 112777 000031 004134 MOVB @31,@IOR ; DISABLE SERIAL POLL
955 022602 052777 000004 004124 BIS @BIT2,@SMR ; GOTO STANDBY STATE,A TO LACS
956 022610 000240 DONC: NOP ; JUST ENTRY POINT
957 022612 042777 177400 004112 BIC @177400,@CIR ; CLEAR INTERRUPT CAUSING BITS
958 022620 012737 177700 027006 MOV @177700,CNT1 ; PREPARE DELAY
959 022626 052777 000400 004110 BIS @BIT8,@CSR ; REQUEST SERVICE
960 022634 052777 000100 004070 BIS @100,@CIR ; ENABLE INTERRUPT A
961 022642 032777 000100 004062 11: BIT @100,@CIR ; HAS INTERRUPT OCCURRED ?
962 022650 001407 BEQ 21 ; IF YES,CONTINUE
963 022652 005237 027006 INC CNT1 ;
964 022656 005737 027006 TST CNT1 ;
965 022662 001367 BNE 11 ;
966 022664 ERROR 55. ;
967 022670 022777 040201 004034 21: CMP @040201,@CIR ; CAUSED SRQ THE INTERRUPT ?
968 022676 001417 BEQ POLL1 ; IF YES,CONTINUE
969 022700 ADP @040201,@CIR ;
970 022732 ERROR 56. ;
971 022736 104000 POLL1: SCOPE ;
972 022740 005037 027014 CLR DONE ; CLEAR DONE FLAG
973 022744 052777 000002 003762 BIS @BIT1,@SMR ; GOTO CACS
974 022752 112777 000030 003756 MOVB @30,@IOR ; SERIAL POLL ENABLE
975 022760 013700 026770 MOV LIIND,RO ; BUILD MLA A

```

976	022764	063700	026762		ADD	IECAAD,RO	:	"
977	022770	110077	003742		MOVB	RO,BIOR	:	SET A TO LADS
978	022774				DATAACC		:	
979	023012	012701	000000		MOV	#0,R1	:	BUILD FIRST TALKER ADDRESS
980	023016	013700	026766	P1:	MOV	TAIND,RO	:	
981	023022	060100			ADD	R1,RO	:	BUILD TALKER ADDRESS
982	023024	110077	003706		MOVB	RO,BIOR	:	
983	023030				DATAACC		:	
984	023046	052777	000004	003660	BIS	#BIT2,BSMR	:	GOTO STANDBY STATE,A TO LACS
985	023054				STALL		:	
986	023104	122777	000000	003626	CMPS	#0,BIORMB	:	POLLING ADDRESS ?
987	023112	001452			BEQ	58	:	IF NOT,TR: THE NEXT ONE
988	023114	120137	026764		CMPS	R1,IECBAD	:	LEGAL ADDRESS ?
989	023120	001416			BEQ	38	:	IF YES,CONTINUE
990	023122	032737	040000	007160	BIT	#BIT14,PSDSWR	:	PRINTOUT SELECTED ?
991	023130	001012			BNE	38	:	
992	023132				WRITE	MES21	:	GIVE INFORMATION ABOUT
993	023142				DUMP	OCT,R1	:	ILLEGAL IEC BUS ADDRESS
994	023152				ERROR	57.	:	
995	023156	117702	003556	38:	MOVB	BIORMB,R2	:	SAVE STATUS BYTE
996	023162	042702	177400		BIC	#177400,R2	:	
997	023166	022702	000100		CMPS	#100,R2	:	STATUS BYTE OK ?
998	023172	001416			BEQ	48	:	IF YES,CONTINUE
999	023174				ADP	#100,R2	:	
1000	023224				ERROR	58.	:	
1001	023230	012737	177777	027014	48:	MOV	#177777,DONE	SET DONE FLAG
1002	023236	000411			BR	PEND1	:	GOTO FINISH THIS TESTPART
1003	023240	005201			58:	INC	R1	GET NEXT TALKER ADDRESS
1004	023242	022701	000037		CMPS	#37,R1	:	ALL DONE ?
1005	023246	001405			BEQ	PEND1	:	
1006	023250	052777	000002	003456	BIS	#BIT1,BSMR	:	IF NOT,GOTO CACS
1007	023256	000137	023016		JMP	P1	:	AND SELECT THE NEXT ONE
1008	023262	005737	027014	PEND1:	TST	DONE	:	ONE POLLING ADDRESS FOUND AT LEAST ?
1009	023266	001002			BNE	78	:	IF YES,GOTO NEXT PART OF THE TEST
1010	023270				ERROR	59.	:	
1011	023274	104000			78:	SCOPE	:	
1012	023276	052777	000002	003430	BIS	#BIT1,BSMR	:	GOTO CACS
1013	023304	112777	000031	003424	MOVB	#31,BIOR	:	DISABLE SERIAL POLL
1014	023312	052777	000004	003414	BIS	#BIT2,BSMR	:	GOTO STANDBY STATE,A TO LACS
1015	023320	032777	040000	003404	BIT	#BIT14,BCIR	:	SRQ CLEARED BY "SPD" ?
1016	023326	001402			BEQ	DON1	:	IF YES,CONTINUE
1017	023330				ERROR	60.	:	
1018	023334	000240			DON1:	NOP	:	JUST ENTRY POINT
1019	023336				MCLEAR		:	
1020	023352				CACS		:	
1021	023414	013700	026770		MOV	LIIND,RO	:	BUILD MLA B
1022	023420	063700	026764		ADD	IECBAD,RO	:	
1023	023424	110077	003306		MOVB	RO,BIOR	:	SET B TO LADS
1024	023430				DATAACC		:	
1025	023446	013700	026766		MOV	TAIND,RO	:	BUILD MTA A
1026	023452	063700	026762		ADD	IECAAD,RO	:	
1027	023456	110077	003254		MOVB	RO,BIOR	:	SET A TO TADS
1028	023462				DATAACC		:	
1029	023500	052777	000004	003226	BIS	#BIT2,BSMR	:	GOTO STANDBY ,SET A TO TACS,B TO LACS
1030	023506	042777	177760	003230	SPOVFL:	BIC	#177760,BCSR	CLEAR INTERRUPT CAUSING BITS
1031	023514	012777	177777	003224	MOV	#177777,BCR	:	PREPARE BCR WITH HIGHEST VALUE
1032	023522	013777	027002	003220	MOV	BUFBB,BCAR	:	FIRST BUFFER ADDRESS

1033	023530	052777	000101	003206		BIS	#101,@CSR	; SET INTERRUPT ENABLE AND FUNCTION BIT
1034	023536	112777	000252	003172		MOVB	#252,@IOR	; DUMMY VALUE TO I/O REGISTER
1035	023544					DATA		
1036	023562	032777	000100	003154	1:	BIT	#BIT6,@CSR	; HAS INTERRUPT OCCURRED ?
1037	023570	001374				BNE	1:	; IF YES,CONTINUE
1038	023572	052777	000400	003144		BIS	#BIT8,@CSR	; REQUEST SERVICE
1039	023600	104000				SCOPE		
1040	023602	005037	027014		POLL2:	CLR	DONE	; CLEAR DONE FLAG
1041	023606	052777	000002	003120		BIS	#BIT1,@SMR	; GOTO CACS
1042	023614	112777	000137	003114		MOVB	#137,@IOR	; UNTALK IEC11-A
1043	023622					DATA		
1044	023640	112777	000030	003070		MOVB	#30,@IOR	; SERIAL POLL ENABLE
1045	023646	013700	026770			MOV	LIIND,R0	; BUILD MLA A
1046	023652	063700	026762			ADD	IECAD,R0	; " "
1047	023656	110077	003054			MOVB	R0,@IOR	; SET A TO LADS
1048	023662					DATA		
1049	023700	012701	000000			MOV	#0,R1	; BUILD FIRST TALKER ADDRESS
1050	023704	013700	026766		P2:	MOV	TAIND,R0	
1051	023710	060100				ADD	R1,R0	; BUILD TALKER ADDRESS
1052	023712	110077	003020			MOVB	R0,@IOR	
1053	023716					DATA		
1054	023734	052777	000004	002772		BIS	#BIT2,@SMR	; GOTO STANDBY STATE,A TO LACS
1055	023742					STALL		
1056	023772	122777	000000	002740		CMPB	#0,@IORH8	; POLLING ADDRESS ?
1057	024000	001452				BEQ	5:	; IF NOT,TRY THE NEXT ONE
1058	024002	120137	026764			CMPB	R1,IECBAD	; LEGAL ADDRESS ?
1059	024006	001416				BEQ	3:	; IF YES,CONTINUE
1060	024010	032737	040000	007160		BIT	#BIT14,PSDSWR	; PRINTOUT SELECTED ?
1061	024016	001012				BNE	3:	
1062	024020					WRITE	MES21	; GIVE INFORMATION ABOUT
1063	024030					DUMP	OCT,R1	; ILLEGAL IEC-BUS ADDRESS
1064	024040					ERROR	61.	
1065	024044	117702	002670		3:	MOVB	@IORH8,R2	; SAVE STATUS BYTE
1066	024050	042702	177400			BIC	#177400,R2	
1067	024054	022702	000104			CMP	#104,R2	; STATUS BYTE OK ?
1068	024060	001416				BEQ	4:	; IF YES,CONTINUE
1069	024062					ADP	#104,R2	
1070	024112					ERROR	62.	
1071	024116	012737	177777	027014	4:	MOV	#177777,DONE	; SET DONE FLAG
1072	024124	000411				BR	PEND2	; GOTO FINISH THIS TESTPART
1073	024126	005201			5:	INC	R1	; GET NEXT TALKER ADDRESS
1074	024130	022701	000037			CMP	#37,R1	; ALL DONE ?
1075	024134	001405				BEQ	PEND2	
1076	024136	052777	000002	002570		BIS	#BIT1,@SMR	; IF NOT,GOTO CACS
1077	024144	000137	023704			JMP	P2	; AND SELECT THE NEXT ONE
1078	024150	005737	027014		PEND2:	TST	DONE	; ONE POLLING ADDRESS FOUND AT LEAST ?
1079	024154	001002				BNE	7:	; IF YES,GOTO NEXT PART OF THE TES
1080	024156					ERROR	63.	
1081	024162	052777	000002	002544	7:	BIS	#BIT1,@SMR	; GOTO CACS
1082	024170	112777	000031	002540		MOVB	#31,@IOR	; DISABLE SERIAL POLL
1083	024176	052777	000004	002530		BIS	#BIT2,@SMR	; GOTO STANDBY STATE,A TO LACS
1084	024204	000240			DON2:	NOP		; JUST ENTRY POINT
1085	024206					MCLEAR		
1086	024222					CACS		
1087	024264	013700	026770			MOV	LIIND,R0	; BUILD MLA B
1088	024270	063700	026764			ADD	IECBAD,R0	
1089	024274	110077	002436			MOVB	R0,@IOR	; SET B TO LADS

1090	024300					DATACC		:	
1091	024316	013700	026766			MOV	TAINO,RO	:	BUILD MTA A
1092	024322	063700	026762			ADD	IECAAD,RO	:	"
1093	024326	110077	002404			MOVB	RO,@IOR	:	SET A TO TADS
1094	024332					DATACC		:	
1095	024350	052777	000004	002356		BIS	#BIT2,@SMR	:	GOTO STANDBY ,SET A TO TACS,B TO LACS
1096	024356	042777	177400	002360	SPEND:	BIC	#177400,@CSR	:	CLEAR INTERRUPT CAUSING BITS
1097	024364	013777	027002	002356		MOV	BUFBB,@BAR	:	FIRST BUFFER ADDRESS
1098	024372	012777	000001	002346		MOV	#1,@BCR	:	PREPARE BYTE COUNT REGISTER
1099	024400	012777	100777	002344		MOV	#100777,@MCR	:	PREPARE MATCH CHARACTER
1100	024406	052777	000101	002330		BIS	#101,@CSR	:	ENABLE FUNCTION (DMA) AND INTERRUPT
1101	024414	112777	000377	002314		MOVB	#377,@IOR	:	FIRST BYTE IN I/O REGISTER
1102	024422					DATACC		:	
1103	024440	032777	000100	002276	1\$:	BIT	#BIT6,@CSR	:	HAS INTERRUPT OCCURRED ?
1104	024446	001374				BNE	1\$	:	IF YES,CONTINUE
1105	024450	052777	000400	002266		BIS	#BIT8,@CSR	:	REQUEST SERVICE
1106	024456	104000				SCOPE		:	
1107	024460	005037	027014		POLL3:	CLR	DONE	:	CLEAR DONE FLAG
1108	024464	052777	000002	002242		BIS	#BIT1,@SMR	:	GOTO CACS
1109	024472	112777	000137	002236		MOVB	#137,@IOR	:	UNTALK IEC11 A
1110	024500					DATACC		:	
1111	024516	112777	000030	002212		MOVB	#30,@IOR	:	SERIAL POLL ENABLE
1112	024524	013700	026770			MOV	LIIND,RO	:	BUILD MLA A
1113	024530	063700	026762			ADD	IECAAD,RO	:	"
1114	024534	110077	002176			MOVB	RO,@IOR	:	SET A TO LADS
1115	024540					DATACC		:	
1116	024556	012701	000000			MOV	#0,R1	:	BUILD FIRST TALKER ADDRESS
1117	024562	013700	026766		P3:	MOV	TAINO,RO	:	
1118	024566	060100				ADD	R1,RO	:	BUILD TALKER ADDRESS
1119	024570	110077	002142			MOVB	RO,@IOR	:	
1120	024574					DATACC		:	
1121	024612	052777	000004	002114		BIS	#BIT2,@SMR	:	GOTO STANDBY STATE,A TO LACS
1122	024620					STALL		:	
1123	024650	122777	000000	002062		CMPB	#0,@IORMB	:	POLLING ADDRESS ?
1124	024656	001452				BEQ	5\$	:	IF NOT,TRY THE NEXT ONE
1125	024660	120137	026764			CMPB	R1,IECBAD	:	LEGAL ADDRESS ?
1126	024664	001416				BEQ	3\$	:	IF YES,CONTINUE
1127	024666	032737	040000	007160		BIT	#BIT14,PSDSWR	:	PRINTOUT SELECTED ?
1128	024674	001012				BNE	3\$	:	
1129	024676					WRITE	MES21	:	GIVE INFORMATION ABOUT
1130	024706					DUMP	OCT,R1	:	ILLEGAL IEC-BUS ADDRESS
1131	024716					ERROR	64.	:	
1132	024722	117702	002012		3\$:	MOVB	@IORMB,R2	:	SAVE STATUS BYTE
1133	024726	042702	177400			BIC	#177400,R2	:	
1134	024732	022702	000110			CMP	#110,R2	:	STATUS BYTE OK ?
1135	024736	001416				BEQ	4\$	:	IF YES,CONTINUE
1136	024740					ADP	#110,R2	:	
1137	024770					ERROR	65.	:	
1138	024774	012737	177777	027014	4\$:	MOV	#177777,DONE	:	SET DONE FLAG
1139	025002	000411				BR	PEND3	:	GOTO FINISH THIS TESTPART
1140	025004	005201			5\$:	INC	R1	:	GET NEXT TALKER ADDRESS
1141									

1147	025034					ERROR	66.	
1148	025040	052777	000002	001666	7:	BIS	#BIT1, @SMR	; GOTO CACS
1149	025046	112777	000031	001662		MOVB	#31, @IOR	; DISABLE SERIAL POLL
1150	025054	052777	000004	001652		BIS	#BIT2, @SMR	; GOTO STANDBY STATE, A TO LACS
1151	025062	000240			DON3:	NOP		; JUST ENTRY POINT
1152	025064					MCLEAR		
1153	025100					CACS		
1154	025142	013700	026770			MOV	LIIND, RO	; BUILD MLA B
1155	025146	063700	026764			ADD	IECBAD, RO	; "
1156	025152	110077	001560			MOVB	RO, @IOR	; SET B TO LADS
1157	025156					DATAACC		
1158	025174	013700	026766			MOV	TAIND, RO	; BUILD MTA A
1159	025200	063700	026762			ADD	IECAAD, RO	; "
1160	025204	110077	001526			MOVB	RO, @IOR	; SET A TO TADS
1161	025210					DATAACC		
1162	025226	052777	000004	001500		BIS	#BIT2, @SMR	; GOTO STANDBY, SET A TO TACS, B TO LACS
1163	025234	005037	027016		SPNEX:	CLR	NEXFLG	; CLEAR INTERRUPT CAUSING BITS
1164	025240	042777	177760	001476		BIC	#177760, @CSR	
1165	025246	012737	026722	000004		MOV	#NEXMEM, @4	
1166	025254	052777	000060	001462		BIS	#60, @CSR	
1167	025262	012700	160000			MOV	#160000, RO	; FIRST LOC IN I/O PAGE
1168	025266	005710			1:	TST	(RO)	; FIND A
1169	025270	005737	027016			TST	NEXFLG	; NON-EXISTANT
1170	025274	001001				BNE	2:	; MEMORY ADDRESS
1171	025276	000773				BR	1:	
1172	025300	012737	000006	000004	2:	MOV	#6, @4	
1173	025306	010077	001436			MOV	RO, @BAR	
1174	025312	012777	000001	001426		MOV	#1, @BCR	; PREPARE BCR
1175	025320	052777	000101	001416		BIS	#101, @CSR	; ENABLE INTERRUPT AND FUNCTION (DMA)
1176	025326	112777	000252	001402		MOVB	#252, @IOR	; DUMMY VALUE TO OUTPUT REGISTER
1177	025334					DATAACC		
1178	025352	032777	000100	001364	3:	BIT	#BIT6, @CSR	; HAS INTERRUPT OCCURRED ?
1179	025360	001374				BNE	3:	; IF YES, CONTINUE
1180	025362	052777	000400	001354		BIS	#BIT8, @CSR	; REQUEST SERVICE
1181	025370	104000				SCOPE		
1182	025372	005037	027014		POLL4:	CLR	DONE	; CLEAR DONE FLAG
1183	025376	052777	000002	001330		BIS	#BIT1, @SMR	; GOTO CACS
1184	025404	112777	000137	001324		MOVB	#137, @IOR	; UNTALK IEC11-A
1185	025412					DATAACC		
1186	025430	112777	000030	001300		MOVB	#30, @IOR	; SERIAL POLL ENABLE
1187	025436	013700	026770			MOV	LIIND, RO	; BUILD MLA A
1188	025442	063700	026762			ADD	IECAAD, RO	; "
1189	025446	110077	001264			MOVB	RO, @IOR	; SET A TO LADS
1190	025452					DATAACC		
1191	025470	012701	000000			MOV	#0, R1	; BUILD FIRST TALKER ADDRESS
1192	025474	013700	026766		P4:	MOV	TAIND, RO	
1193	025500	060100				ADD	R1, RO	; BUILD TALKER ADDRESS
1194	025502	110077	001230			MOVB	RO, @IOR	
1195	025506					DATAACC		
1196	025524	052777	000004	001202		BIS	#BIT2, @SMR	; GOTO STANDBY STATE, A TO LACS
1197	025532					STALL		
1198	025562	122777	000000	001150		CMPB	#0, @IORH8	; POLLING ADDRESS ?
1199	025570	001452				BEQ	5:	; IF NOT, TRY THE NEXT ONE
1200	025572	120137	026764			CMPB	R1, IECBAD	; LEGAL ADDRESS ?
1201	025576	001416				BEQ	3:	; IF YES, CONTINUE
1202	025600	032737	040000	007160		BIT	#BIT14, PSDSWR	; PRINTOUT SELECTED ?
1203	025606	001012				BNE	3:	

1261	026320					DATAACC			
1262	026336	052777	000004	000370		BIS	#BIT2,BSMR		GOTO STANDBY STATE,A TO LACS
1263	026344					STALL			
1264	026374	122777	000000	000336		CMPB	#0,BSMR		POLLING ADDRESS ?
1265	026402	001452				BEQ	51		IF NOT,TRY THE NEXT ONE
1266	026404	120137	026764			CMPB	R1,IECBAD		LEGAL ADDRESS ?
1267	026410	001416				BEQ	31		IF YES,CONTINUE
1268	026412	032737	040000	007160		BIT	#BIT14,PSDSWR		PRINTOUT SELECTED ?
1269	026420	001012				BNE	31		
1270	026422					WRITE	MES21		GIVE INFORMATION ABOUT
1271	026432					DUMP	OCT,R1		ILLEGAL IEC-BUS ADDRESS
1272	026442					ERROR	70.		
1273	026446	117702	000266		31:	MOVB	BSMR,R2		SAVE STATUS BYTE
1274	026452	042702	177400			BIC	#177400,R2		
1275	026456	022702	000104			CMP	#104,R2		STATUS BYTE OK ?
1276	026462	001416				BEQ	41		IF YES,CONTINUE
1277	026464					ADP	#104,R2		
1278	026514					ERROR	71.		
1279	026520	012737	177777	027014	41:	MOV	#177777,DONE		SET DONE FLAG
1280	026526	000411				BR	PEND5		GOTO FINISH THIS TESTPART
1281	026530	005201			51:	INC	R1		GET NEXT TALKER ADDRESS
1282	026532	022701	000037			CMP	#37,R1		ALL DONE ?
1283	026536	001405				BEQ	PEND5		
1284	026540	052777	000002	000166		BIS	#BIT1,BSMR		IF NOT,GOTO CACS
1285	026546	000137	026306			JMP	P5		AND SELECT THE NEXT ONE
1286	026552	005737	027014		PEND5:	TST	DONE		ONE POLLING ADDRESS FOUND AT LEAST ?
1287	026556	001002				BNE	71		IF YES,GOTO NEXT PART OF THE TEST
1288	026560					ERROR	72.		
1289	026564	052777	000002	000142	71:	BIS	#BIT1,BSMR		GOTO CACS
1290	026572	112777	000031	000136		MOVB	#31,BSMR		DISABLE SERIAL POLL
1291	026600	052777	000004	000126		BIS	#BIT2,BSMR		GOTO STANDBY STATE,A TO LACS
1292	026606	000240			DONS:	NOP			JUST ENTRY-POINT
1293	026610	005737	027024			TST	TSTREP		TEST LOOP SELECTED ?
1294	026614	001402				BEQ	81		IF NOT,LEAVE THE TEST
1295	026616	000137	022260			JMP	REP7		
1296	026622				81:	WRITE	MESEND		END OF TEST
1297	026632	000207				RETURN			

```

1299
1300
1301
1302
1303 026654 052737 177777 027022 .SBTTL INTERRUPT SERVICE ROUTINES
1304 026642 000002 INTTST: BIS #177777,INTFLG ; SET INTERRUPT FLAG
1305
1306
1307
1308 026644 000240 INTSRV: NOP ; JUST ENTRY POINT
1309 026646 000002 RTI ;
1310
1311
1312
1313 026650 PRIOR: PSET #0 ; CLR PSW
1314 026674 062706 000004 ADD #4,SP ; INCREASE SP BY 4
1315 026700 000113 JMP (R3) ; GOTO PRINTOUT
1316
1317
1318
1319 026702 032777 040000 000022 INTA: BIT #BIT14,@CIR ; CAUSED SRQ THE INTERRUPT ?
1320 026710 001003 BNE 18 ;
1321 026712 052777 000100 000012 BIS #100,@CIR ; REENABLE INTERRUPT
1322 026720 000002 18: RTI ;
1323
1324
1325
1326 026722 052737 177777 027016 NEXMEM: BIS #177777,NEXFLG ; SET FLAG FOR NEX INDICATION
1327 026730 000002 RTI ;
1328
1329
1330

```

IEC11-1 IESI MACRO M1200 30 MAR 84 16:11 PAGE 41
CONSTANTS, VARIABLES

SEQ 120

1332			.SBTTL	CONSTANTS, VARIABLES	
1333	026732	160010	CIR:	160010	; CIR ADDRESS DEFAULT VALUE
1334	026734	160012	SMR:	160012	; SMR "

E10

IEC11 B TEST MACRO M1200 30 MAR 84 16:11 PAGE 42
CONSTANTS, VARIABLES

SEQ 121

1336 026736 160014

IOR: 160014

; IOR "

1338	026740	160015	IORMB:	160015
1339	026742	160016	VSR:	160016
1340	026744	160020	CSR:	160020
1341	026746	160022	BCR:	160022
1342	026750	160024	BAR:	160024
1343	026752	160026	MCR:	160026
1344	026754	160027	MCRMB:	160027
1345	026756	000270	VECA:	270
1346	026760	000274	VECB:	274
1347	026762	000035	IECAAD:	35
1348	026764	000036	IECBAD:	36
1349	026766	000100	TAIND:	100
1350	026770	000040	LIIND:	40
1351			:	
1352			:	
1353	026772	000000	MEMSAV:	.WORD 0
1354	026774	000000	BUFAB:	.WORD 0
1355	026776	000000	BUFAE:	.WORD 0
1356	027000	000000	BUFFL:	.WORD 0
1357	027002	000000	BUFBB:	.WORD 0
1358	027004	000000	BUFBE:	.WORD 0
1359			:	
1360			:	
1361	027006	000000	CNT1:	.WORD 0
1362	027010	000000	CNT2:	.WORD 0
1363	027012	000000	CHAR:	.WORD 0
1364	027014	000000	DONE:	.WORD 0
1365	027016	000000	NEXFLG:	.WORD 0
1366	027020	000000	BCINP:	.WORD 0
1367	027022	000000	INTFLG:	.WORD 0
1368	027024	000000	TSTREP:	.WORD 0
1369	027026	000000	ASAVE:	.WORD 0
1370	027030	000000	BSAVE:	.WORD 0

```

; IOR HIGH-BYTE ADDRESS DEFAULT VALUE
; VSR      "
; CSR      "
; BCR      "
; BAR      "
; MCR      "
; MCR HIGH BYTE ADDRESS DEFAULT VALUE
; DEFAULT OF IEC11-A VECTOR ADDRESS
; DEFAULT OF IEC11 B VECTOR ADDRESS
; IEC11-A BUS ADDRESS DEFAULT VALUE
; IEC11 B BUS ADDRESS DEFAULT VALUE
; TALKER INDICATION
; LISTENER INDICATION

; MEMORY AREA FOR BUFFERS
; FIRST ADDRESS IN BUFFER A
; LAST ADDRESS IN BUFFER A
; LENGTH OF BUFFERS IN BYTES
; FIRST ADDRESS IN BUFFER B
; LAST ADDRESS IN BUFFER B

; DELAY OR COUNTER
; DELAY OR COUNTER
; CHARACTER FOR MCR
;
; NEX-PROCEDURE (TEST3) TIMEOUT FLAG
; REQUESTED BYTE COUNT FOR DMA
; INTERRUPT FLAG
; FLAG FOR LOOP ON SINGLE TESTS
;
;

```

1372					.SBTTL	MESSAGES
1373					.MLIST	BEX
1374 027032	133	111	105	MID:	.ASCII	/[IEC11 B TEST IN CONNECTION/
1375 027066	133	127	111		.ASCII	/[WITH AN IEC11 A INTERFACE/
1376 027120	133	103	132		.ASCII	/[CZIECAO/
1377 027130	133	103	123		.ASCII	/[CSS MUNICH MARCH 1979/
1378 027156	133	115	117		.ASCII	/[MODIFIED FOR SOC SUBMISSION MARCH 1984]/
1379 027226	133	133	133	MES1:	.ASCII	/[[[ENTER FIRST REGISTER ADDRESS OF IEC11 A/
1380 027300	133	050	104		.ASCII	/[(DEFAULT IS 160010)]/
1381 027325	133	105	116	MES2:	.ASCII	/[ENTER VECTOR ADDRESS OF IEC11 A/
1382 027365	133	050	104		.ASCII	/[(DEFAULT IS 270)]/
1383 027407	133	105	116	MES3:	.ASCII	/[ENTER FIRST REGISTER ADDRESS OF IEC11 B/
1384 027457	133	050	104		.ASCII	/[(DEFAULT IS 160020)]/
1385 027504	133	105	116	MES4:	.ASCII	/[ENTER VECTOR ADDRESS OF IEC11 B/
1386 027544	133	050	104		.ASCII	/[(DEFAULT IS 274)]/
1387 027566	133	105	116	MES5:	.ASCII	/[ENTER IEC-BUS ADDRESS OF IEC11 A/
1388 027627	133	050	104		.ASCII	/[(DEFAULT IS 35)]/
1389 027650	133	126	105	MES6:	.ASCII	/[VECTOR INPUT IS NOT EQUAL TO CONTENTS OF VSR]/
1390 027726	133	105	116	MES7:	.ASCII	/[ENTER IEC BUS ADDRESS OF IEC11-B/
1391 027767	133	050	104		.ASCII	/[(DEFAULT IS 36)]/
1392 030010	133	133	124	MES8:	.ASCII	/[[[TEST 1: REGISTER STATIC TEST]]/
1393 030047	133	133	124	MES9:	.ASCII	/[[[TEST 2: TALKER AND LISTENER FUNCTION TEST]]/
1394 030123	133	133	124	MES10:	.ASCII	/[[[TEST 3: GENERAL INTERRUPT AND DMA FUNCTION TEST]]/
1395 030205	133	133	124	MES11:	.ASCII	/[[[TEST 4: DMA-TRANSFER FROM B TO A (B IS TALKER)]]/
1396 030267	133	133	124	MES12:	.ASCII	/[[[TEST 5: DMA TRANSFER FROM A TO B (B IS LISTENER)]]/
1397 030353	133	133	102	MES13:	.ASCII	/[[[BUS REQUEST LEVEL IS 7 !]]/
1398 030406	133	133	102	MES14:	.ASCII	/[[[BUS REQUEST LEVEL IS 6 !]]/
1399 030441	133	133	102	MES15:	.ASCII	/[[[BUS REQUEST LEVEL IS 5 !]]/
1400 030474	133	133	102	MES16:	.ASCII	/[[[BUS REQUEST LEVEL IS 4 !]]/
1401 030527	133	133	105	MES17:	.ASCII	/[[[ENTER BYTE COUNT NUMBER : (MAXIMAL)]]/
1402 030575	133	102	131	MES18:	.ASCII	/[[[BYTE COUNT NUMBER :]]]/
1403 030622	133	133	124	MES19:	.ASCII	/[[[TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)]]/
1404 030712	133	133	124	MES20:	.ASCII	/[[[TEST 7: SERIAL POLL PROCEDURE TEST]]/
1405 030757	133	133	111	MES21:	.ASCII	/[[[ILLEGAL STATUS BYTE ON IEC BUS ADDRESS :]]]/
1406 031034	133	133	116	MESNEX:	.ASCII	/[[[NEXT TEST TO RUN ?]]/
1407 031061	133	133	105	MESEND:	.ASCII	/[[[END OF TEST]]/
1408					.EVEN	
1409					.LIST	BEX

1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467

031100
031110 017601 000012
031114 016602 000014
031120 104000
031122 010211
031124 011100
031126 005102
031130 040200
031132 005102
031134 020200
031136 001432
031140
031146 001010
031150
031160
031170
031216
031222 000736
031224 104000
031226 040211
031230 011100
031232 005102
031234 040200
031236 005102
031240 005700
031242 001433
031244
031252 001010
031254

REGTST: PUSH R0,R1,R2,R3
MOV B12(SP),R1
MOV 14(SP),R2
11: SCOPE
MOV R2,BR1
MOV BR1,R0
COM R2
BIC R2,R0
COM R2
CMP R2,R0
BEQ 21
TST14 PSDSWR
BNE S1
WRITE RGMS1
DUMP OCT,R1
51: ADP R2,R0
ERROR 255.
BR 11
21: SCOPE
BIC R2,BR1
MOV BR1,R0
COM R2
BIC R2,R0
COM R2
TST R0
BEQ 31
TST14 PSDSWR
BNE 61
WRITE RGMS1

.SBTTL STANDARD CSS REGISTER TEST

GROO1
.....

31 MAR 76

ROUTINE DESCRIPTION:

INPUT: REGISTER,BITMASK

OUTPUT: ONLY IF ERRORS OCCUR:

BIT MASK OF R AND W BITS IN REGISTER NR: xxxxxx

CONTENTS SHOULD BE: BUT WAS

ERROR 000255,000254 OR 000253

FUNCTION: BEFORE THIS ROUTINE CALL AT FIRST MUST BE

PUSHED THE BITMASK AND THEN THE REGISTER

ONTO THE STACK.

THE ROUTINE CHECKS ALL READ/WRITE BITS OF

THE CORRESPONDING REGISTER AND PRINTS ERROR

MESSAGES IF THE READ AND WRITE WAS NOT IDENT.

EXAMPLE: PUSH #128,0CSR

CALL REGTST

; GROO1
; GET ADDRESS OF REGISTER;GROO1
; GET BITMASK ;GROO1
; PREPARE SCOPE LOOP
; TRY TO SET ALL BITS
; TRY TO READ IT BACK
; BUILD MASK COMPLEMENT
; CLEAR UNUSED BITS
; REBUILD MASK
; ALL BITS SET ?
; BRANCH IF YES
; IF ADP AFTER ERROR IS NOT ALLOWED.
; GOTO S1,ELSE WRITE
; WRITE AT REGISTER NR:
; DMP REGISTER NUMBER

; NOT ALL BITS SET
; RESTART TEST
; SAVE SCOPE LOOP ADDRESS
; CLEAR BITS UNDER MASK
; READ BITS
; BUILD MASK COMPLEMENT
; MASK OUT UNUSED BITS
; REBUILD MASK
; ARE ALL BITS RESET
; BRANCH IF YES
; IF ADP IS NOT ALLOWED
; GOTO 61
; AT REGISTER

```

1468 031264      DUMP      OCT,R1      ; DMP REGISTER NR
1469 031274      ADP       63,R0
1470 031324      ERROR     254.        ; NOT ALL BITS RESET
1471 031330      BR        1#          ; RESTART TEST
1472 031332      005102      3# :      CGM      R2          ; BUILD COMPLEMENT OF MASK
1473 031334      004737      031506    CALL     RANDOM      ; GET A RANDOM #
1474 031340      040200      BIC      R2,R0      ; MASK OUT UNUSED BITS
1475 031342      104000      SCOPE      ; SAVE SCOPE LOOP ADDRESS
1476 031344      010011      MOV      R0,(R1)    ; TRY TO SET RANDOM BITS
1477 031346      011103      MOV      @R1,R3    ; READ IT BACK ; G001
1478 031350      040203      BIC      R2,R3    ; CLEAR UNUSED BITS ; G001
1479 031352      020300      CMP      R3,R0    ; ALL BITS OK ? ; G001
1480 031354      001432      BEQ      4#          ; BRANCH IF YES
1481 031356      TST14      PSDSWR      ; IF ADP IS NOT ALLOWED
1482 031364      001023      BNE      7#          ; GOTO 7#
1483 031366      WRITE      RG#MS#      ; ELSE WRITE REGISTER NR
1484 031376      DUMP      OCT,R1      ; DUMP REGISTER NUMBER
1485 031406      ADP       R0,R3      ; SHOULD BE R0,WAS R3
1486 031434      7# :      ERROR     253.        ; DATA CHECK IN REGISTER
1487 031440      000627      BR        1#          ; RESTART TEST
1488 031442      005102      4# :      CGM      R2          ; REBUILD MASK
1489 031444      $$$LOOP    1000,1#
1490 031466      POP      R3,R2,R1,R0      ; G001
1491 031476      062706      000006      ADD      @6,SP      ; REBUILD STACK
1492 031502      000176      177772      JMP      @ 6(SP)    ; RETURN TO CALLER
1493
1494
1495
1496 031506      013746      031544      ;
1497 031512      013700      031546      ;
1498 031516      006316      ;
1499 031520      005500      ;
1500 031522      006200      ;
1501 031524      005516      ;
1502 031526      061600      ;
1503 031530      005600      ;
1504 031532      012637      031544      ;
1505 031536      010037      031546      ;
1506 031542      000207      ;
1507 031544      135753      ;
1508 031546      024624      ;
1509
1510 031550      133      102      111  RG#MS# : .NLIST      BEX
1511
1512
1513 031624      PRGEND:      .ASCII  /[BIT MASK OF R AND W BITS IN REGISTER NR: ]/
1514      001204      .LIST      BEX
                        .EVEN
                        .END      START

```

ACC4	020040	D..TYP	006420	MERROR	007652	NOR.ER	004376	SAV	011312
ADDRIN	011316	EMTIBL	002416	MERRP2	007662	NOR..E	004372	SCOPAD	007172
ASAVE	027026	EMTIND	002424	MER1	010304	NO6	010612	SCOPE	104000
BAR	026750	EMTO	002424	MER2	010333	OFFCNT	012240	SEQ	012160
BCINP	027020	EMT1	002432	MESEND	031061	OFFSET	012112	SEQPNT	012156
BCIN4	017376	EMT2	002470	MESNEX	031034	OUTFLG	007152	SINGLE	012114
BCIN5	020470	ENDBIT	016410	MES1	027226	PEND1	023262	SISDR0	172200
BCOVFL	016216	ERRFLG	007154	MES10	030123	PEND2	024150	SMR	026734
BCR	026746	IAD..E	004302	MES11	030205	PEND3	025026	SP8TOA	026150
BDDRIN	011724	IAE..E	005016	MES12	030267	PEND4	025740	SPEFLG	007170
BEG6	021576	IAS..E	004466	MES13	030353	PEND5	026552	SPEND	024356
BIT0	000001	IAY..E	004116	MES14	030406	PMODE	030000	SPNEX	025234
BIT1	000002	IECAAD	026762	MES15	030441	POLL0	022340	SPOVFL	023506
BIT10	002000	IECBAD	026764	MES16	030474	POLL1	022736	SRO	177572
BIT11	004000	INFLAG	007150	MES17	030527	POLL2	023602	SR3	172516
BIT12	010000	INPREQ	007156	MES18	030575	POLL3	024460	START	001204
BIT13	020000	INP..E	003776	MES19	030622	POLL4	025372	STATUS	007164
BIT14	040000	INTA	026702	MES2	027325	POLL5	026262	STCHGE	014564
BIT15	100000	INTB4	020100	MES20	030712	PR	016622	TAIND	026766
BIT2	000004	INTFLG	027022	MES21	030757	PRERR	017324	TESTAL	012144
BIT3	000010	INTSRV	026644	MES3	027407	PRGENO	031624	TEST1	012242
BIT4	000020	INTTST	026634	MES4	027504	PRIEX	017330	TEST2	013074
BIT5	000040	IOR	026736	MES5	027566	PRIOR	026650	TEST3	014352
BIT6	000100	IORHB	026740	MES6	027650	PRI4	017310	TEST4	017350
BIT7	000200	I..DMP	000001	MES7	027726	PRI5	017236	TEST5	020442
BIT8	000400	KBBEND	007146	MES8	030010	PRI6	017164	TEST6	021350
BIT9	001000	KBBPNT	007146	MES9	030047	PRI7	017112	TEST7	022206
BSAVE	027030	KBBUFF	007026	MID	027032	PROMES	007202	TKB	177562
BUFAB	026774	KDSAR0	172360	MILADR	007445	PR1	017050	TKS	177560
BUFAE	026776	KDSDR0	172320	MILBIN	007360	PR2	017126	TPB	177566
BUFBB	027002	KISAR0	172340	MILDEC	007316	PR3	017200	TPS	177564
BUFBE	027004	KISAR5	172352	MILEMT	007421	PR4	017252	TSTREP	027024
BUFFL	027000	KISAR6	172354	MILOCT	007256	PS	177776	TT	012220
BUFPRP	010614	KISAR7	172356	MILSCP	007524	PSDSWR	007160	TTE	012242
CDM..E	010614	KISDR0	172300	MILTR1	010104	P0	022412	TYP..E	001252
CHAR	027012	KISDR6	172314	MILTR2	010144	P1	023016	UBMPR	170200
CIR	026732	KISDR7	172316	MKBOVF	007213	P2	023704	UDSAR0	177660
CMODE	140000	LAB1	013340	MNOSUB	010032	P3	024562	UDSDR0	177620
CMP4	020276	LAB11	012042	MPWRFL	007704	P4	025474	UISAR0	177640
CMP5	021204	LAB2	013416	MSYERR	007503	P5	026306	UISAR4	177650
CMP6	022010	LAB3	013474	MVERSN	010156	RA	031544	UISAR5	177652
CNT1	027006	LAB4	013612	M..DET	001662	RANDOM	031506	UISAR6	177654
CNT2	027010	LAB5	013760	M..HIM	001766	RB	031546	UISAR7	177656
CSR	026744	LAB6	014036	M..MMM	002006	RED..E	005626	UISDR0	177600
DEFA	011374	LAB7	014114	M..RET	001726	REGTST	031100	UISDR4	177610
DEFB	012002	LAB8	014176	M..SAV	001424	REP1	012252	UISDR5	177612
DMA4	020050	LAB9	014326	M..TRP	001776	REP2	013106	UISDR6	177614
DMA6	021676	LAST4	020106	M..TSE	001374	REP3	014372	UISDR7	177616
DMP..E	006674	LIIND	026770	M..TST	001516	REP4	017450	U..DET	001760
DONE	027014	MAPER	010472	N	000005	REP5	020542	VECA	026756
DONO	022610	MCR	026752	NEXBIT	014712	REP6	021564	VECAIN	011216
DON1	023334	MCRHB	026754	NEXFLG	027016	REP7	022260	VECB	026760
DON2	024204	MEMDEF	010426	NEXMEM	026722	RGAIN	010752	VECBIN	011646
DON3	025062	MEMEND	007022	NEX1	015214	RGBIN	011402	VSR	026742
DON4	025774	MEMSAV	026772	NEX2	015424	RGMS	031550	WRT..E	006060
DON5	026606	MERCOV	007737	NEX3	015626	RUBFLG	007162	X	000340
DUMMES	007210	MERHLT	010002	NEX4	016022	R..TYP	000001	XX	000004

Z\$\$Z	=	000004	\$.ASF	007174	\$.DOC	006426	\$.IOT	003056	\$.RED	005020
Z\$\$\$Z	=	000000	\$.ADP	006706	\$.EMT	002326	\$.KBI	003424	\$.RRS	003400
\$MNOAV		010351	\$.ASC	005464	\$.ER	007166	\$.KBO	006122	\$.RSV	003360
\$POINT		006614	\$.BIN	005504	\$.IAA	004120	\$.MAP	002116	\$.SPC	012010
\$SHLD		007176	\$.BUF	005444	\$.IAD	004210	\$.MSP	005700	\$.STX	006262
\$TABLE		006616	\$.DBN	006630	\$.IAE	004472	\$.NOR	004304	\$.TOT	006754
\$WAS		007200	\$.DDC	006520	\$.IAS	004402	\$.OCT	005112	\$.TRP	002524
\$\$MTYP		007020	\$.DEC	005270	\$.IAY	004010	\$.PRO	006136	\$.TYP	001242
\$\$PTYP		007016	\$.DMP	006400	\$.INP	003574	\$.PWR	003126	\$.WRT	005654
\$\$SPE		003520								

. ABS. 031624 000
000000 001
CSSMON 000000 002
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 13624 WORDS (54 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:03:37
ZIECAO,ZIECAO/-SP=NEWMAC/ML.COM,ZIECAO