IDENTIFICATION

PRODUCT CODE: AC T879A-MC
PRODUCT TITLE: CZIECA0 IEC11-B PDP-11 DIAGNOSTIC
PRODUCT DATE: JANUARY 1984
DEPARTMENT: CSS, MUNICH
AUTHOR: J. HOLLER

TABLE OF CONTENTS

1.0 ABSTRACT

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

2.0 REQUIREMENTS

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 PRELIMINARY DIAGNOSTICS

ALL DIAGNOSTIC PROGRAMS FOR THE IEC11-A MUST RUN SUCCESSFUL.

3.0 LOADING PROCEDURE

3.1 METHOD

SEE DIAGNOSTIC MONITOR DESCRIPTION

4.0 STARTING PROCEDURE

SEE DIAGNOSTIC MONITOR DESCRIPTION

5.0 OPERATING PROCEDURES

5.1 SWITCH SETTINGS

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 PROCEDURE TEST
END OF TEST
TEST 1: REGISTER STATIC TEST
<CTRL C>
COM>ABO<CR>

NEXT TEST TO RUN ?
COM>
;
;
;       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 "REGTST" 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 TYP 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 P R O G R A M D E S C R I P T I O N
****

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
SUCCESSFULL !!

THE FOLLOWING SECTION CONTAINS THE FLOWCHARTS FOR ALL TESTS.

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

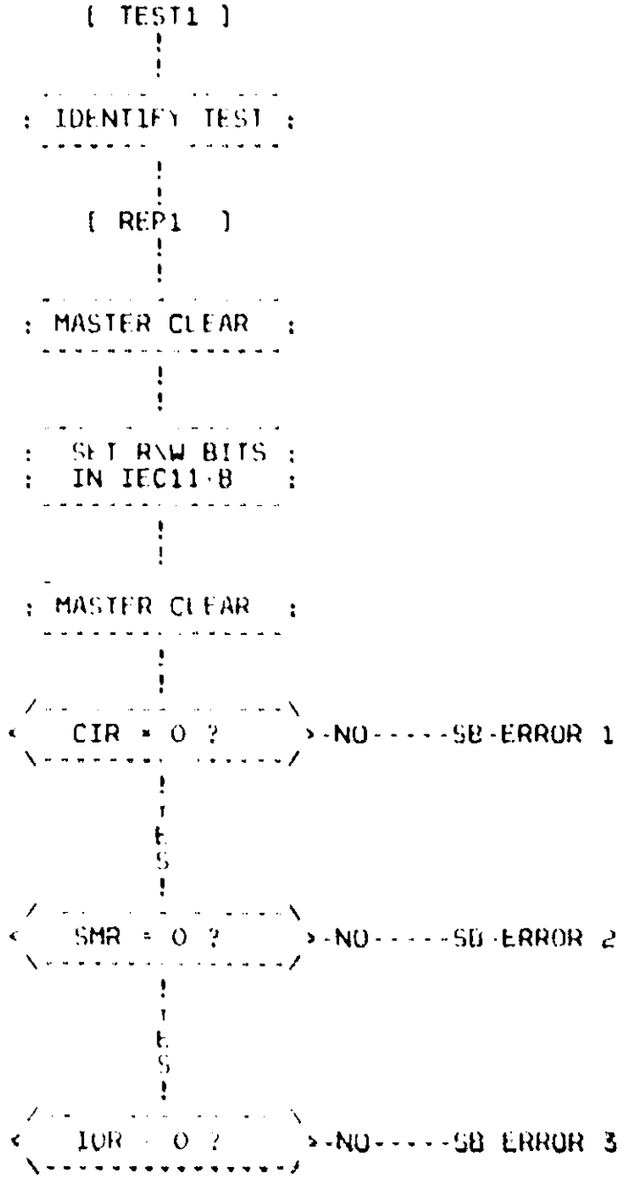
SB-ERROR 35 : SB STANDS FOR AN ADDITIONAL PRINTOUT BEFORE
ERROR NUMBER PRINT OF THE FORM :

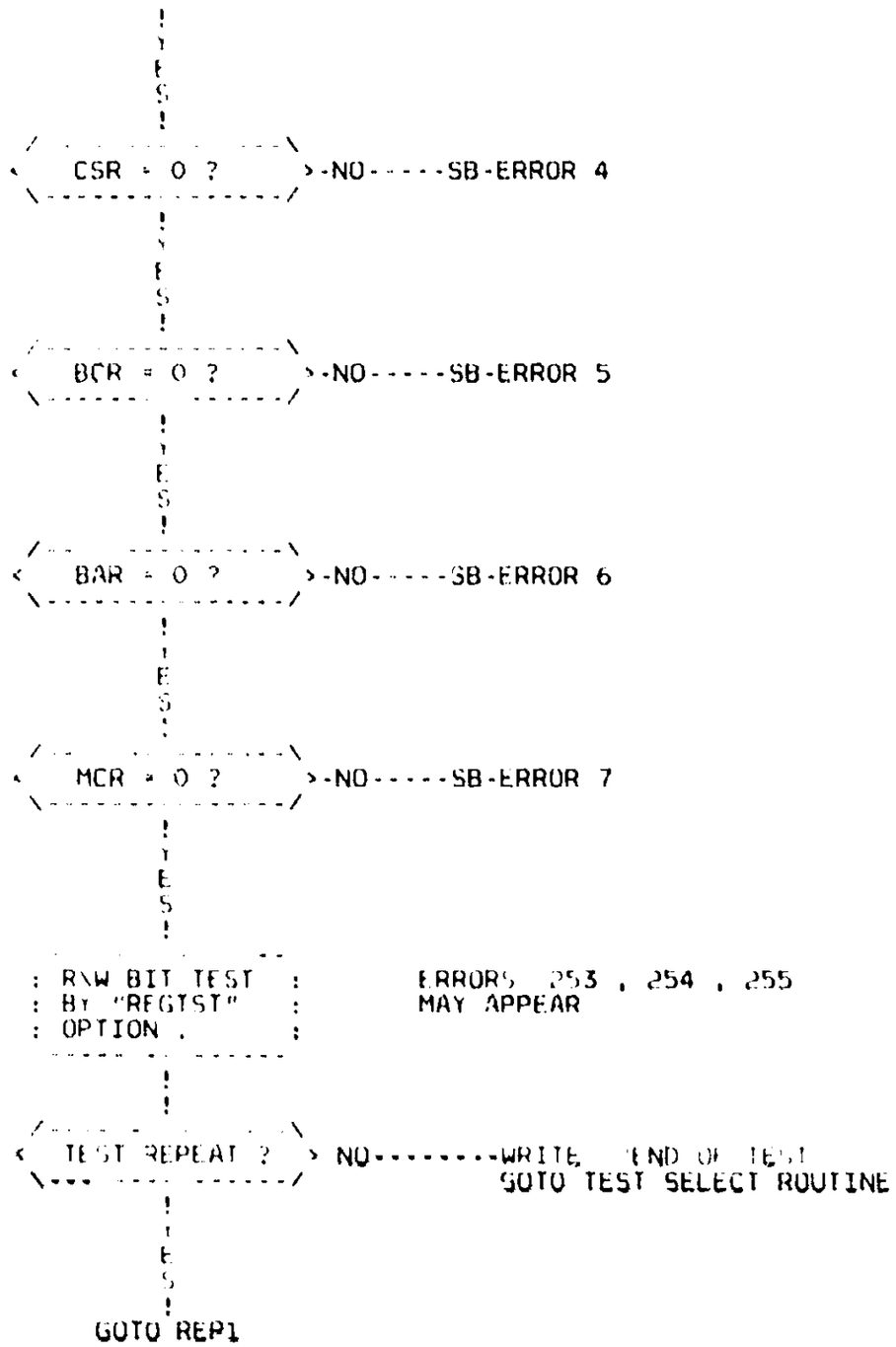
CONTENTS SHOULD BE : 000026 BUT WAS : 000025

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

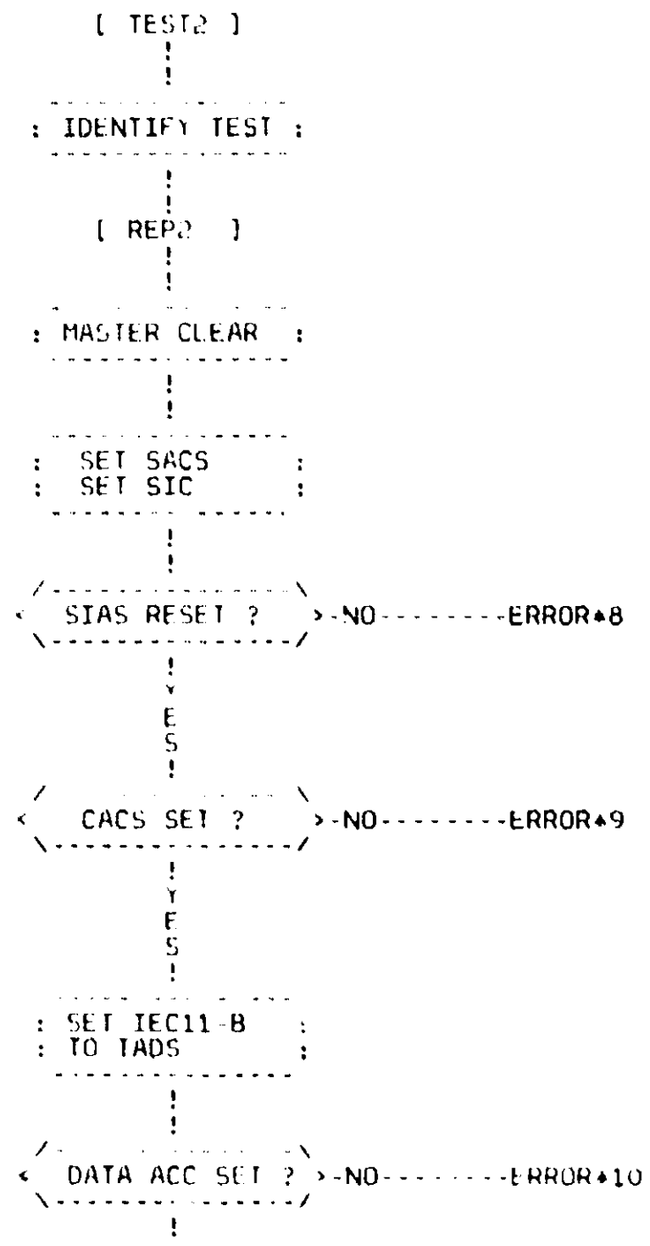
********** TEST 1: REGISTER STATIC TEST

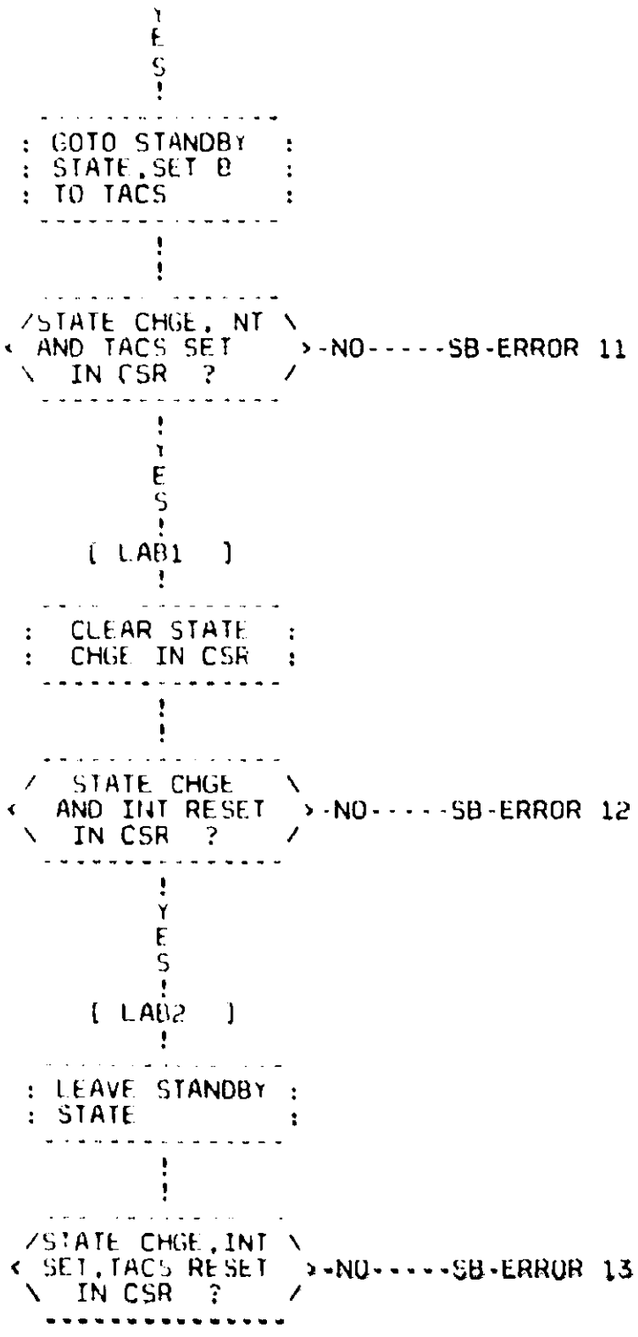
THE MASTER CLEAR FUNCTION OF OF IEC11-A AND IEC11B
IS TESTED.
ALL READ/WRITE BITS IN IEC11 B ARE CHECKED WITH
ALTERNATE PATTERN (EXCEPT BIT 0 IN CSR.)

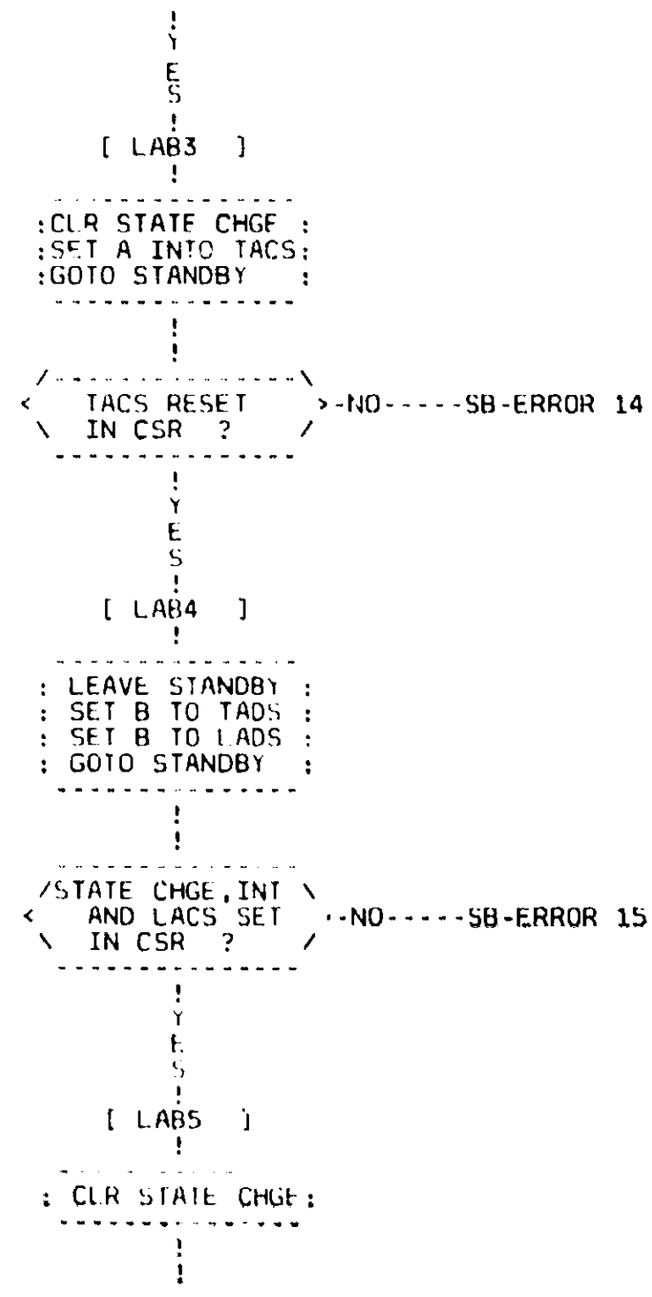


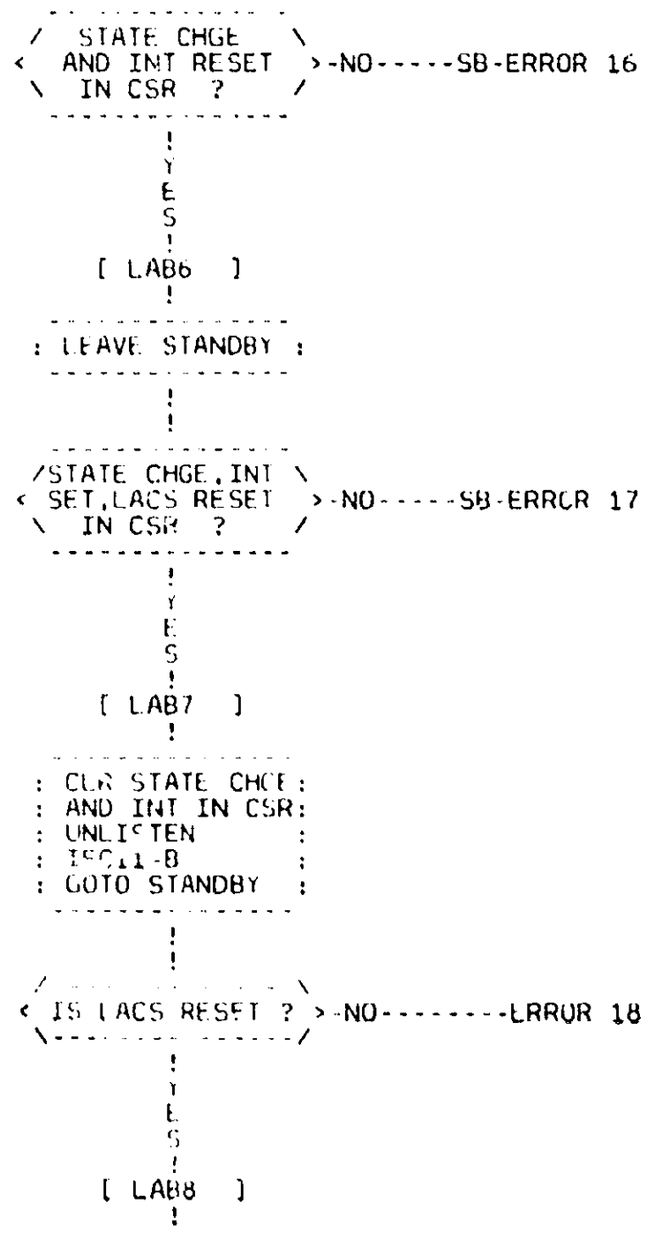

CSR = 0 ?
NO
SB-ERROR 4
YES
BCR = 0 ?
NO
SB-ERROR 5
YES
BAR = 0 ?
NO
SB-ERROR 6
YES
MCR = 0 ?
NO
SB-ERROR 7
YES
R\W BIT TEST BY "REGTST" OPTION .
ERRORS 253 , 254 , 255 MAY APPEAR
TEST REPEAT ?
NO
WRITE "END OF TEST"
GOTO TEST SELECT ROUTINE
YES
GOTO REP1

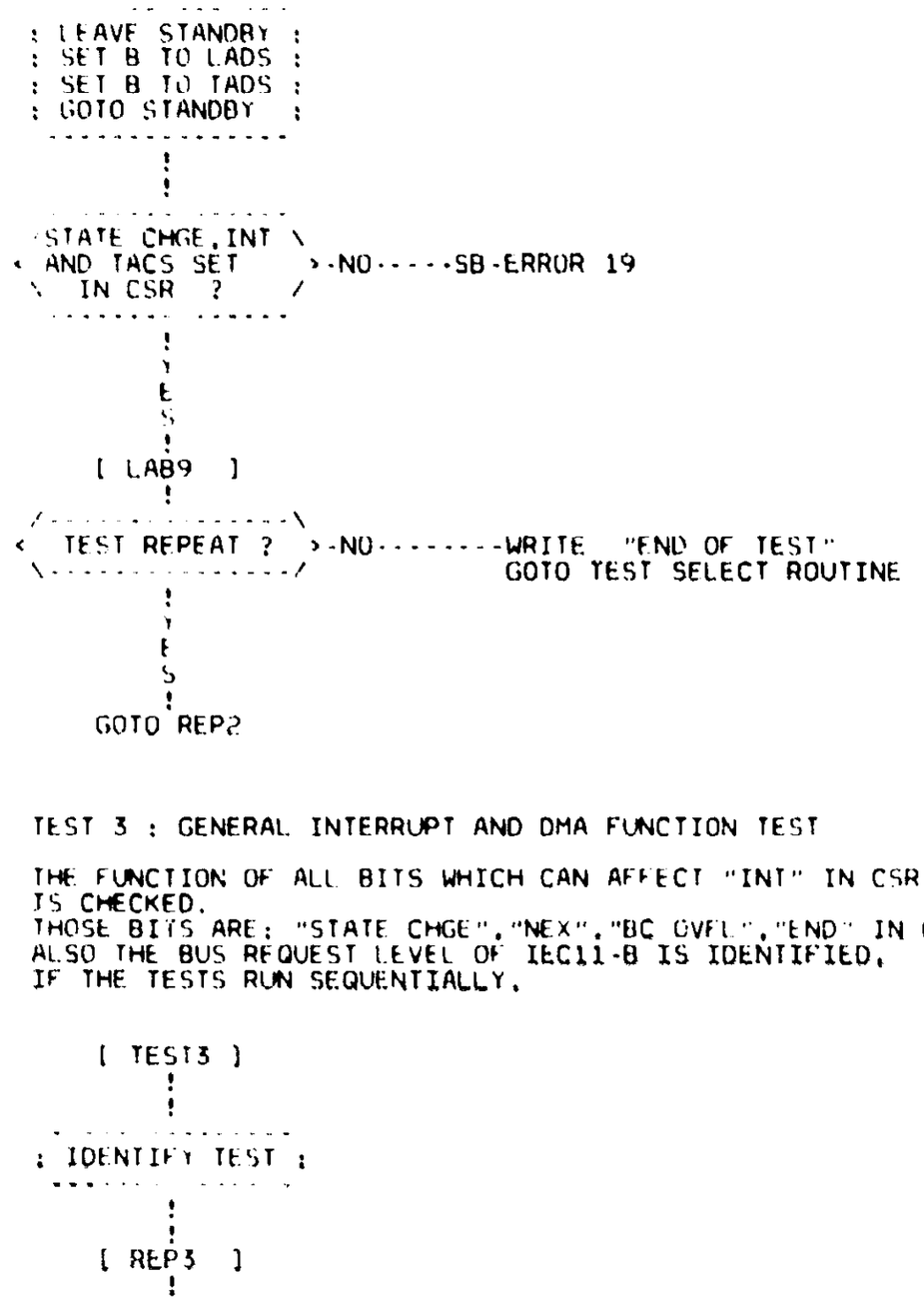
********** TEST 2 : TALKER AND LISTENER FUNCTION TEST




Y
E
S
GOTO STANDBY STATE,SET B TO TACS
STATE CHGE, NT AND TACS SET IN CSR ?
NO
SB-ERROR 11
Y
E
S
[LAB1]
CLEAR STATE CHGE IN CSR
STATE CHGE AND INT RESET IN CSR ?
NO
SB-ERROR 12
Y
E
S
[LAB2]
LEAVE STANDBY STATE
STATE CHGE,INT SET,TACS RESET IN CSR ?
NO
SB-ERROR 13


YES
[LAB3]
CLR STATE CHGF
SET A INTO TACS
GOTO STANDBY
TACS RESET IN CSR ?
NO
SB-ERROR 14
YES
[LAB4]
LEAVE STANDBY
SET B TO TADS
SET B TO LADS
GOTO STANDBY
STATE CHGE,INT AND LACS SET IN CSR ?
NO
SB-ERROR 15
YES
[LAB5]
CLR STATE CHGE

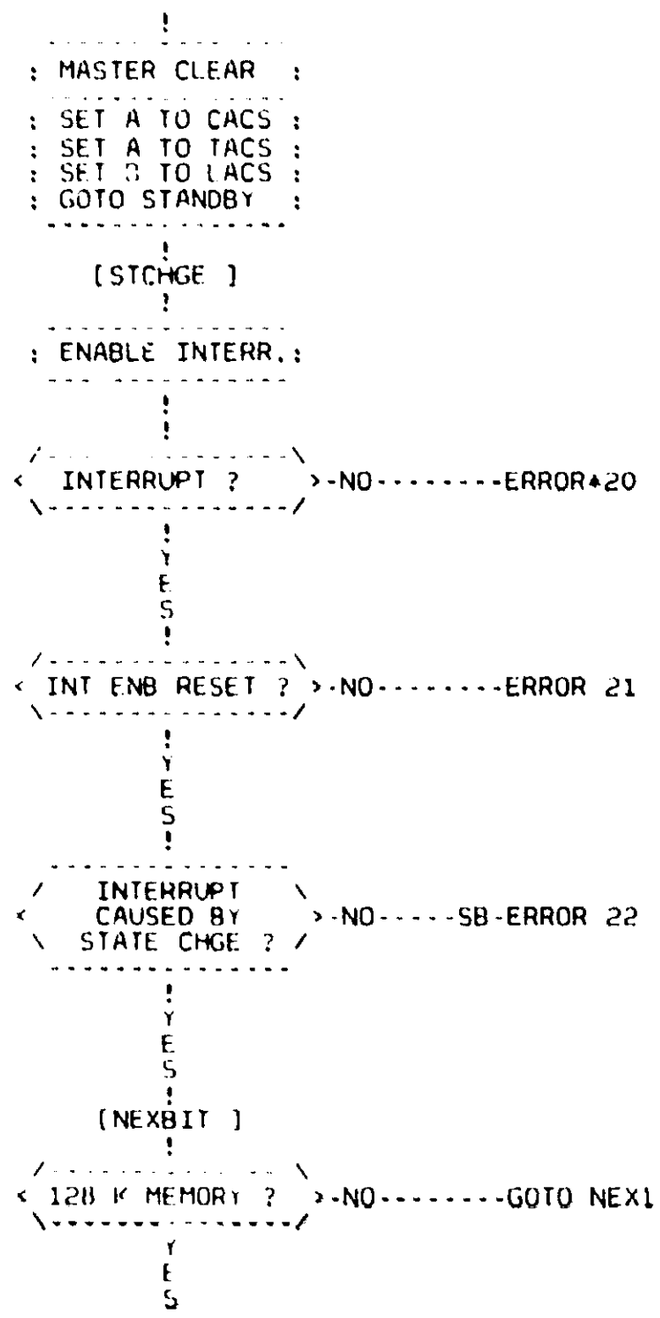

STATE CHGE AND INT RESET IN CSR ?
NO
SB-ERROR 16
YES
LAB6
LEAVE STANDBY
STATE CHGE,INT SET,LACS RESET IN CSR ?
NO
SB-ERROR 17
YES
LAB7
CLR STATE CHGE AND INT IN CSR
UNLISTEN
GOTO STANDBY
IS LACS RESET ?
NO
ERROR 18
YES
LAB8

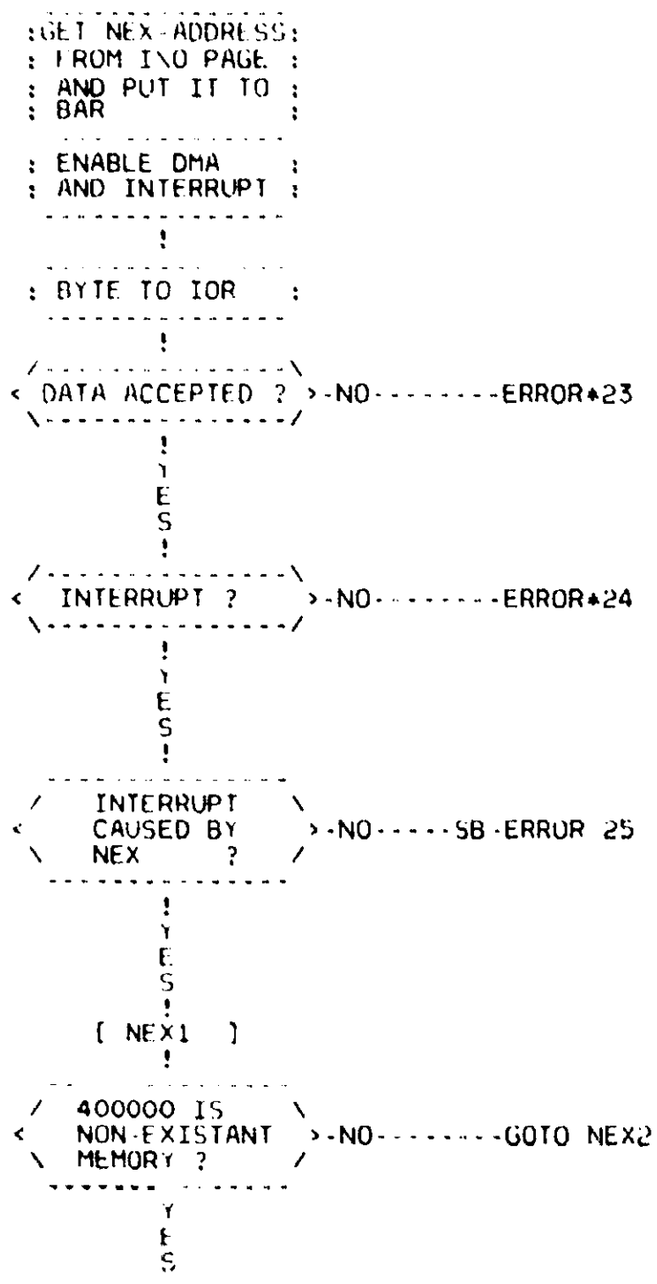


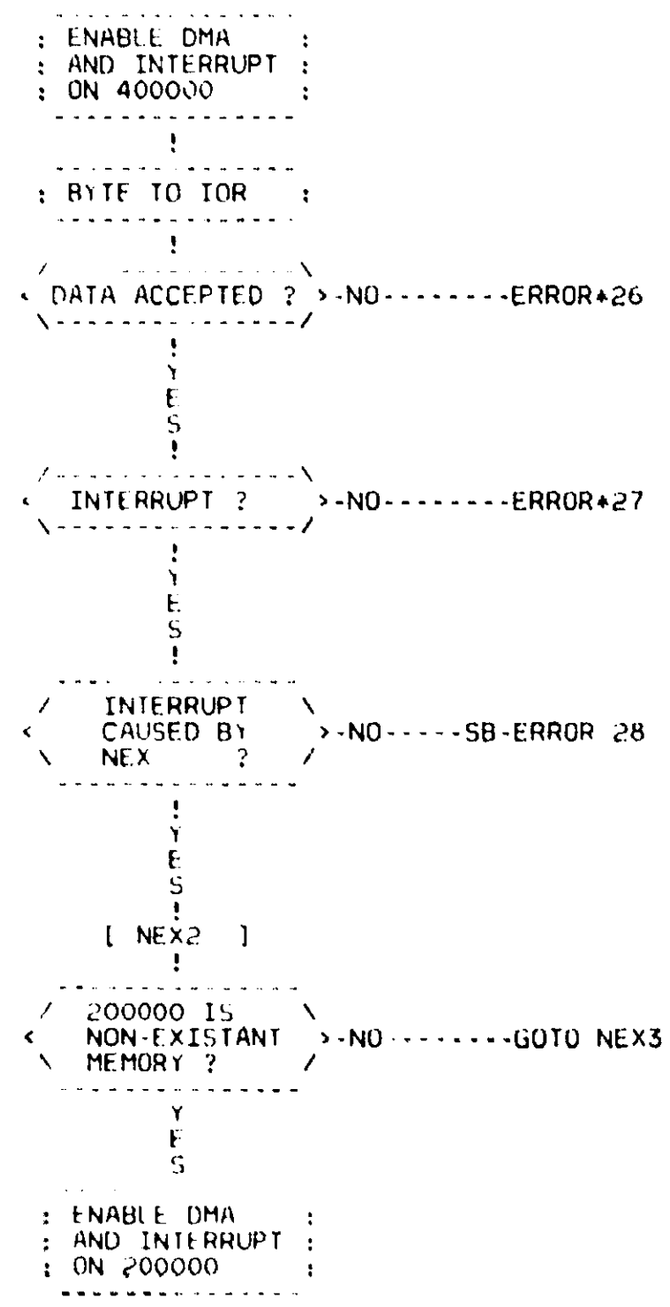
****

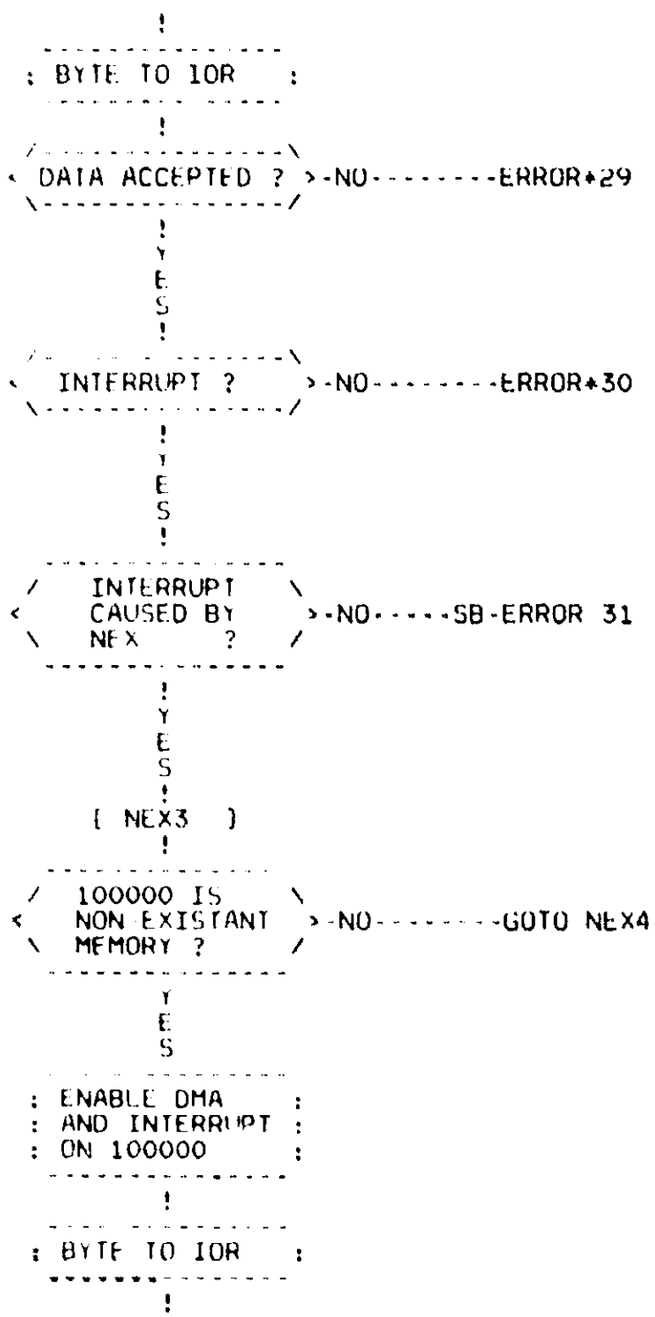
TEST 3 : GENERAL INTERRUPT AND DMA FUNCTION TEST

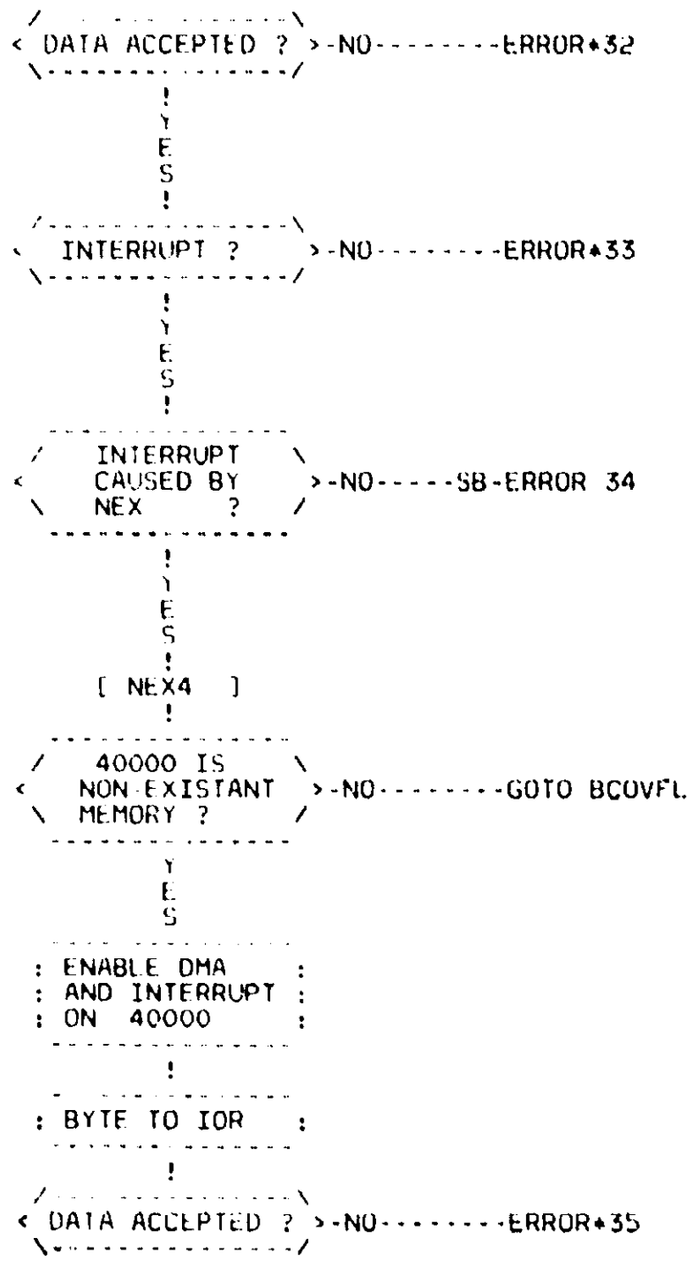
THE FUNCTION OF ALL BITS WHICH CAN AFFECT "INT" IN CSR
IS CHECKED.
THOSE BITS ARE: "STATE CHGE","NEX","BC OVFL","END" IN CSR
ALSO THE BUS REQUEST LEVEL OF IEC11-B IS IDENTIFIED,
IF THE TESTS RUN SEQUENTIALLY.

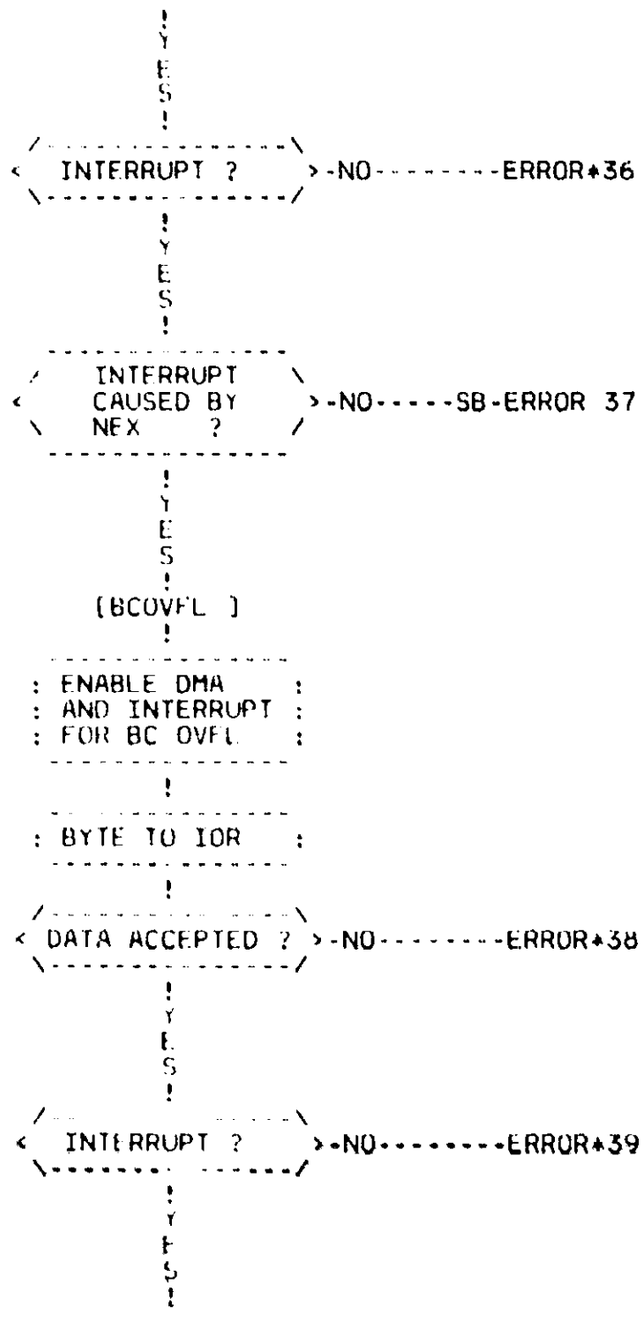

MASTER CLEAR
SET A TO CACS
SET A TO TACS
SET B TO LACS
GOTO STANDBY
[STCHGE]
ENABLE INTERR.
INTERRUPT ?
NO
ERROR*20
YES
INT ENB RESET ?
NO
ERROR 21
YES
INTERRUPT CAUSED BY STATE CHGE ?
NO
SB-ERROR 22
YES
[NEXBIT]
128 K MEMORY ?
NO
GOTO NEXT
YES

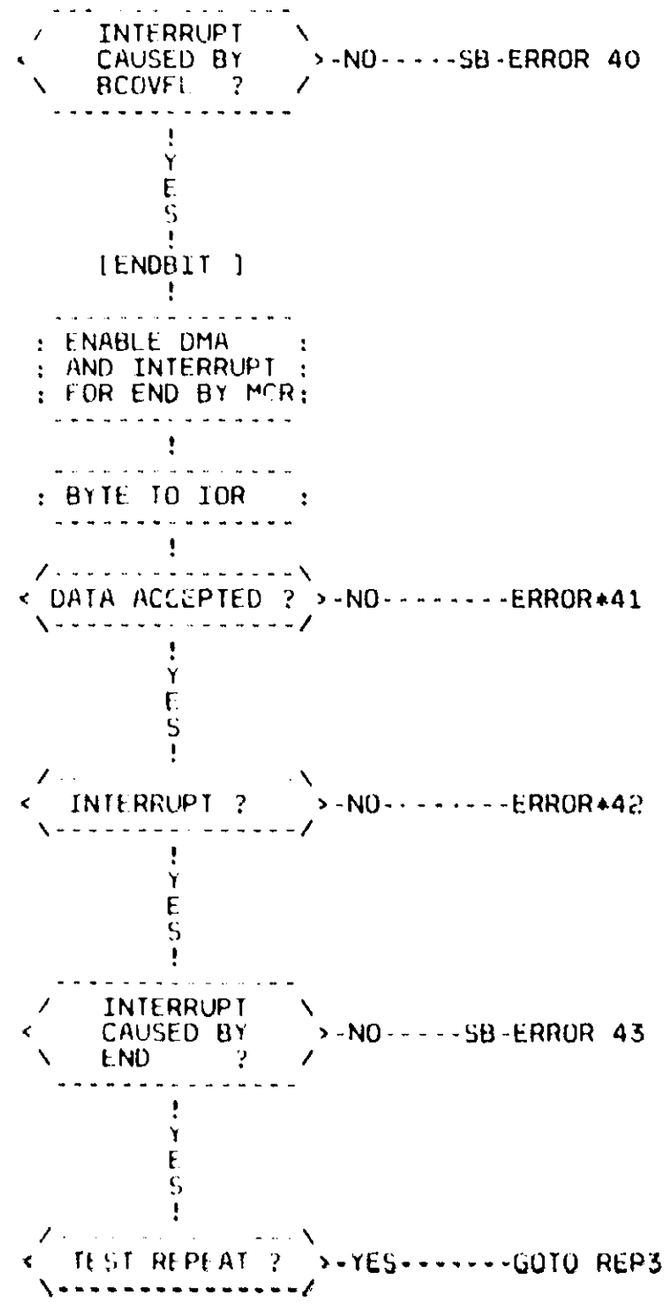

GET NEX-ADDRESS FROM I\O PAGE AND PUT IT TO BAR
ENABLE DMA AND INTERRUPT
BYTE TO IOR
DATA ACCEPTED ?
NO
ERROR*23
YES
INTERRUPT ?
NO
ERROR*24
YES
INTERRUPT CAUSED BY NEX ?
NO
SB-ERROR 25
YES
[NEX1]
400000 IS NON-EXISTANT MEMORY ?
NO
GOTO NEX2
YES

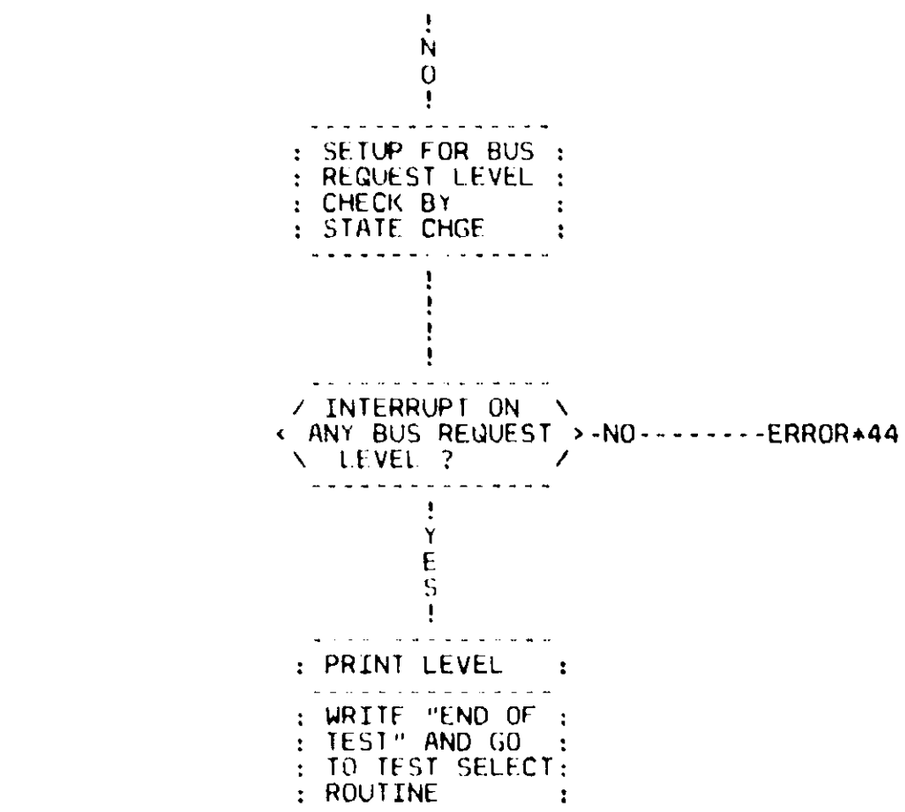

ENABLE DMA AND INTERRUPT ON 400000
BYTE TO IOR
DATA ACCEPTED ?
NO
ERROR*26
YES
INTERRUPT ?
NO
ERROR*27
YES
INTERRUPT CAUSED BY NEX ?
NO
SB-ERROR 28
YES
[NEX2]
200000 IS NON-EXISTANT MEMORY ?
NO
GOTO NEX3
YES
ENABLE DMA AND INTERRUPT ON 200000


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

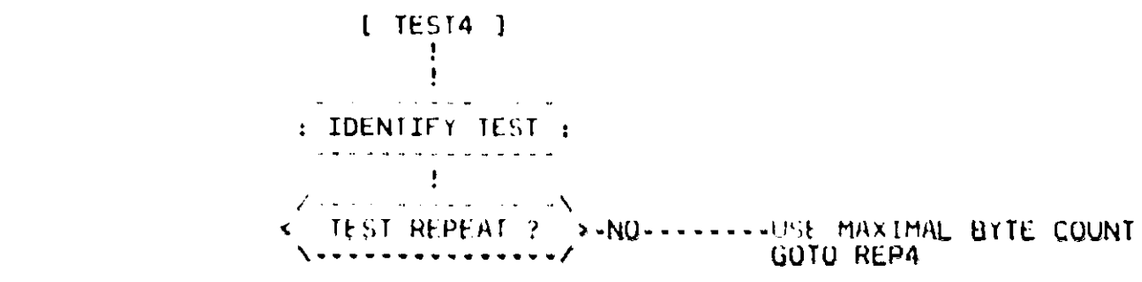

DATA ACCEPTED ?
NO
ERROR*32
YES
INTERRUPT ?
NO
ERROR*33
YES
INTERRUPT CAUSED BY NEX ?
NO
SB-ERROR 34
YES
[NEX4]
40000 IS NON-EXISTANT MEMORY ?
NO
GOTO BCOVFL
YES
ENABLE DMA AND INTERRUPT ON 40000
BYTE TO IOR
DATA ACCEPTED ?
NO
ERROR*35

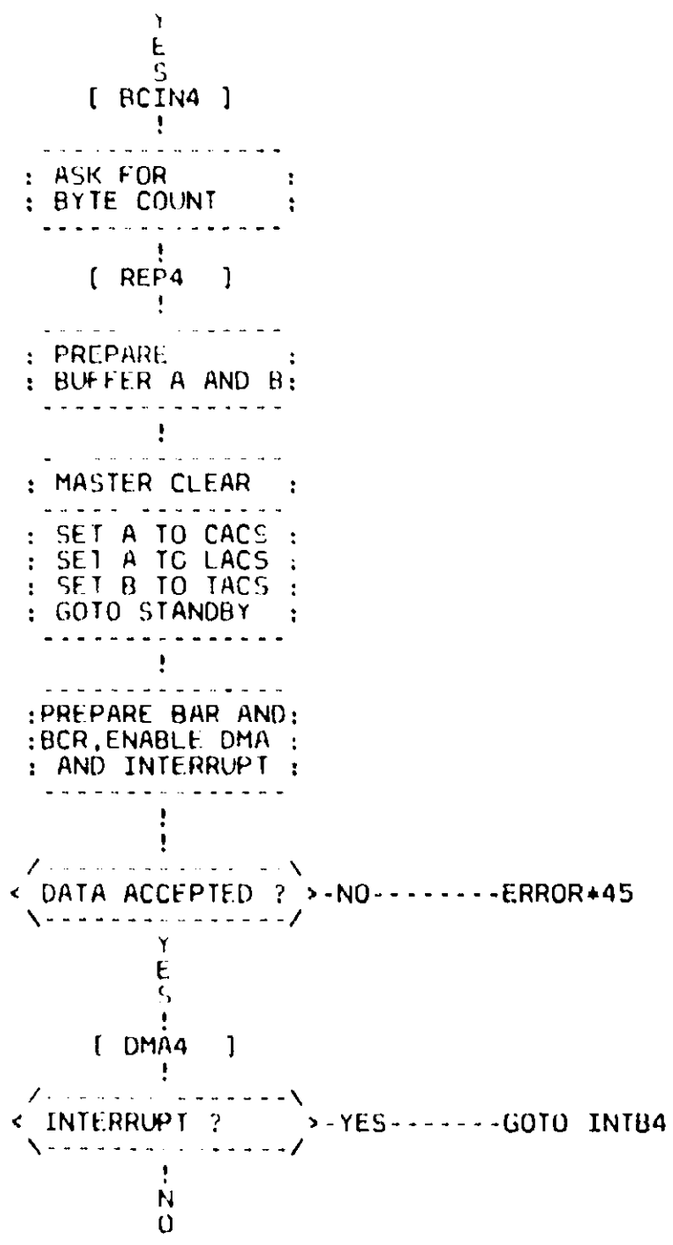

YES
INTERRUPT ?
NO
ERROR#36
YES
INTERRUPT CAUSED BY NEX ?
NO
SB-ERROR 37
YES
[BCOVFL]
ENABLE DMA AND INTERRUPT FOR BC OVFL
BYTE TO IOR
DATA ACCEPTED ?
NO
ERROR#38
YES
INTERRUPT ?
NO
ERROR#39
YES

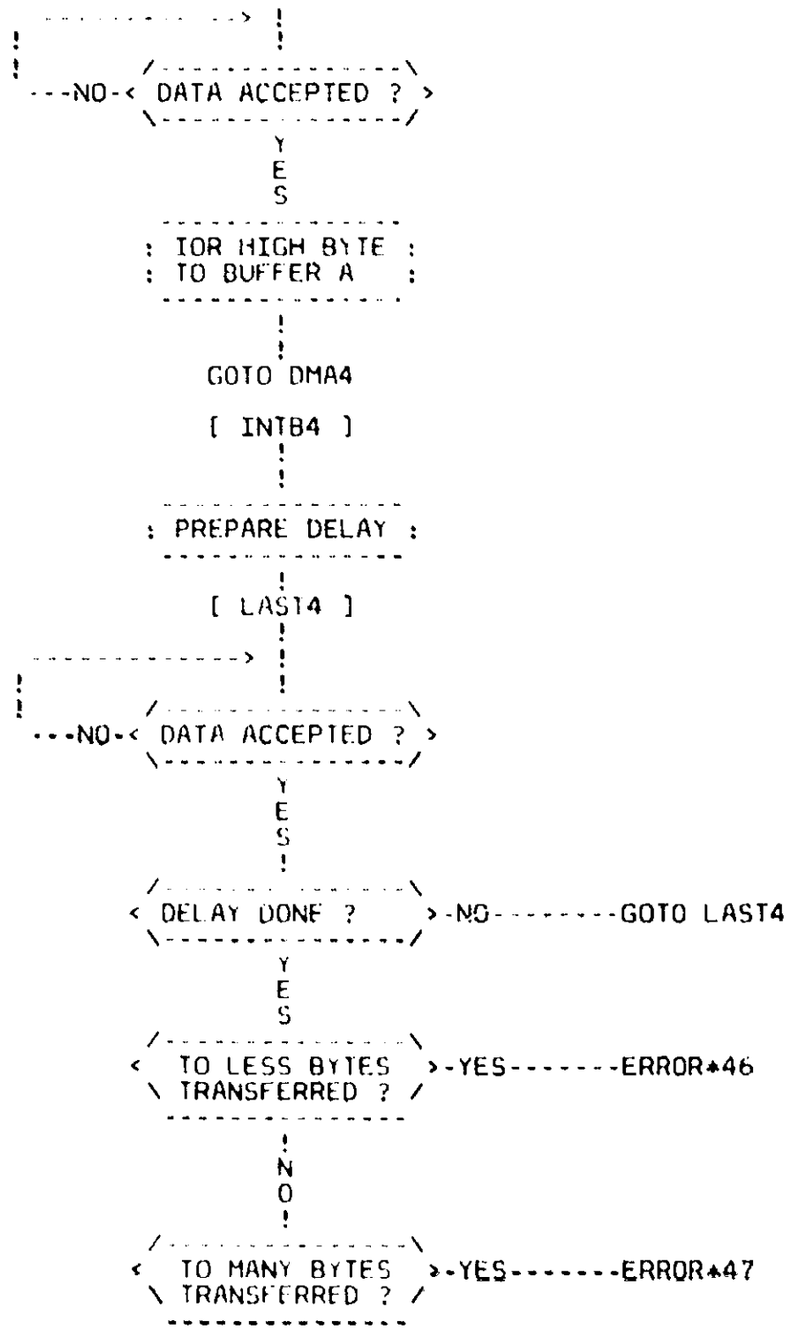

INTERRUPT CAUSED BY BCOVFL ?
NO
SB-ERROR 40
YES
[ENDBIT]
ENABLE DMA AND INTERRUPT FOR END BY MCR
BYTE TO IOR
DATA ACCEPTED ?
NO
ERROR*41
YES
INTERRUPT ?
NO
ERROR*42
YES
INTERRUPT CAUSED BY END ?
NO
SB-ERROR 43
YES
TEST REPEAT ?
YES
GOTO REP3

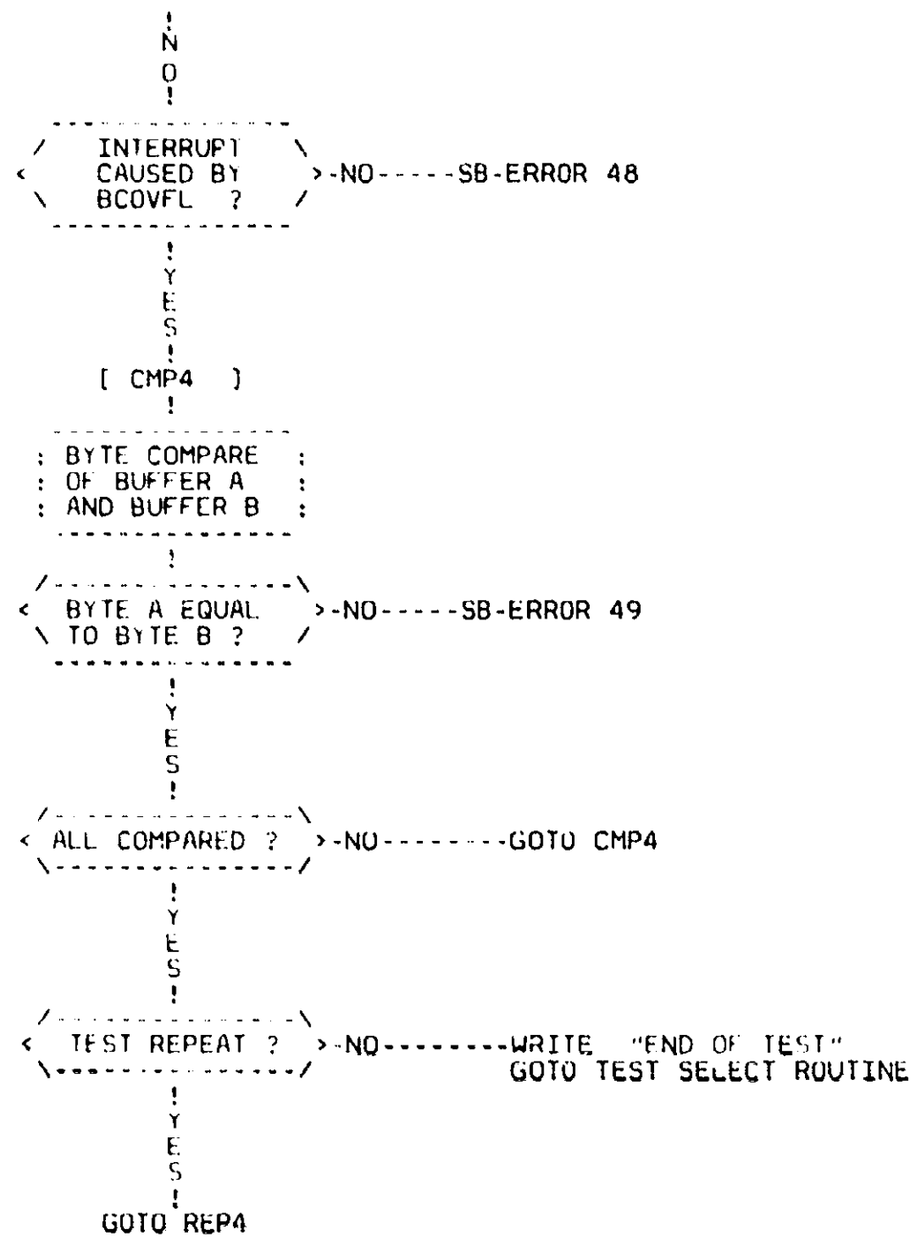


********** TEST 4 : DMA-TRANSFER FROM B TO A (B IS TALKER)

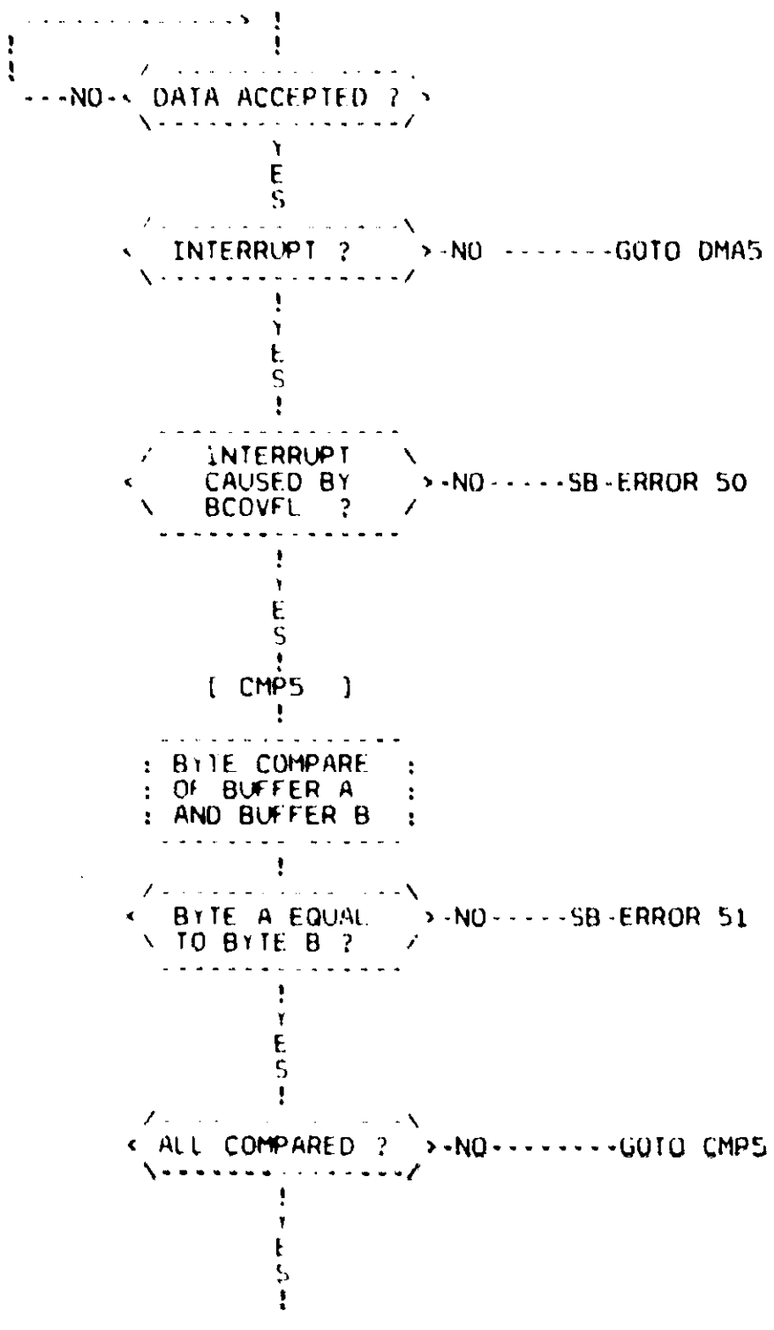


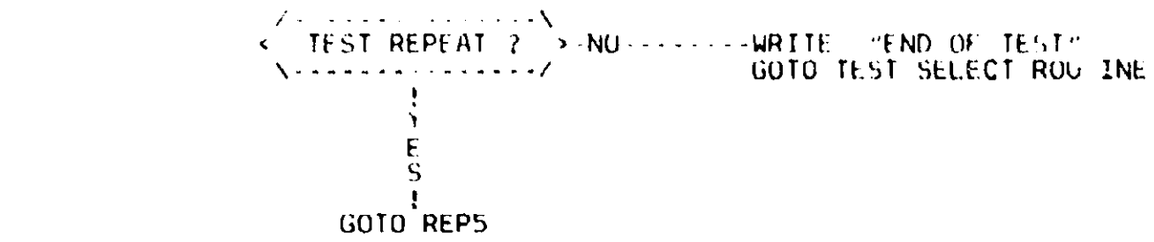

Y
E
S
[BCIN4]
ASK FOR BYTE COUNT
[REP4]
PREPARE BUFFER A AND B
MASTER CLEAR
SET A TO CACS
SET A TO LACS
SET B TO TACS
GOTO STANDBY
PREPARE BAR AND BCR, ENABLE DMA AND INTERRUPT
DATA ACCEPTED ?
NO
ERROR*45
Y
E
S
[DMA4]
INTERRUPT ?
YES
GOTO INTB4
N
O


DATA ACCEPTED ?
NO
YES
IOR HIGH BYTE
TO BUFFER A
GOTO DMA4
[INTB4]
PREPARE DELAY
[LAST4]
DATA ACCEPTED ?
NO
YES
DELAY DONE ?
NO
GOTO LAST4
YES
TO LESS BYTES
TRANSFERRED ?
YES
ERROR*46
NO
TO MANY BYTES
TRANSFERRED ?
YES
ERROR*47

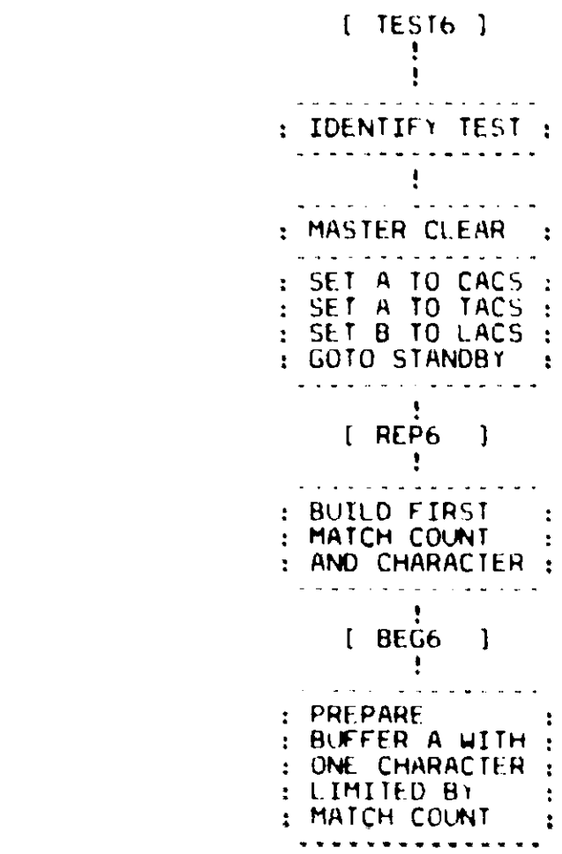

NO
INTERRUPT CAUSED BY BCOVFL ?
NO
SB-ERROR 48
YES
[CMP4]
BYTE COMPARE OF BUFFER A AND BUFFER B
BYTE A EQUAL TO BYTE B ?
NO
SB-ERROR 49
YES
ALL COMPARED ?
NO
GOTO CMP4
YES
TEST REPEAT ?
NO
WRITE "END OF TEST"
GOTO TEST SELECT ROUTINE
YES
GOTO REP4

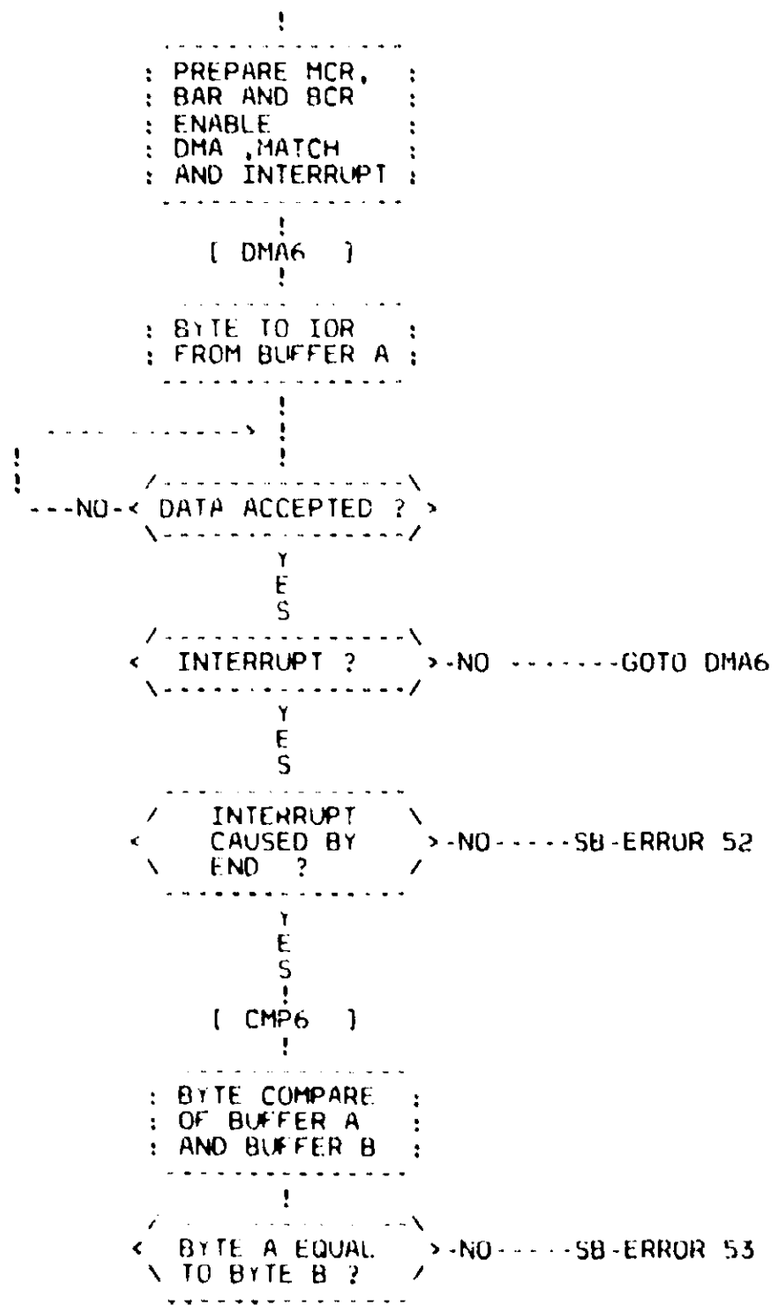
********** TEST 5 : DMA-TRANSFER FROM A TO B (B IS LISTENER)

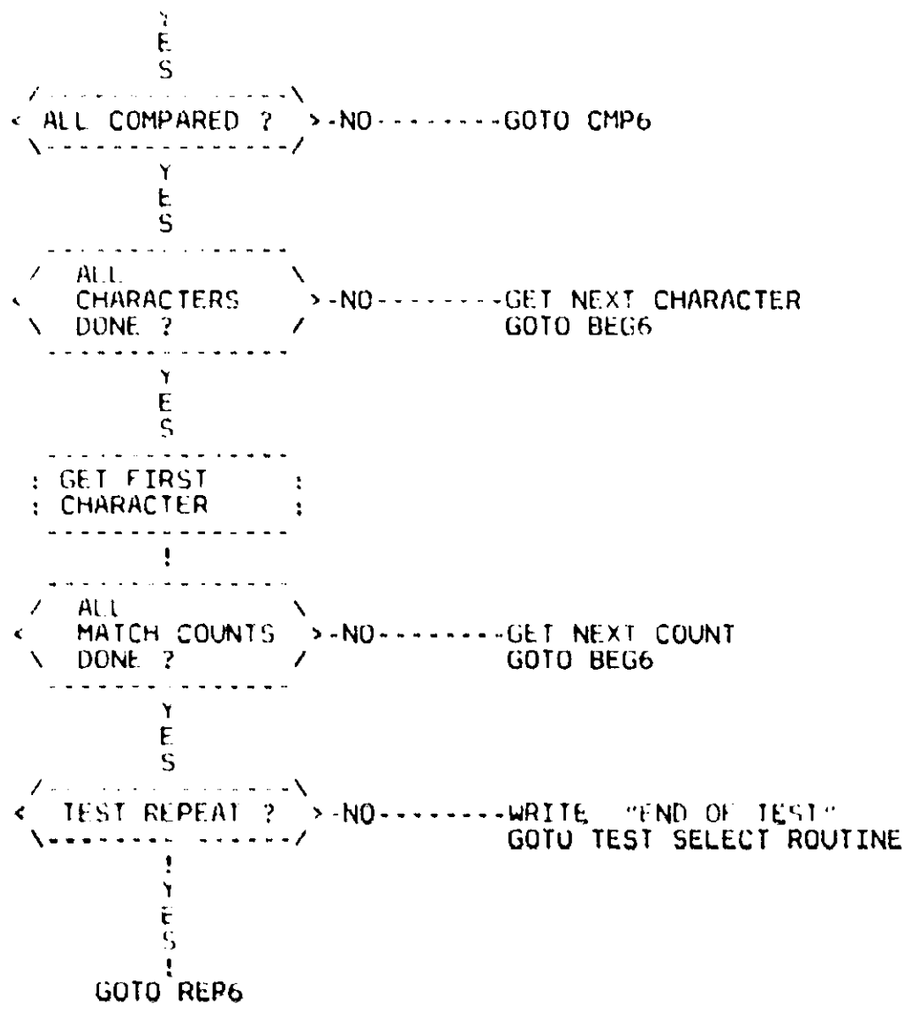

DATA ACCEPTED ?
NO
YES
INTERRUPT ?
NO
GOTO DMA5
YES
INTERRUPT CAUSED BY BCOVFL ?
NO
SB-ERROR 50
YES
[CMP5]
BYTE COMPARE OF BUFFER A AND BUFFER B
BYTE A EQUAL TO BYTE B ?
NO
SB-ERROR 51
YES
ALL COMPARED ?
NO
GOTO CMP5
YES



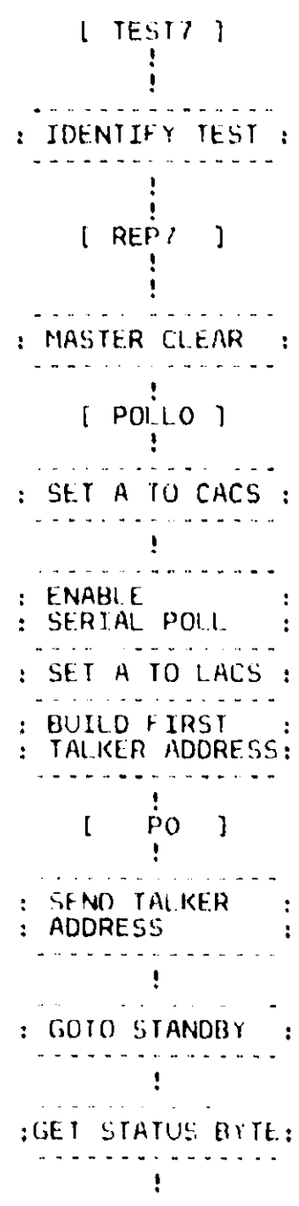
********** TEST 6 : MATCH CHARACTER REGISTER TEST (B IS LISTENER)

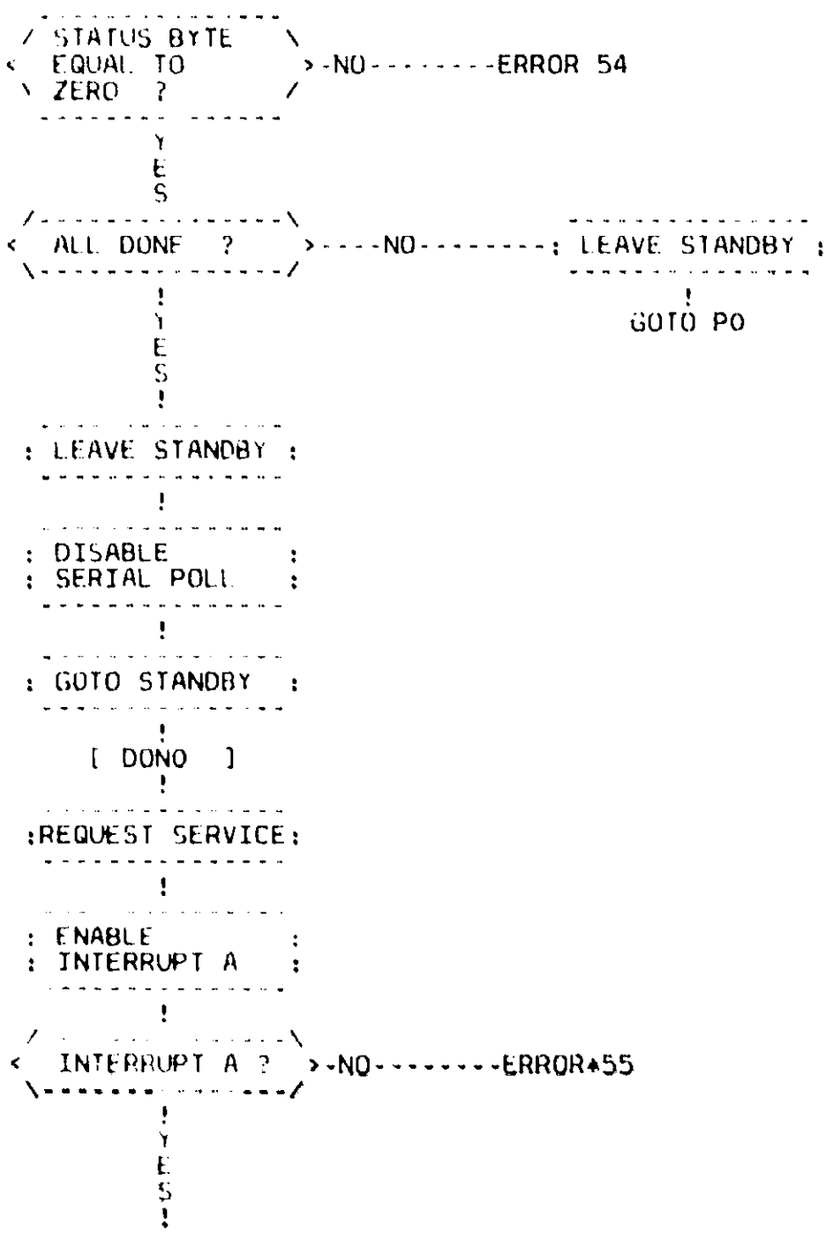


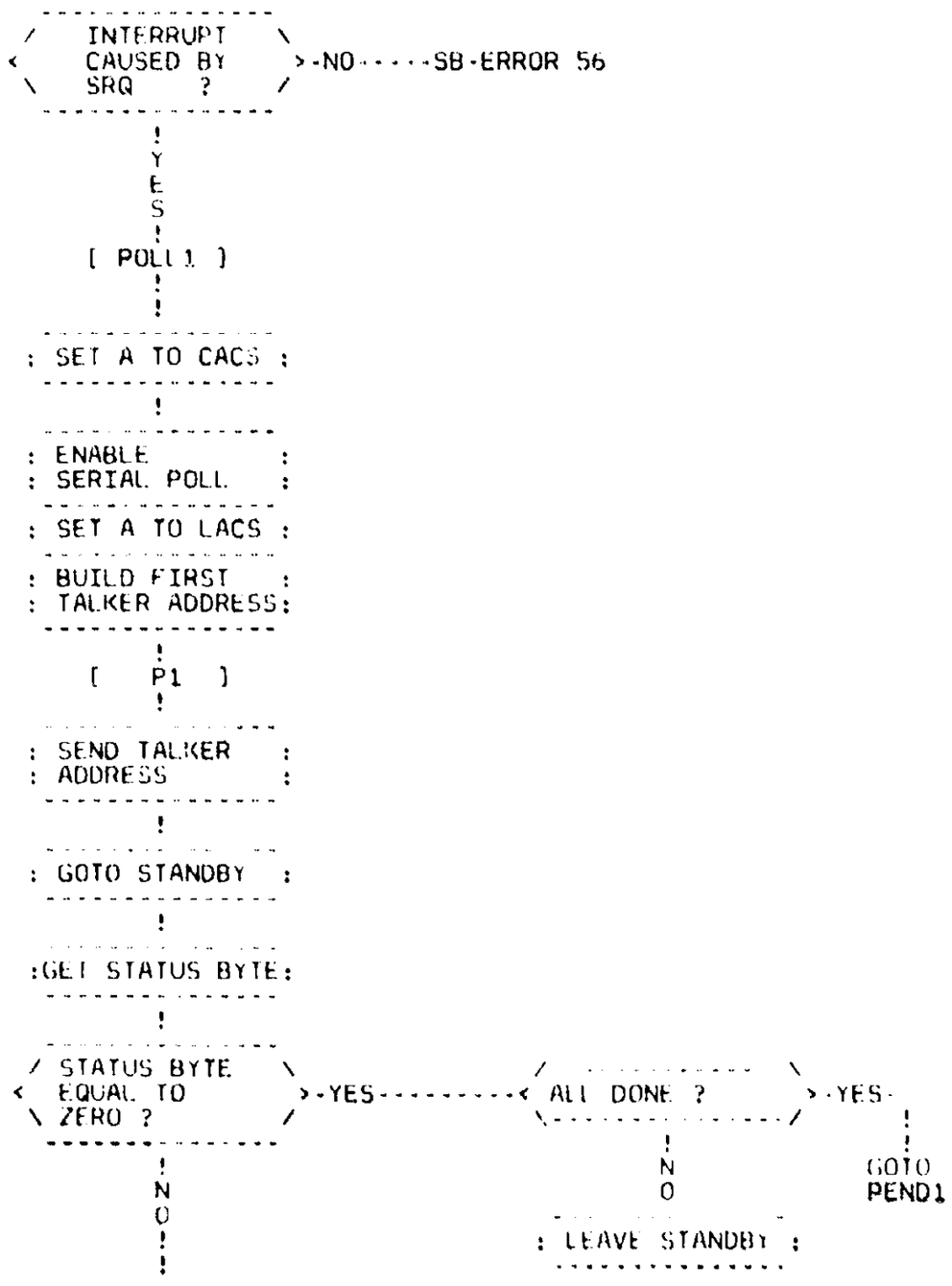

PREPARE MCR, BAR AND BCR ENABLE DMA ,MATCH AND INTERRUPT
[DMA6]
BYTE TO IOR FROM BUFFER A
DATA ACCEPTED ?
NO
YES
INTERRUPT ?
NO
GOTO DMA6
YES
INTERRUPT CAUSED BY END ?
NO
SB-ERROR 52
YES
[CMP6]
BYTE COMPARE OF BUFFER A AND BUFFER B
BYTE A EQUAL TO BYTE B ?
NO
SB-ERROR 53

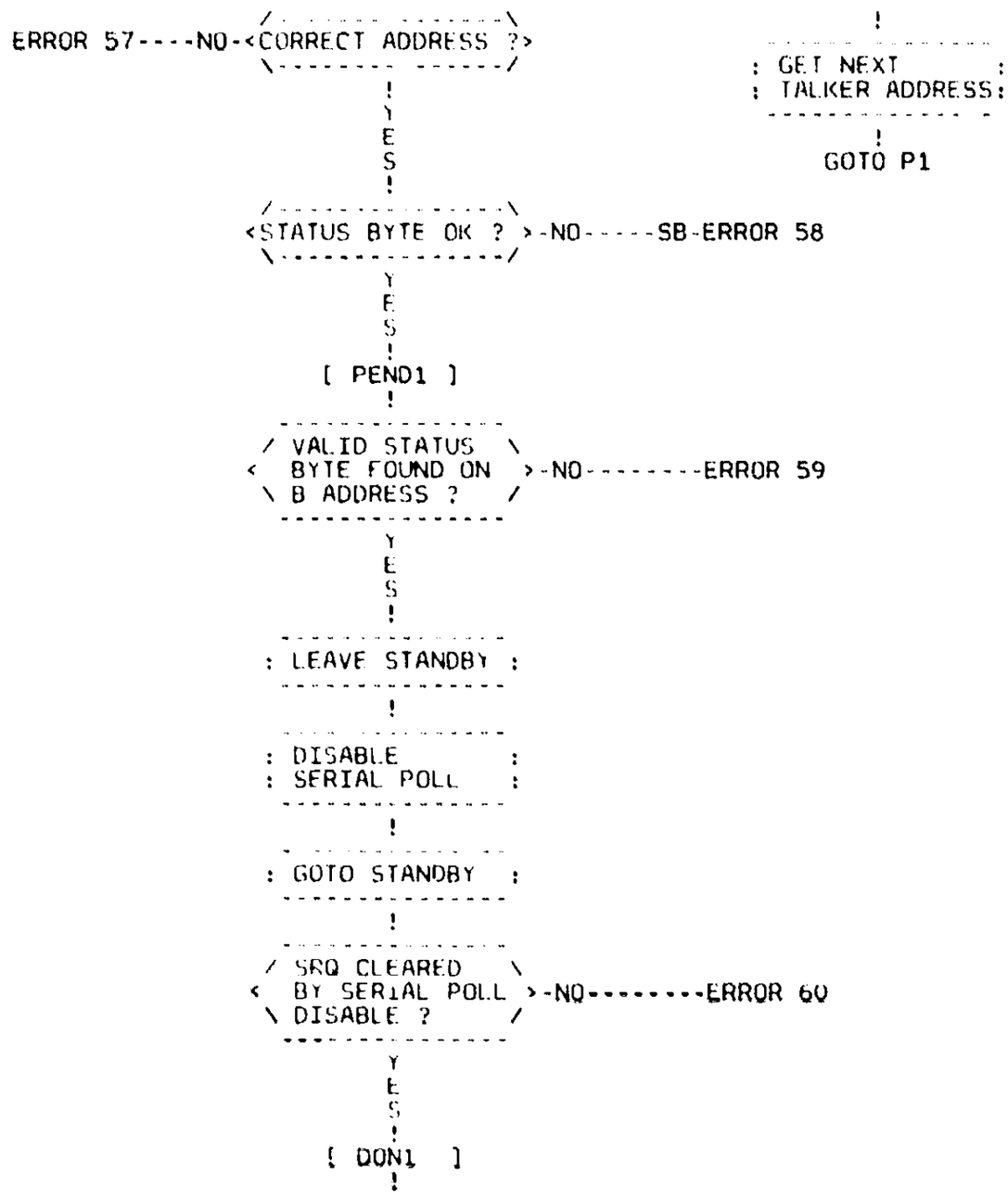

YES
ALL COMPARED ?
NO
GOTO CMP6
YES
ALL CHARACTERS DONE ?
NO
GET NEXT CHARACTER
GOTO BEG6
YES
GET FIRST CHARACTER
ALL MATCH COUNTS DONE ?
NO
GET NEXT COUNT
GOTO BEG6
YES
TEST REPEAT ?
NO
WRITE "END OF TEST"
GOTO TEST SELECT ROUTINE
YES
GOTO REP6

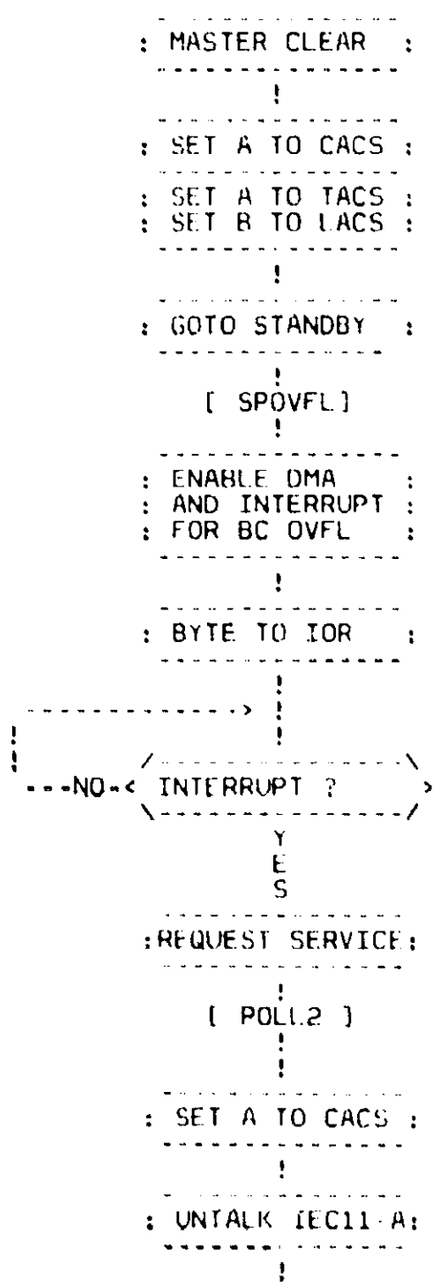
********** TEST 7 : SERIAL POLL PROCEDURE TEST

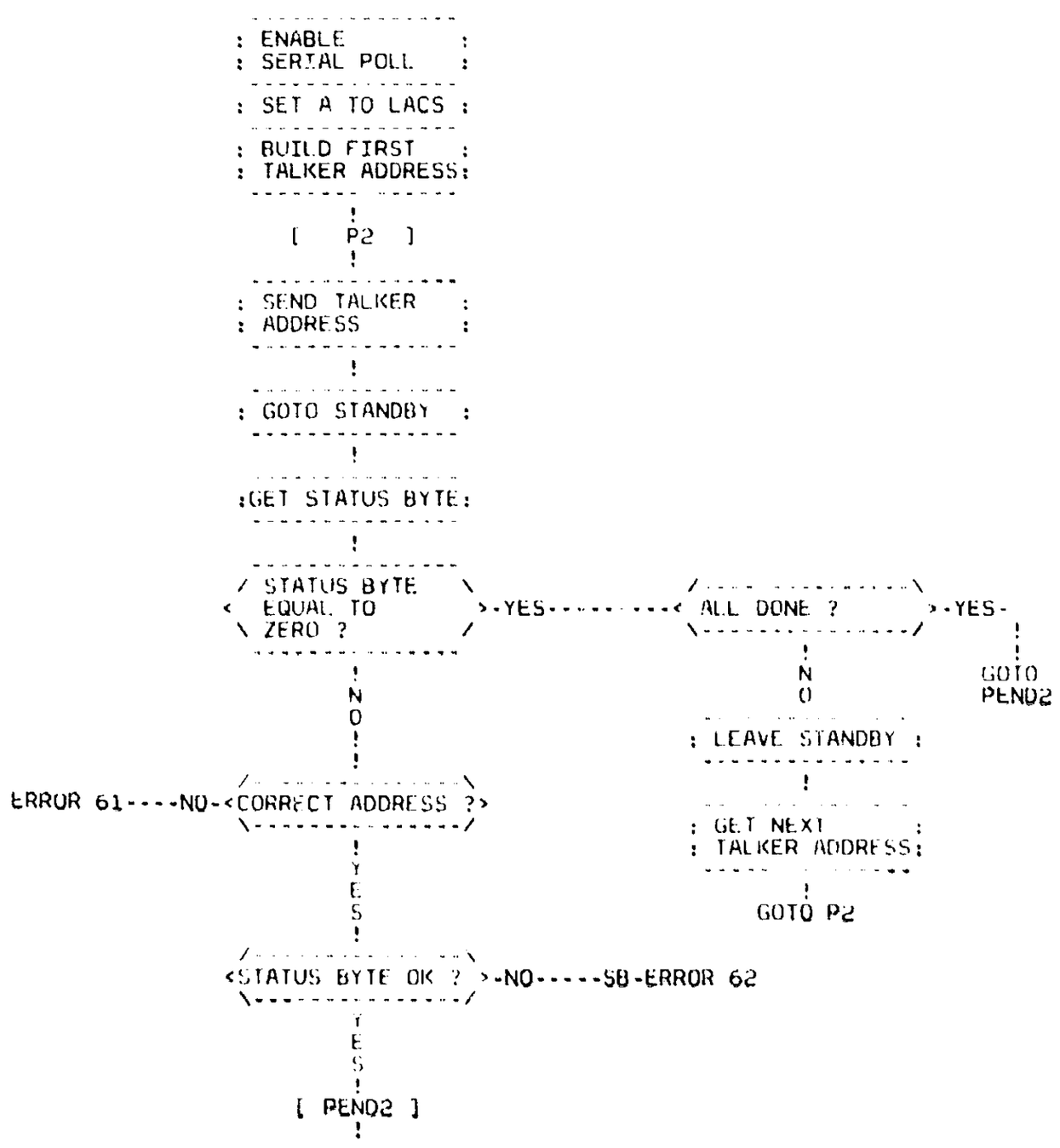


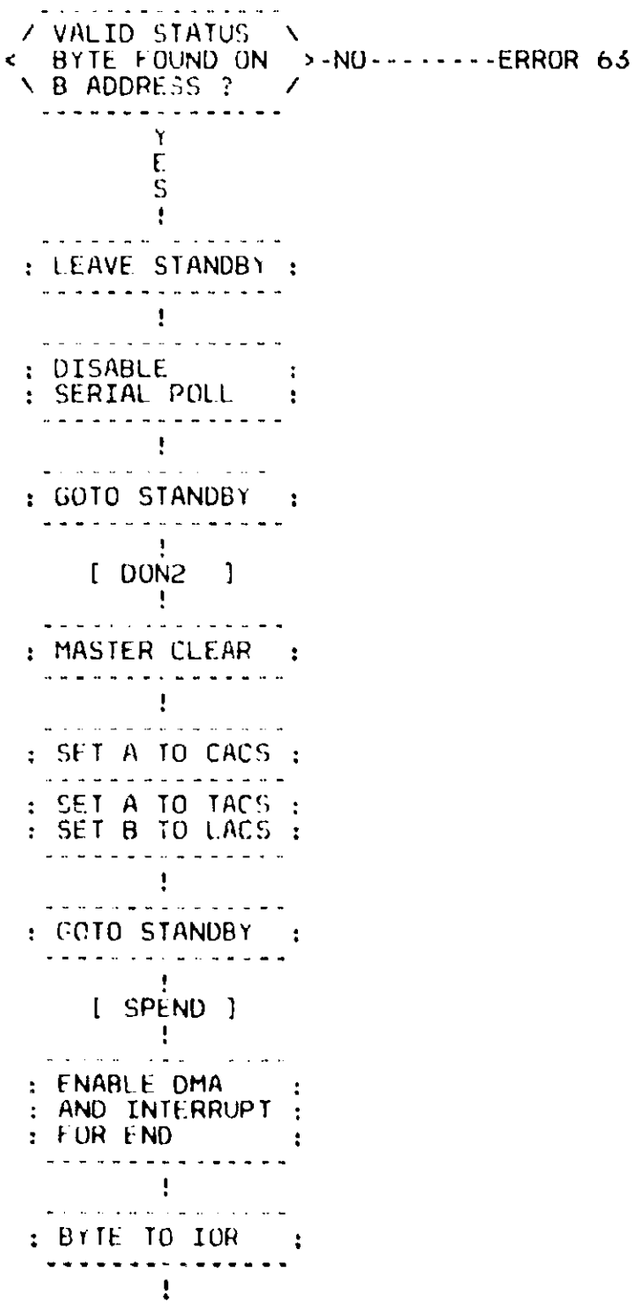

STATUS BYTE EQUAL TO ZERO ?
NO
ERROR 54
YES
ALL DONE ?
NO
LEAVE STANDBY
GOTO PO
YES
LEAVE STANDBY
DISABLE SERIAL POLL
GOTO STANDBY
[DONO]
REQUEST SERVICE
ENABLE INTERRUPT A
INTERRUPT A ?
NO
ERROR*55
YES

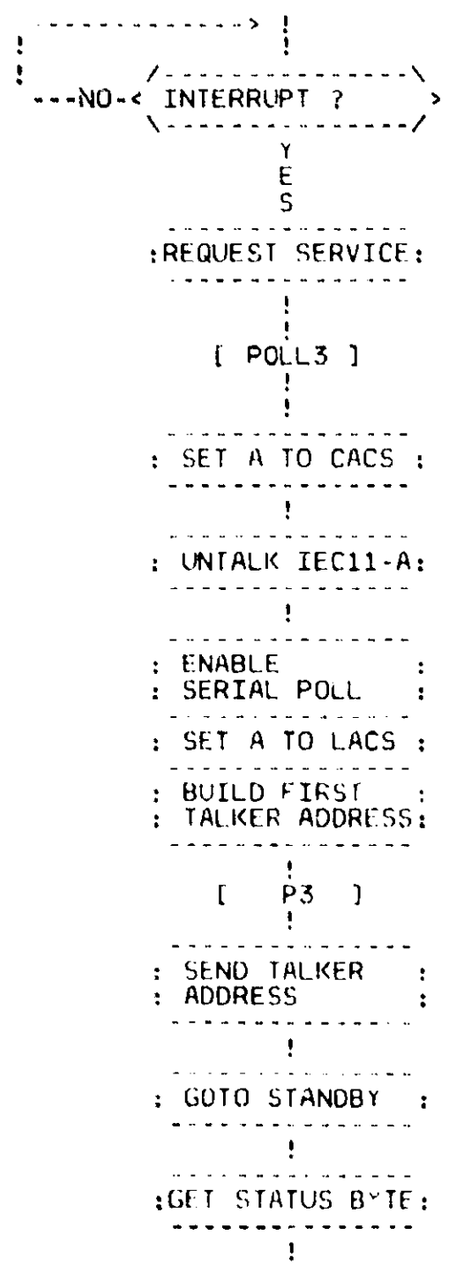

INTERRUPT CAUSED BY SRQ ?
NO
SB-ERROR 56
YES
[POLL 1]
SET A TO CACS
ENABLE SERIAL POLL
SET A TO LACS
BUILD FIRST TALKER ADDRESS
[P1]
SEND TALKER ADDRESS
GOTO STANDBY
GET STATUS BYTE
STATUS BYTE EQUAL TO ZERO ?
YES
ALL DONE ?
YES
GOTO PEND1
NO
NO
LEAVE STANDBY

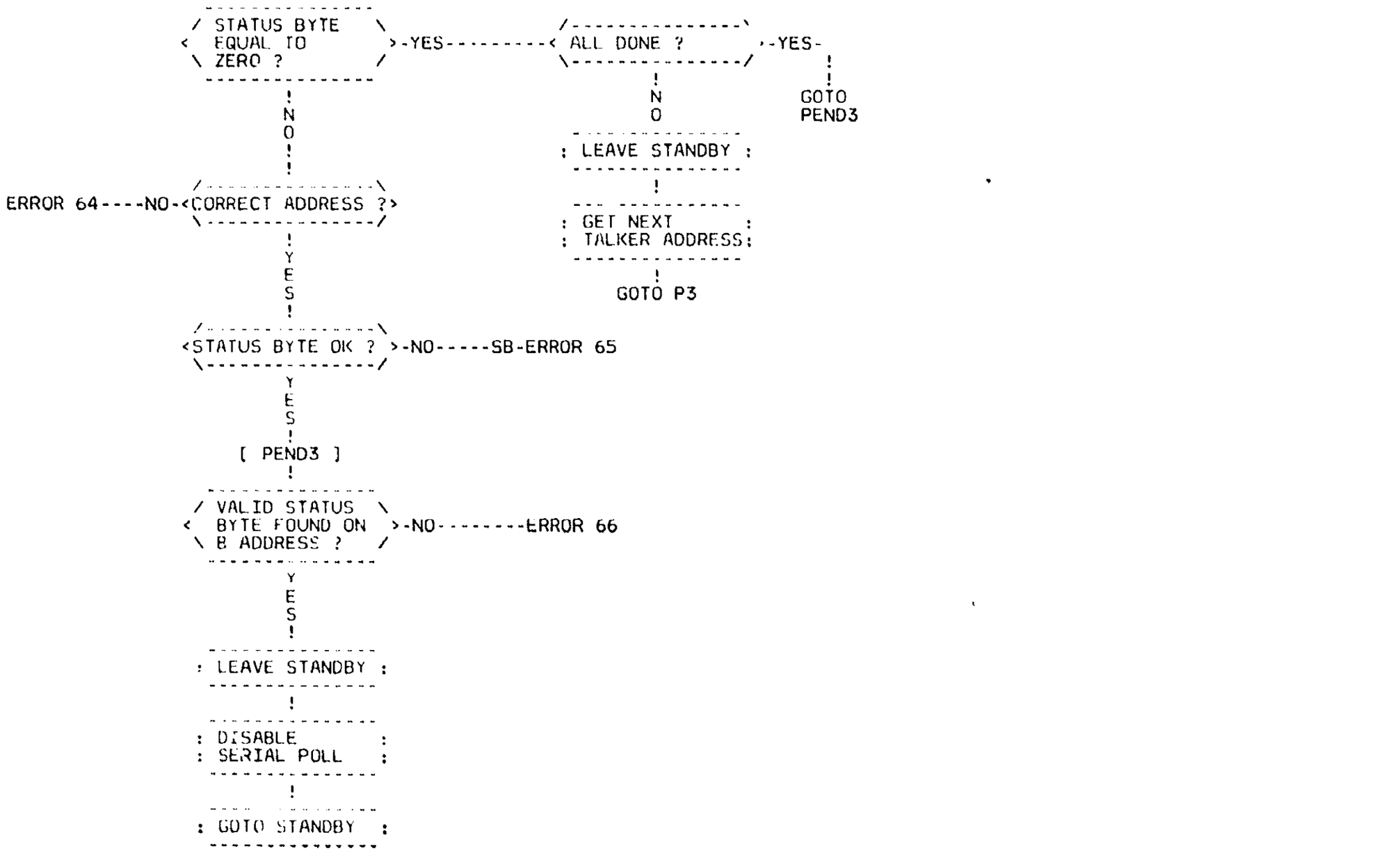

ERROR 57
NO
CORRECT ADDRESS ?
YES
STATUS BYTE OK ?
NO
SB-ERROR 58
YES
[PEND1]
VALID STATUS BYTE FOUND ON B ADDRESS ?
NO
ERROR 59
YES
LEAVE STANDBY
DISABLE SERIAL POLL
GOTO STANDBY
SRQ CLEARED BY SERIAL POLL DISABLE ?
NO
ERROR 60
YES
[DON1]
GET NEXT TALKER ADDRESS
GOTO P1

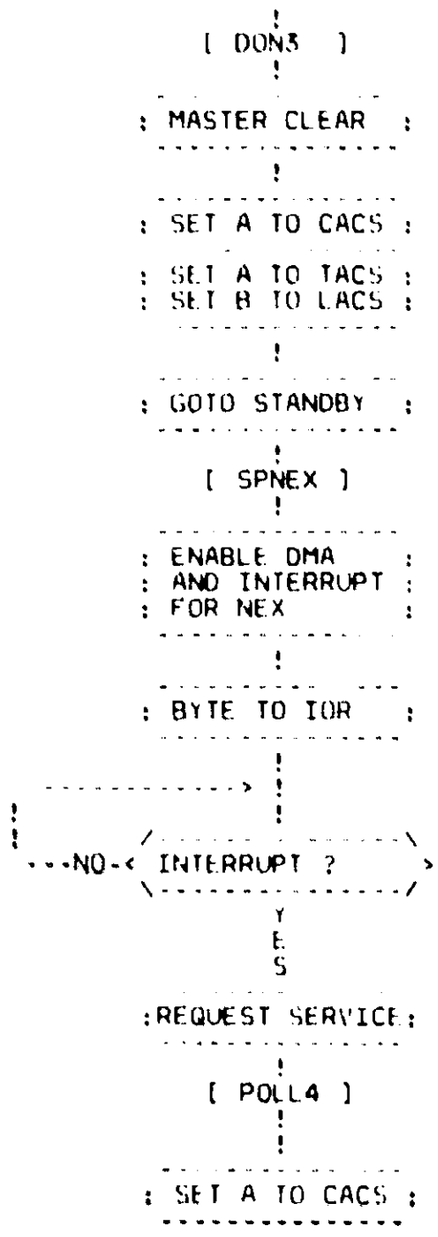

MASTER CLEAR
SET A TO CACS
SET A TO TACS
SET B TO LACS
GOTO STANDBY
[SPOVFL]
ENABLE DMA
AND INTERRUPT
FOR BC OVFL
BYTE TO IOR
INTERRUPT ?
NO
YES
REQUEST SERVICE
[POLL2]
SET A TO CACS
UNTALK IEC11-A

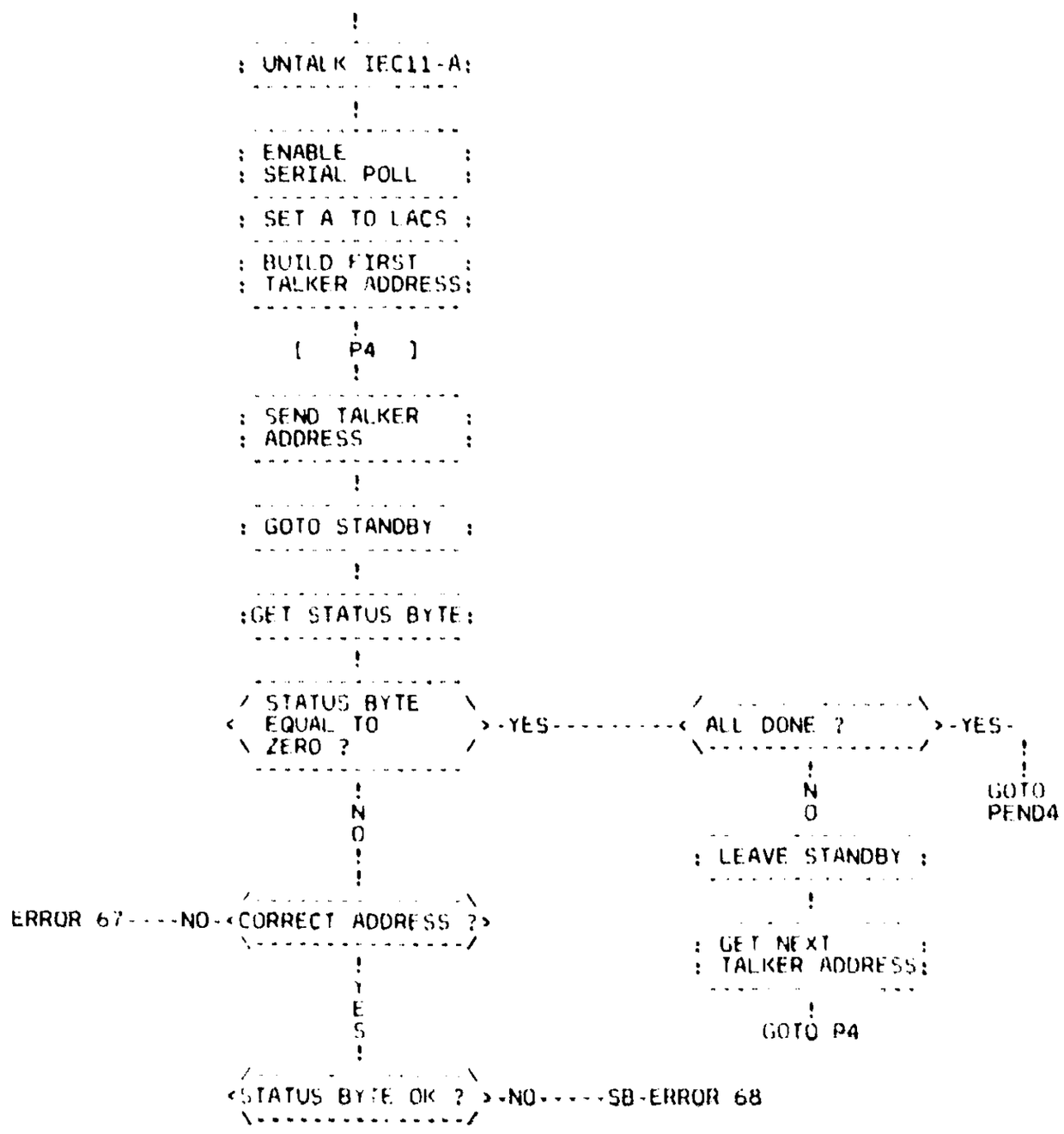

ENABLE SERIAL POLL
SET A TO LACS
BUILD FIRST TALKER ADDRESS
[P2]
SEND TALKER ADDRESS
GOTO STANDBY
GET STATUS BYTE
STATUS BYTE EQUAL TO ZERO ?
YES
NO
ALL DONE ?
YES
NO
GOTO PEND2
LEAVE STANDBY
GET NEXT TALKER ADDRESS
GOTO P2
ERROR 61
NO
CORRECT ADDRESS ?
YES
STATUS BYTE OK ?
NO
SB-ERROR 62
YES
[PEND2]

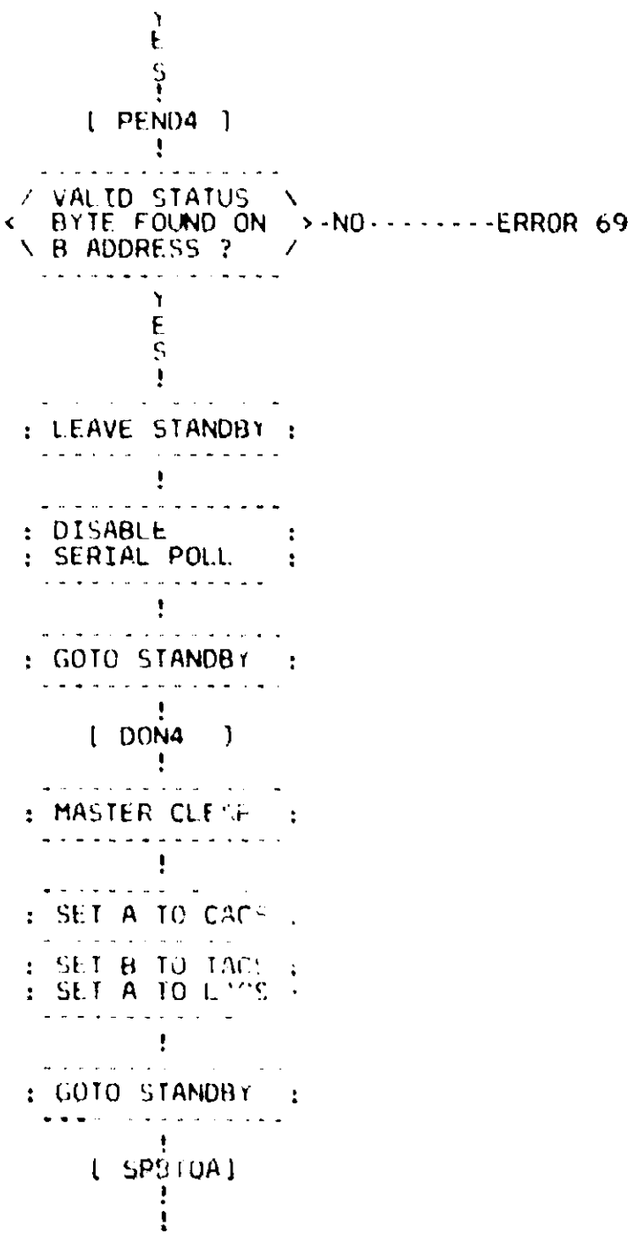

VALID STATUS BYTE FOUND ON B ADDRESS ?
NO
ERROR 63
YES
LEAVE STANDBY
DISABLE SERIAL POLL
GOTO STANDBY
DON2
MASTER CLEAR
SET A TO TACS
SET B TO LACS
GOTO STANDBY
SPEND
ENABLE DMA AND INTERRUPT FOR END
BYTE TO IOR

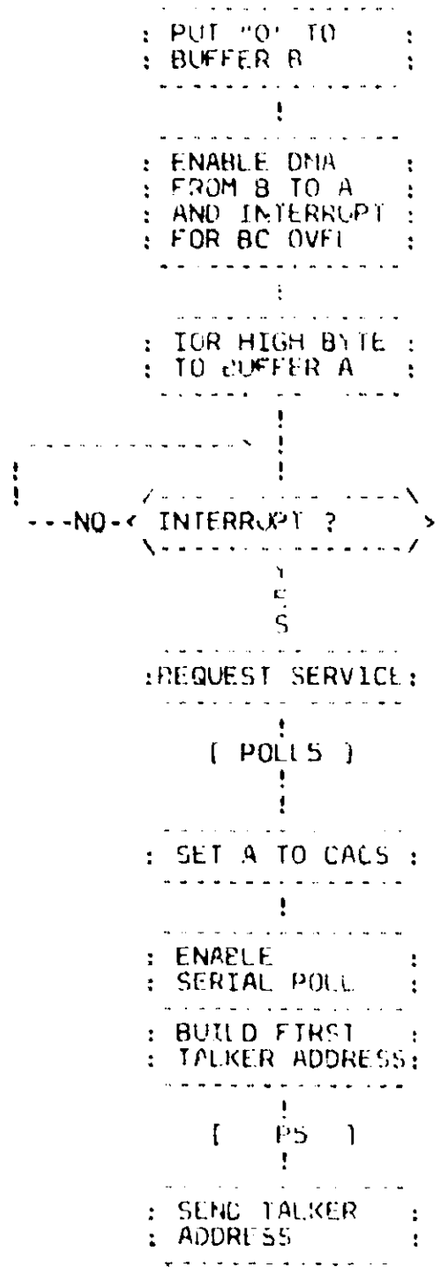

INTERRUPT ?
NO
YES
REQUEST SERVICE
[POLL3]
SET A TO CACS
UNTALK IEC11-A
ENABLE SERIAL POLL
SET A TO LACS
BUILD FIRST TALKER ADDRESS
[P3]
SEND TALKER ADDRESS
GOTO STANDBY
GET STATUS BYTE

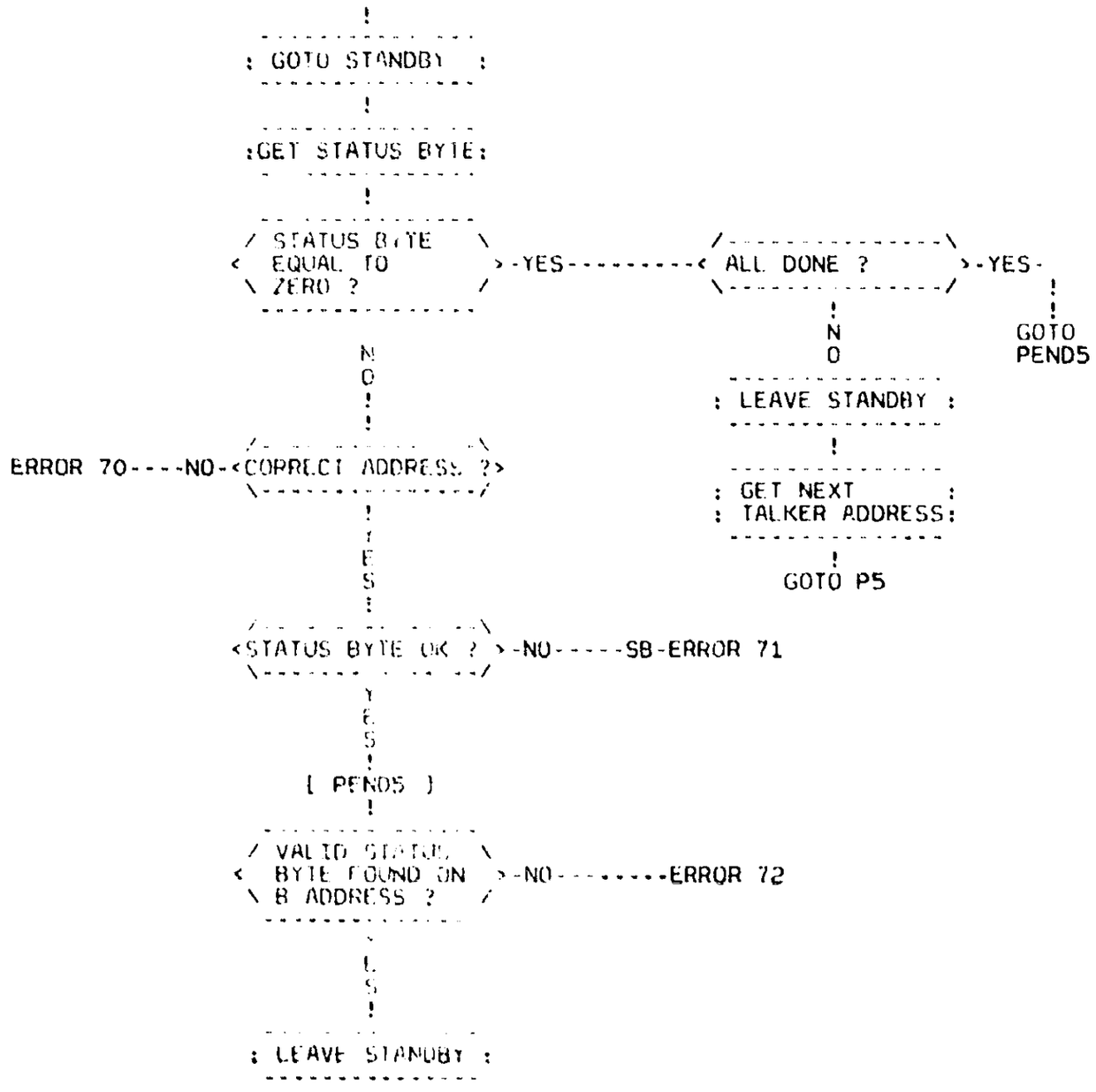

STATUS BYTE EQUAL TO ZERO ?
YES
ALL DONE ?
YES
NO
GOTO PEND3
NO
LEAVE STANDBY
GET NEXT TALKER ADDRESS
GOTO P3
ERROR 64
NO
CORRECT ADDRESS ?
YES
STATUS BYTE OK ?
NO
SB-ERROR 65
YES
[PEND3]
VALID STATUS BYTE FOUND ON B ADDRESS ?
NO
ERROR 66
YES
LEAVE STANDBY
DISABLE SERIAL POLL
GOTO STANDBY

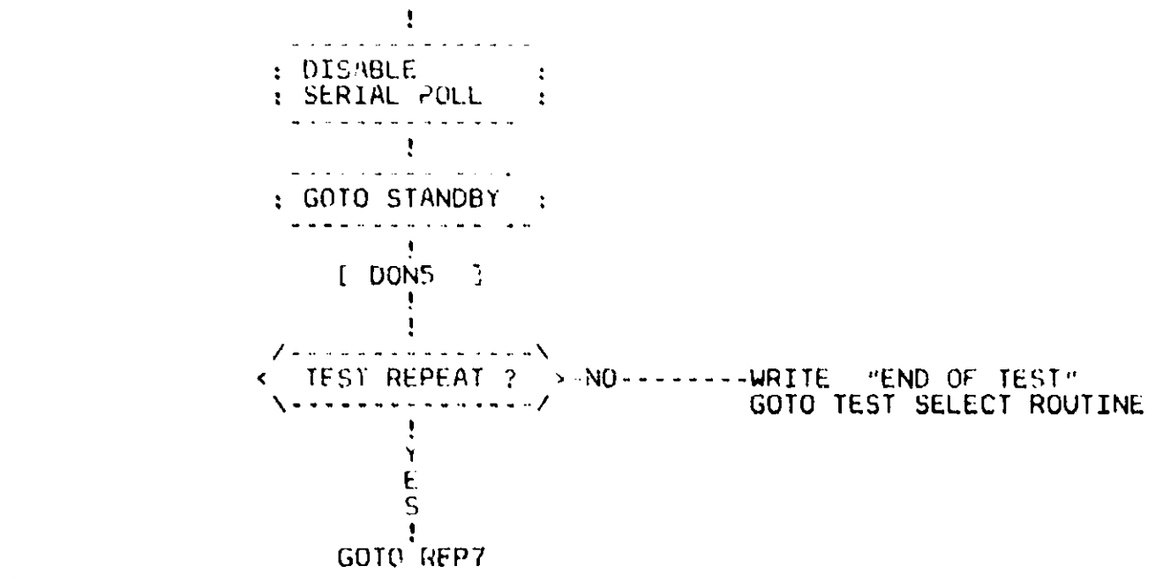

[DON3]
MASTER CLEAR
SET A TO CACS
SET A TO TACS
SET B TO LACS
GOTO STANDBY
[SPNEX]
ENABLE DMA
AND INTERRUPT
FOR NEX
BYTE TO IOR
NO
INTERRUPT ?
YES
REQUEST SERVICE
[POLL4]
SET A TO CACS


UNTALK IEC11-A
ENABLE SERIAL POLL
SET A TO LACS
BUILD FIRST TALKER ADDRESS
P4
SEND TALKER ADDRESS
GOTO STANDBY
GET STATUS BYTE
STATUS BYTE EQUAL TO ZERO ?
YES
ALL DONE ?
YES
GOTO PEND4
NO
LEAVE STANDBY
ERROR 67
NO
CORRECT ADDRESS ?
GET NEXT TALKER ADDRESS
GOTO P4
YES
STATUS BYTE OK ?
NO
SB-ERROR 68


Y
E
S
[PEND4]
VALID STATUS BYTE FOUND ON B ADDRESS ?
NO
ERROR 69
Y
E
S
LEAVE STANDBY
DISABLE SERIAL POLL
GOTO STANDBY
[DON4]
MASTER CLEAR
SET A TO CACS
SET B TO TACS
SET A TO LMS
GOTO STANDBY
[SPSTOA]


PUT "0" TO BUFFER B
ENABLE DMA FROM B TO A AND INTERRUPT FOR BC OVFL
IOR HIGH BYTE TO BUFFER A
INTERRUPT ?
NO
YES
REQUEST SERVICE
[POLLS]
SET A TO CACS
ENABLE SERIAL POLL
BUILD FIRST TALKER ADDRESS
[PS]
SEND TALKER ADDRESS


GOTO STANDBY
GET STATUS BYTE
STATUS BYTE EQUAL TO ZERO ?
YES
ALL DONE ?
YES
GOTO PEND5
NO
LEAVE STANDBY
GET NEXT TALKER ADDRESS
GOTO P5
ERROR 70
NO
CORRECT ADDRESS ?
YES
STATUS BYTE OK ?
NO
SB-ERROR 71
YES
[PEND5]
VALID STATUS BYTE FOUND ON B ADDRESS ?
NO
ERROR 72
YES
LEAVE STANDBY



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.
;
;
;               O P E R A T I N G   P R O C E D U R E S
;               ***************************************
;
;               TO OPERATE AN ERROR SEE UNDER  A)
;               TO MODIFY A LOCATION SEE UNDER  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 DIGNOSTIC MONITOR WAITS FOR
;               INPUT OF OPTIONS.
;
;               PRESS <CNTRL. C> AND THE "PROMPT STRING" IS PRINTED AT
;               THE BEGINNING OF A NEW LINE.
;
;               CDM>
;               ***     (PROGRAM PRINTS WILL BE UNDERLINED FROM NOW)
;
;               NOW YOU CAN SELECT THE OPTIONS YOU NEED TO FIND THE ERROR.
;               THE SYNTAX OF A COMMAND LINE IS AS FOLLOWS:
;
;               CDM>ERR:/OPT1/OPT2/OPT3/...OPTN/CONT <CR>
;                   ***
;
;               IF THERE IS A ERROR IN THE INPUT LINE THE FOLLOWING
;               MESSAGE IS PRINTED:
;
;               SYNTAX ERROR !
;               ??????????????
;
;               YOU CAN NOW REPEAT  THE INPUT.
;
;               ANY COMBINATION OF OPTIONS IS LEGAL, BUT PROBABLY NOT USEFUL.
;               WHEN YOU TYPE  /SCANS  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.
;
;       8)      OPTION          :       DMP
;               THIS OPTION CAN BE USED TO CHECK THE CONTENT OF A
;               MEMORY LOCATION OR A REGISTER.
;               IT CAN ONLY BE CALLED AFTER AN ERROR OR WHEN YOU INTERRUPTED
;               THE RUNNING PROGRAM BY PRESSING "CNTRL C".
```

```
;
;               EXAMPLE:
;               AFTER AN ERROR PRINTOUT YOU PRESSED "CNTRL C" AND THE MONITOT
;               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/O/1000
;               ****
;
;               AND A RETURN, THE MONITOR WILL PRINT AT THE BEGINNING OF
;               THE NEXT LINE THE CONTENT OF LOCATION 1000:
;
;               CDM>DMP/O/1000
;               ****
;               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 DIGIT  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
;               +++++++
```

TABLE OF CONTENTS

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

```
 7 000000                                .PSECT  CSSMON
 8                                ;
 9                                ;       MZ001
10                                ;       *****
11                                ;       THIS CHANGE ALLOWS TO USE A MACRO THAT EITHER CLEARS
12                                ;       THE PSW BY A "CLR" OR A "MTPS".
13                                ;
14                                ;       GS001
15                                ;       *****
16                                ;       THIS CHANGE ALLOWS TO START THE PROGRAM AT LOC. 200
17                                ;
18                                ;
19 000000                               .ASECT
20          000000                      .=0
21          000004                      .REPT   4
22                                      .+2                     ;POINT TO VECTOR +2
23                                      IOT                     ;HANDLE ILLEGAL TRAP
24                                      .ENDR                   ;
25 000020  003056                       $..IO1                  ;ILLEGAL TRAP-HANDLER
26 000022  000000                       0                       ;PRIORITY = 0
27 000024  003126                       $..PWR                  ;POWERFAIL-RESTART-HANDLER
28 000026  000000                       0                       ;PRIORITY = 0
29 000030  002326                       $..EMT                  ;EMT-HANDLER
30 000032  000000                       0                       ;PRIORITY = 0
31 000034  002524                       $..TRP                  ;TAP-HANDLER
32 000036  000000                       0                       ;PRIORITY = 0
33          000030                      .REPT   30              ;                       ;**GS001**
34                                      .+2                     ;POINTER TO VECTOR +2   ;**GS001**
35                                      IOT                     ;HANDLE ILLEGAL TRAP    ;**GS001**
36                                      .ENDR                   ;                       ;**GS001**
37 000200  000137  001000                JMP     10$             ;INITIALIZE SYSTEM      ;**GS001**
38          000137                      .REPT   137             ;                       ;**GS001**
39                                      .+2                     ;POINTER TO VECTOR +2   ;**GS001**
40                                      IOT                     ;HANDLE ILLEGAL TRAP    ;**GS001**
41                                      .ENDR                   ;                       ;**GS001**
42 001000  000137  001204       10$:    JMP     START           ; INITIALIZE SYSTEM
43          000100                      .REPT   100
44                                      0                       ;STACK-AREA
45                                      .ENDR
```

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

```
                                        .SBTTL  MEMORY TEST WITH MEMORY MANAGEMENT UNIT SETUP
                                ;
                                ;  SUBROUTINE:  M..TST
                                ;  ***********  ******
                                ;
                                ;THIS ROUTINE CHECKS THE AVAILABILITY OF A MEMORY MANAGEMENT UNIT
                                ;IN PDP-11 PROCESSORS AND SETS KISAR0, ..SAR5 TO THE LOW 24K-WORDS
                                ;OF THE COMPUTER MEMORY .
                                ;KISAR7 IS USED TO MAP THE SO-CALLED I/O PAGE .
                                ;KISAR6 IS USED TO MAP THE HIGHEST MEMORY ADDRESS CHECK POINTER .
                                ;
                                ;INPUT:         (SP)    *RETURN ADDRESS
                                ;
                                ;OUTPUT:        R0      *LOW 16 ADDRESS BITS OF HIGHEST MEMORY ADDRESS
                                ;               R1      *BITS 0,1 ADDRESS BITS 16,17
                                ;
                                ;AUTHOR:        BERNHARD BAUDISCH CSS/DP MUNICH 19-DEC-78
                                ;
                                ;
001516                          M..TST:                         ;REFERENCE LABEL
001516  012737  001776  000250          MOV     #M..TRP,@#250   ;SETUP SEGMENTATION TRAP CATCHER
001524  005037  000252                  CLR     @#252           ;...
001530  012737  001776  000004          MOV     #M..TRP,@#4     ;SETUP TRAP-4 CATCHER
001536  005037  000006                  CLR     @#6             ;...
001542  004737  002006                  CALL    M..MMM          ;SETUP DEFAULT MAPPING,ENABLE MM
001546  012737  007400  172354          MOV     #7400,KISAR6    ;MAP WITH KISAR6 THROUGH MEMORY
001554  012737  000001  007020          MOV     #1,$$MTYP       ;REMEMBER THAT WE HAVE MEMORY MANAGEMENT!
001562  012700  157776                  MOV     #157776,R0      ;GET HIGHEST ADDRESS WITHIN APR6 RANGE
001566  012737  001766  000004          MOV     #M..HIM,@#4     ;SET UP TRAP 4 CATCHER
001574  005710                  20$:    TST     @R0             ;SEE IF ADDRESS PRESENT
001576  103031                  BCC     M..DET                  ;CC IF ADDRESS FOUND
001600  162737  000200  172354          SUB     #200,KISAR6     ;MAP TO NEXT LOWER 4K WORD BANK
001606  002401                  BLT     30$                     ;THIS CASE SHOULD NEVER HAPPEN
001610  000771                  BR      20$                     ;CHECK NEXT LOWER 4-K WORDS
001612                          30$:                            ;HERE ONLY IF MAIN PROCESSOR HARDWARE ERROR
001612                          PSET    #0                      ;SET PRIORITY TO 0
001636                          WRITE   MEMDEF                  ;MEMORY MANAGEMENT DEFECT
001646  012716  001374                  MOV     #M..TSE,(SP)    ;MODIFY RETURN ADDRESS TO UNMAPPED SYSTEM
001652  042737  000001  177572          BIC     #BIT0,SR0       ;DISABLE MM
001660  000422                  BR      M..RET                  ;AND GO AHEAD LIKE UNMAPPED SYSTEM
001662                          M..DET:                         ;HIGHEST MEMORY ADDRESS FOUND
001662                          PUSH    KISAR6                  ;GET MAP REGISTER CONTENTS
                        000005          .REPT   5               ;
                                        ASL     (SP)            ;PUT ADDRESS BIT 17 INTO PSW CARRY
                                        .ENDR                   ;
001700  103402                  BCS     40$                     ;CS IF ADDRESS BIT 17 SET
001702  005001                  CLR     R1                      ;HERE IF BIT17 NOT SET
001704  000402                  BR      45$                     ;...
001706  012701  000001          40$:    MOV     #BIT0,R1        ;FLAG BIT 17 FOUND
001712  006316                  45$:    ASL     (SP)            ;GET ADDRESS BIT 16
001714  006101                  ROL     R1                      ;INSERT BIT IN MEMORY EXTENSION MASK
001716  042700  160000                  BIC     #160000,R0      ;MASK OUT APR SELECTION BITS
001722  050016                  BIS     R0,(SP)                 ;BUILD 16 BIT BASE ADDRESS
001724  012600                  MOV     (SP)+,R0                ;  ->R0 FOR OUTPUT TO CALLER
001726  012737  000252  000250  M..RET: MOV     #252,@#250      ;RESET SEGMENTATION TRAP CATCHER
001734  012737  000004  000252          MOV     #4,@#252        ;...
001742  012737  000006  000004          MOV     #6,@#4          ;RESET TRAP-4 CATCHER
```

```
                                                ;...
                                        RTS     PC              ;RETURN TO CALLER
                                U..DET:                         ;HERE IF MEMORY MANAGEMENT ADDRESS NOT FOUND
                                        MOV     #M..TSE,(SP)    ;MODIFY RETURN ADDRESS
                                        BR      M..RET          ;RETURN COMMON
                                M..HIM:                         ;TRAP-4 AND SEGMENTATION TRAP CATCHER
                                        BIS     #BIT0,2(SP)     ;SET CARRY IN RETURN PSW SETUP
                                        RTI                     ;RETURN WITH CARRY SET INDICATOR
                                ;
                                M..TRP:
                                        CMP     (SP)+,(SP)+     ;CLEAN STACK OF INT. VALUES
                                        TST     (SP)+           ;CLEAN STACK FROM RETURN ADDRESS
                                        JMP     U..DET          ;NO MEMORY MANAGEMENT UNIT AVAILABLE
                                ;
                                ;
                                ;
                                ;       SUBROUTINE M..MMM
                                ;       ****************
                                ;
                                ;       THIS ROUTINE IS CALLED TO SET UP THE APR AND PDR
                                ;       REGISTER WITH DEFAULT VALUES AND TO ENABLE MEMORY
                                ;       MANAGEMENT.
                                ;
                                ;       INPUT:  NONE
                                ;       OUTPUT: ALL PDRS ARE PRESET WITH 77406
                                ;               APR0 IS PRESET WITH 0
                                ;               APR1    "           200
                                ;               APR2    "           400
                                ;               .
                                ;               .
                                ;               APR7    "           7600
                                ;
                                ;               MM IS ENABLED
                                ;
                                ;
                                M..MMM:
                                        PUSH    R0,R1,R2        ;SAVE
                                        MOV     #77406,R0       ;GET PAGE DESCRIPTOR REGISTER SETUP
                                        MOV     #8.,R1          ;SETUP FOR 8 PDR'S
                                        MOV     #KISDR0,R2      ;GET KISDR0 ADDRESS
                                10$:    MOV     R0,(R2)+        ;SETUP NEXT PDR
                                        DEC     R1              ;COUNT SETUPS
                                        BNE     10$             ;NE IF MORE TO DO
                                        MOV     #KISAR0,R2      ;GET KISAR BASE ADDRESS
                                        CLR     (R2)+           ;SETUP KISAR 0
                                        MOV     #200,(R2)+      ;SETUP KISAR 1
                                        MOV     #400,(R2)+      ;SETUP KISAR 2
                                        MOV     #600,(R2)+      ;SETUP KISAR 3
                                        MOV     #1000,(R2)+     ;SETUP KISAR 4
                                        MOV     #1200,(R2)+     ;SETUP KISAR 5
                                        MOV     #1400,(R2)+     ;SETUP KISAR 6
                                        MOV     #7600,@R2       ;SETUP KISAR 7 TO I/O PAGE
                                        BIS     #BIT0,SR0       ;ENABLE MEMORY MANAGEMENT
                                        POP     R2,R1,R0        ;RESTORE REGS.
                                        RETURN                  ;BACK TO CALLER
                                        .SBTTL  $..MAP MAPPING ROUTINE
```

```
                                ;
                                ; SUBROUTINE:   $..MAP
                                ; ****************
                                ;
                                ;
                                ;       THIS ROUTINE IS USED TO MAP WITH APR6 TO THE PHYSICAL MEMORY
                                ;       ABOVE 28. K AND COMPARE DMA READ AND WRITE DATA
                                ;
                                ;       INPUT:  (SP)    =RETURN ADDRESS
                                ;              2(SP)    =100 OCTAL BYTE BLOCK OFFSET
                                ;              4(SP)    =MAPPING POINTER
                                ;
                                ;       OUTPUT: KERNEL APR6 IS MAPPED TO BUFFER START ADDRESS
                                ;               MAPPING POINTER IS SET TO BASE ADDRESS OF APR6*140000
                                ;               CARRY IS SET IF NO MEMORY AVAILABLE
                                ;               BREAKPOINT TRAP, IF YOU TRY TO MAP TO CDM (0-4.K)
                                ;
                                ;       AUTHOR: ALBERT BREM CSS MUC 16-JAN-79
                                ;
                                ;
002116                          $..MAP: PUSH    R0,R1           ;SAVE REGISTERS
002122                                  IF 6(SP) GE #200 GOTO 10$;DONT MAP TO SYSTEM AREA
002136  000003                         BPT                     ;DONT MAP TO SYSTEM AREA
002140  000137  002316                  JMP     2$              ;LEAVE
002144  004737  002006          10$:    CALL    M..MMM          ;ENABLE MM,SETUP DEFAULT MAPPING
002150  013700  007022                  MOV     MEMEND,R0       ;SAVE HIGHEST MEMORY LOCATION
        000006                          .REPT   6
                                        ROR     R0              ;COMPUTE 100 OCTAL BYTE BLOCKS
                                        .ENDR
002170  042700  176000                  BIC     #176000,R0      ;"
002174  013701  007024                  MOV     MEMEND+2,R1     ;SAVE MEMORY EXTENSION
        000012                          .REPT   10.
                                        ASL     R1              ;COMPUTE ADDITIONAL BLOCKS
                                        .ENDR
002224  060001                          ADD     R0,R1           ;TOTAL BLOCKS OF 32.WORDS MEMORY
002226  042737  000001  177776          BIC     #BIT0,@#PS      ;CLEAR CARRY
002234  020166  000006                  CMP     R1,6(SP)        ;IS MEMORY AVAILABLE?
002240  002023                          BGE     1$              ;YES
002242                                  PREAD   R0              ;SAVE CURRENT PSW
002262  052700  000001                  BIS     #BIT0,R0        ;SET CARRY FOR ERROR INDICATION
002266                                  PSET    R0              ;"
002306  000403                          BR      2$              ;DONT TOUCH APR6
002310  016637  000006  172354  1$:     MOV     6(SP),@#SAR6    ;MAP TO BUFFER AREA
002316                          2$:     POP     R1,R0           ;RESTORE REGISTERS
002322  012616                          MOV     (SP)+,(SP)      ;CLEAN STACK OF ARGUMENT
002324  000207                          RETURN                  ;BACK TO CALLER
                                ;
                                ;
                                ;
```

```
                        .SBTTL  $..EMT
                ;
                ;  SUBROUTINE:  $..EMT
                ;  ***********  ******
                ;
                ;THIS ROUTINE HANDLES ALL EMT-TRAPS THAT MIGHT OCCUR
                ;IN A DIAGNOSTIC PROGRAM, AND DIRECTS THE PROGRAM TO
                ;CONTINUE AT THE APPROPRIATE FUNCTION SUBROUTINES.
                ;
                ;INPUT:         2(SP) = PS
                ;               (SP) = PC
                ;OUTPUT:        NONE
                ;
                ;AUTHOR:        BERT HUBER CSS/DP MUNICH 14-JULI-76
                ;
                ;
                        .ENABL LSB
002326  012746  177776  $..EMT: MOV     #-2,-(SP)       ;REBUILD EMT-ADDR.
002332  066616  000002          ADD     2(SP),(SP)      ;       "
002336  017616  000000          MOV     @(SP),(SP)      ;GET EMT-CODE
002342  042716  177400          BIC     #177400,(SP)    ;EXTRACT EMT-NR.
002346  006316                  ASL     (SP)            ;CONVERT TO TABLE-OFFSET
002350  021627  000004          CMP     (SP),#EMTTND-EMTTBL-2;VALID EMT-NR.?
002354  101413                  BLOS    1$              ;IF LOS, YES
002356                          WRITE MILEMT            ;ILLEGAL EMT
002366                          DUMP    OCT,2(SP)       ;
002400  000137  001000          JMP     @#1000          ;RESTART PROGRAM
002404  062716  002416  1$:     ADD     #EMTTBL,(SP)    ;GET FUCTION CODE ADDR.
002410  017616  000000          MOV     @(SP),(SP)      ;       "
002414  000136                  JMP     @(SP)+          ;ENTER FUCTION SUBROUTINE
002416  002424  EMTTBL: EMT0
002420  002432          EMT1
002422  002470          EMT2
002424          EMTTND
                ;
                ;SUBROUTINE TO SAVE SCOPE-LOOP ADDR.
                ;
002424  011637  007172  EMT0:   MOV     (SP),SCOPAD     ;SAVE ADDR. OF SCOPE-LOOP
002430  000002          RTI                             ;RETURN TO CALLER
                ;
                ;SUBROUTINE TO SETUP A VECTOR
                ;
002432  017646  000010  EMT1:   MOV     @10(SP),-(SP) ;GET POINTER TO VECTOR ADDR.
002436  016676  000010  000000  MOV     10(SP),@(SP)    ;SETUP VECTOR
002444  062716  000002          ADD     #2,(SP)         ;POINT TO NEW PS
002450  016636  000006          MOV     6(SP),@(SP)+    ;SETUP NEW PS
002454  012666  000004          MOV     (SP)+,4(SP)     ;CLEAN UP STACK
002460  012666  000004          MOV     (SP)+,4(SP)     ;       "
002464  005726                  TST     (SP)+           ;       "
002466  000002                  RTI                     ;RETURN TO CALLER
                ;
                ; SUBROUTINE TO RESET A VECTOR
                ;
002470  017646  000004  EMT2:   MOV     @4(SP),-(SP)    ;GET POINTER TO VECTOR A
002474  011646                  MOV     (SP),-(SP)      ;DUPLICATE IT
002476  062716  000002          ADD     #2,(SP)         ;BUILD ADDRESS OF VECTOR
002502  012636                  MOV     (SP)+,@(SP)+    ;RESET VECTOR
```

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

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

$..TRP

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

```
                                .SBTTL  $..IOT
                        ;*
                        ; SUBROUTINE:   $..IOT
                        ; ***********   *******
                        ;
                        ; THIS SUBROUTINE HANDLES ILLEGAL TRAPS OR INTERRUPTS.
                        ;
                        ; IF SUCH A TRAP OR INTERRUPT OCCURS, A MESSAGE IS
                        ; PRINTED, SHOWING THE VECTOR TRAPPED TO, AND THE PROGRAMS PC
                        ; WHERE IT WAS INTERRUPTED.
                        ; THE PROGRAM HALTS THEN, AND MUST BE RESTARTED.
                        ;
                        ; INPUT:       6(SP)   =       PS OF PROGRAM
                        ;              4(SP)   =       PC OF PROGRAM
                        ;              2(SP)   =       PS OF VECTOR
                        ;               (SP)   =       PC OF VECTOR +2
                        ;
                        ; OUTPUT:      MESSAGE TO THE OPERATOR
                        ;
                        ; AUTHOR:      BERT HUBER, CSS/DP MUNICH   22-JUL-76
                        ;-
                                ;.PSECT CSSMON
                                .ENABL  LSB
003056                  $..IOT: WRITE   MILTR1          ;SEND FIRST MESSAGE PART
003066  162716  000002          SUB     #2,(SP)         ;COMPUTE VECTOR
003072                          DUMP    OCT,(SP)        ;DUMP VECTOR ADDRESS
003102                          WRITE   MILTR2          ;SEND SECOND MESSAGE PART
003112                          DUMP    OCT,4(SP)       ;DUMP PROGRAM PC
003124  000777                  BR      .               ;HANG UP SYSTEM
                                .DSABL  LSB
```

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

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

```
                                        .SBTTL  $..RRS
                                ;+
                                ; SUBROUTINE:   $..RRS
                                ; ***********   ******
                                ;
                                ; THIS SUBROUTINE RESTORES R5-R0 FROM THE STACK
                                ;
                                ; INPUT:          (SP)  =       CALLER PC
                                ;                2(SP)  =       SAVED R5
                                ;                4(SP)  =       SAVED R4
                                ;                6(SP)  =       SAVED R3
                                ;               10(SP)  =       SAVED R2
                                ;               12(SP)  =       SAVED R1
                                ;               14(SP)  =       SAVED R0
                                ;               16(SP)  =       OLD CALLER PC
                                ;
                                ; OUTPUT:       REGISTERS ARE RESTORED AND STACK IS CLEANED
                                ;
                                ; AUTHOR:       BERT HUBER, CSS/DP MUNICH   12-APR-76
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
003400  005726                  $..RRS: TST     (SP)+           ;POINT TO SAVED R5
003402                                  POP     R5,R4,R3,R2,R1,R0
003416  005726                          TST     (SP)+           ;SKIP OLD CALLER PC
003420  000176  177760                  JMP     @-20(SP)        ;RETURN TO CALLER
                                        .DSABL  LSB
```

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

$..INP

```
                        .SBTTL  $..1NP
                ;+
                ;SUBROUTINE:    $..INP
                ;***********    ******
                ;
                ;THIS SUBROUTINE IS THE INPUT-HANDLER FOR ALL TERMINALS
                ;USED IN ALL CSS-MUNICH DIAGNOSTIC SOFTWARE GENERATED NOW
                ;AND LATER.
                ;THE SPECIAL FUCTIONS HANDLED BY THIS ROUTINE ARE:
                ;       <CNTRL U> FOR LINE RUBOUT
                ;       <DELETE> FOR CHARACTER RUBOUT
                ;ALL OTHER INPUT IS HANDLED AS NORMAL ASCI-TEXT AND STORED
                ;IN THE KEYBOARD INPUT BUFFER.
                ;
                ;INPUT:         (SP)    -       CALLER PC
                ;               R0      -       CURRENT INPUT CHARACTER
                ;
                ;OUTPUT:        NONE
                ;
                ;AUTHOR:        BERT HUBER CSS/DP MUNICH
                ;-
                        .IDENT  /V1.0/
                    ;   .PSECT  CSSMON
                        .ENABL  LSB
003574  013701  007146  $..INP: MOV     KBBPNT,R1               ;;;GET CURRENT BUFFER-POINTER
003600  012702  006122          MOV     #$..KBO,R2      ;;;POINT TO KEYBOARD-ECHO
003604  022701  007146          CMP     #KBBEND,R1      ;;;INPUT-BUFFER FULL?
003610  001015                  BNE     1$              ;;;IF NE, NO
003612  022700  000025          CMP     #25,R0          ;;;<CNTRL U>?
003616  001415                  BEQ     2$              ;;;IF EQ, YES.
003620  122700  000177          CMPB    #177,R0         ;;;<DELETE>?
003624  001407                  BEQ     1$              ;;;IF EQ, YES.
003626  005037  007150          CLR     INFLAG          ;;;RESET INPUTFLAG
003632                          WRITE   #KBOVF          ;;;PRINT OVERFLOW MESS.
003642  000451                  BR      7$              ;;;PROMPT FOR INPUT
003644  122700  000025  1$:     CMPB    #25,R0          ;;;<CNTL U>?
003650  001436                  BEQ     6$              ;;;IF NE, NO
003652  122700  000177  2$:     CMPB    #177,R0         ;;;<DELETE>?
003656  001016                  BNE     4$              ;;;IF NE, NO
003660  005737  007162          TST     RUBFLG          ;;;FIRST RUBOUT?
003664  001005                  BNE     3$              ;;;IF NE, NO
003666  010637  007162          MOV     SP,RUBFLG       ;;;FLAG RUBOUT MODE
003672  012700  000057          MOV     #'/,R0          ;;;PRINT "/"
003676  004712                  CALL    @R2             ;;;      "
003700  020127  007026  3$:     CMP     R1,#KBBUFF      ;;;FULL LINE DELETED?
003704  001434                  BEQ     INP..E          ;;;IF EQ, YES
003706  114100                  MOVB    (R1),R0                 ;;;NO, REMOVE CHARACTER
003710  004712                  CALL    @R2             ;;;AND ECHO IT
003712  000431                  BR      INP..E          ;;;WAIT FOR NEXT INPUT
003714  005737  007162  4$:     TST     RUBFLG          ;;;RUBOUT MODE?
003720  001407                  BEQ     5$              ;;;IF EQ, NO
003722                          PUSH    R0              ;;;YES, SAVE CURRENT INPUT
003724  012700  000057          MOV     #'/,R0          ;;;NO OF RUBOUT
003730  004712                  CALL    @R2             ;;;ECHO "/"
003732  005037  007162          CLR     RUBFLG          ;;;RESET RUBOUT MODE FLAG
003736                          POP     R0              ;;;GET CURRENT INPUT AGAIN
003740  004712  5$:     CALL    @R2             ;;;ECHO IT
```

```
$..INP

     647 003742  110021                      MOVB    R0,(R1)+        ;;;SAVE IN IPUT BUFFER
     648 003744  000414                      BR      INP..E          ;;;WAIT FOR NEXT INPUT
     649 003746  012700  000136      6$:     MOV     #136,R0         ;;;ECHO "^"
     650 003752  004712                      CALL    @R2             ;;;
     651 003754  012700  000125              MOV     #'U,R0          ;;;ECHO "U"
     652 003760  004712                      CALL    @R2             ;;;     "
     653 003762  005037  007162              CLR     RUBFLG          ;;;LEAVE RUBOUT MODE
     654 003766  012701  007026      7$:     MOV     #KBBUFF,R1      ;;;RESET INPUT BUFFER POINTER
     655 003772  004737  006136              CALL    $..PRO          ;;;PROMPT FOR INPUT
     656 003776  010137  007146      INP..E: MOV     R1,KBBPNT       ;;;SAVE POINTER
     657 004002  116100  177777              MOVB    -1(R1),R0       ;;;SAVE LAST CHARACTER
     658 004006  000207                      RETURN                  ;;;RETURN TO CALLER
     659                                     .DSABL  LSB
```

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

```
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                        .SBTTL  $..IAA
                                ;+
                                ;SUBROUTINE:    $..IAA
                                ;***********    ******
                                ;
                                ;LEGAL INPUT IS "ABO"
                                ;
                                ;       INPUT:  NONE
                                ;       OUTPUT: IF IN THE PROGRM THE LABEL ""$..SPC"" IS USED,
                                ;               THE STACK WILL BE INITIALIZED AND A JUMP TO
                                ;               THIS LABEL IN THE DEVICE TEST PROGRAM IS PERFORMED.
                                ;               IF THIS LAREL IS NOT USED, THE STACK IS INITIALIZED
                                ;               AND A JUMP TO THE BEGIN OF THE DEVICE-TEST PROGRAM
                                ;               IS PERFORMED.
                                ;       MZ001
                                ;       *****
                                ;       THE MACRO "PSET"ALLOWS TO RUN THE CDM EITHER ON PDP
                                ;       OR ON LSI.
                                ;
                                ;
                                ;AUTHOR:        M. ZILLER,      CSS/DP MUNICH   OCT. 1976
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
004120  000005                  $..IAA: RESET                   ;STOP ANY ACTIONS
004122                                  SET6    @#TKS           ;SWITCH ON TTY-INT.
004130  012706  001204                  MOV     #START,SP       ;INITIALIZE STACK
                                        .IF DF $..SPC
004134  012746  012010                  MOV     #$..SPC,-(SP)   ;SET THE LABEL ""$..SPC"" FOR DESTINATION
                                        .ENDC
                                        .IF NDF $..SPC
                                        MOV     #CDM..E,-(SP)   ;SET END OF MONITOR FOR DESTINATION
                                        .ENDC
004140                                  PSET    #0              ;CLEAR PS       ;MZ001
004164  012700  007026                  MOV     #KBBUFF,R0      ;RESET BUFFERS
004170  005020                  1$:     CLR     (R0)+           ;       "
004172  020027  007202                  CMP     R0,#PROMES      ;       "
004176  001374                          BNE     1$              ;       "
004200  012737  007026  007146          MOV     #KBBUFF,KBBPNT  ;REBUILD BEGIN ADDRESS
004206  000207                          RETURN                  ;GOTO POINT OF CONTINUATION
                                        .DSABL  LSB
```

```
                                        ;.PSECT CSSMON
                                ;
                                ;
                                ;
                                ;
                                        .SBTTL  $..IAD
                                ;
                                ;SUBROUTINE     :       $..IAD
                                ;**********             ******
                                ;
                                ;       MZ001
                                ;       *****
                                ;       MODIFICATION TO ALLOW THE <CR> AFTER INPUT
                                ;       TO COMPLETE.
                                ;
                                ;
                                ;       THIS SUBROUTINE IS ENTERED WHEN A DUMP-REQUEST IS ANALYZED.
                                ;       DEPENDING ON HOW THE REQUEST IS TERMINATED, THERE ARE TWO
                                ;       WAYS FOR RETURN:
                                ;
                                ;       A)      WHEN TERMINATED BY <CR> THE CONTENT OF A LOCATION IS
                                ;               PRINTED AND THE RETURN IS PERFORMED
                                ;       B)      WHEN INPUT IS TERMINATED BY A ""/""THE MONITOR WAITS
                                ;               FOR INPUT AND LOADS THIS INPUT INTO THE EXAMINED LOCATION.
                                ;
                                ;INPUT  :       (SP)    =RETURN ADDRESS
                                ;OUTPUT :       MODIFIED LOCATION IF REQUESTED
                                ;
                                ;AUTHOR:        M. ZILLER, SS/DP MUNICH         OCTOBER 1976
                                ;
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
004210  005201                  $..IAD: INC     R1              ;POINT TO "M"
004212  122127  000115                  CMPB    (R1)+,#'M       ;IS IT A "M"
004216  001027                          BNE     77$             ;NO IF NE
004220  122127  000120                  CMPB    (R1)+,#'P       ;IS IT A "P"
004224  001024                          BNE     77$             ;NO IF NE
004226  122127  000057                  CMPB    (R1)+,#'/       ;IS IT A "/"
004232  001021                          BNE     77$             ;NO IF NE
004234  004737  004304                  CALL    $..NOR          ;RETRIEFE INPUT
004240  103416                          BCS     77$             ;IS CS, INPUT FORMAT ERROR
004242  105737  177564          11$:    TSTB    @#TPS           ;IS THE <CR> FINISHED ? ;MZ001
004246  100375                          BPL     11$             ;                       ;MZ001
004250  012737  000012  177566          MOV     #12,@#TPB       ;PRINT A <LF
004256  105737  177564          12$:    TSTB    @#TPS           ;DONE ?
004262  100375                          BPL     12$             ;WAIT FOR COMLETION
004264                                  DUMP    OCT,(R0)                ;AND PRINT
004274  000402                          BR      IAD..E          ;GO TO END
004276  004737  006262          77$:    CALL    $..STX          ;WRITE SYNTAX ERROR MESSAGE
004302  000207                  IAD..E: RETURN                  ;RETURN TO MAIN
                                ;
```

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

H6


```
    864 004460  000402                      BR      IAS..E          ;GO TO END
    865 004462  004737  006262      76$:    CALL    $..STX          ;WRITE  ERROR MESSAGE
    866 004466  000207              IAS..E: RETURN                  ;RETURN
    867 004470  000000              2$:     .WORD   0               ;SORAGE FOR ADDRESS TO MODIFY
    868                                     .DSABL  ! 58
```

```
                        ;
                        ;
                        ;
                                .SBTTL  $..IAE
                        ;+
                        ;SUBROUTINE:    $..IAE
                        ;***********    ******
                        ;
                        ;THIS ROUTINE CHECKS THE CORRECT SYNTAX OF
                        ;AN"ERR"- INPUT STRING AND PROCESSES THE
                        ;REQUESTED 'SWITCHREGISTER-OPTIONS'.
                        ;
                        ;LEGAL SWITCH OPTIONS ARE:
                        ;
                        ;       /HT     -       HALT AFTER ERROR (DEFAULT)
                        ;       /NH     -       NO HALT AFTER ERROR
                        ;       /PR     -       DRINT ERROR MESSAGE (DEFAULT)
                        ;       /NP     -       INHIBIT ERROR PRINTOUT
                        ;       /CT     -       COUNT ERROR
                        ;       /NC     -       NO ERROR COUNTING (DEFAULT)
                        ;       /SC     -       SCOPE LOOP ON THIS ERROR
                        ;       /NS     -       NO SCOPE LOOP (DEFAULT)
                        ;       /CONT   -       IMMEDIATE CONTINUE SWITCH
                        ;
                        ;
                        ;INPUT:         R1      -START ADDR.OF TEXT STRING
                        ;OUTPUT:        BIT SETUP IN SOFTWARE-SWITCH-REGISTER
                        ;
                        ;AUTHOR:        BERT HUBER, CSS/DP MUNICH  28-JULY-76
                        ;-
                                ;.PSECT CSSMON
                                .ENABL  LSB
004472  010602          $..IAE: MOV     SP,R2           ;;;FLAG FOR VERY FIRST CHARACTER
004474  012703  007160          MOV     @PSDSWR,R3      ;;;GET 'SWR" ADDRESS
004500  005004                  CLR     R4              ;;;USE AS 'CHARACTER 1'
004502  062701  000003          ADD     #3,R1           ;;;SKIP IDENTIFYER
004506  122127  000075          CMPB    (R1)+,#'-       ;;;CORRECT SYNTAX?
004512  001137                  BNE     77$             ;;;BRANCH IF NOT
004514  122127  000057  1$:     CMPB    (R1)+,#'/       ;;;IS NEXT CHAR. A '/'
004520  001403                  BEQ     2$              ;;;IF EQ, YES
004522  005702                  TST     R2              ;;;VERY FIRST?
004524  001132                  BNE     77$             ;;;IF YES: SYNTAX ERROR
004526  000526                  BR      11$             ;;;ASSUME END OF STRING
004530  005002          2$:     CLR     R2              ;;;NO LONGER VERY FIRST
004532  005204                  INC     R4              ;;;SET FOR 'C1' FLAG
004534  121127  000116          CMPB    (R1),#'N        ;;;IS IT THE '/N" OPTION?
004540  001002                  BNE     3$              ;;;BRANCH IF NOT
004542  005201                  INC     R1              ;;;ADVANCE TO NEXT CHARACTER
004544  005004                  CLR     R4              ;;;RESET 'C1' FLAG
004546  121127  000123  3$:     CMPB    (R1),#'S        ;;;IS IT A 'S'?
004552  001426                  BEQ     5$              ;;;BRANCH IF YES
004554  121127  000103          CMPB    (R1),#'C        ;;;IS IT A 'C' ?
004560  001437                  BEQ     7$              ;;;     BRANCH IF YES
004562  121127  000110          CMPB    (R1),#'H        ;;;IS IT A'H' ?
004566  001472                  BEQ     9$              ;;; BRANCH IF YES
004570  121127  000120          CMPB    (R1),#'P        ;;;IS IT A 'P' ?
004574  001106                  BNE     77$             ;;;IF NO: SYNTAX ERROR
```

```
927 004576  005704                        TST     R4              ;;;IS IT A 'P' ?
928 004600  001004                        BNE     4$              ;;;BRANCH IF YES
929 004602  052713  040000                BIS     #BIT14,(R3)     ;;;SET SR-BIT FOR /NP
930 004606  005201                30$:    INC     R1              ;;;ADVANCE TO NEXT CHARACTER
931 004610  000741                        BR      1$              ;;;CHECK FOR NEXT SWITCH
932 004612  005201                4$:     INC     R1              ;;;NEXT CHARACTER
933 004614  122127  000122                CMPB    (R1)+,#'R       ;;;IS IT A '/PR' ?
934 004620  001074                        BNE     77$             ;;;IF NO: SYSNTAX ERROR
935 004622  042713  040000                BIC     #BIT14,(R3)     ;;;SET '/PR'-OPTION
936 004626  000732                        BR      1$              ;;;CHECK NEXT SWITCH
937 004630  005704                5$:     TST     R4              ;;;IS IT 'S/' ?
938 004632  001003                        BNE     6$              ;;;BRANCH IF YES
939 004634  042713  100000                BIC     #BIT15,(R3)     ;;;SETUP FOR '/NS' OPTION
940 004640  000762                        BR      30$             ;;;TRY NEXT
941 004642  005201                6$:     INC     R1              ;;;NEXT CHARACTER
942 004644  122127  000103                CMPB    (R1)+,#'C       ;;;IS IT A'/SC' ?
943 004650  001060                        BNE     77$             ;;;IF NO: SYNTAX ERROR
944 004652  052713  100000                BIS     #BIT15,(R3)     ;;;SETUP FOR '/SC'-OPTION
945 004656  000716                        BR      1$              ;;;CHEC( NEXT SWITCH
946 004660  126127  000001  000117  7$:   CMPB    1(R1),#'O       ;;;CHECK FOR '/CONT'
947 004666  001016                        BNE     70$             ;;;IF NE , OTHER
948 004670  126127  000002  000116        CMPB    2(R1),#'N       ;;;CHECK FOR ;/CONT'
949 004676  001045                        BNE     77$             ;;;IF NO: SYNTAX ERROR
950 004700  126127  000003  000124        CMPB    3(R1),#'T       ;;;CHECK FOR '/CONT'
951 004706  001041                        BNE     77$             ;;;IF NO: SYSNTAX ERROR
952 004710  052737  004000  007160        BIS     #BIT11,PSCSWR   ;;;SET 'IMMEDIATE CONTINUE'
953 004716  062701  000005                ADD     #5,R1           ;;;ADJUST POINTER
954 004722  000430                        BR      11$             ;;;END THIS SCAN
955 004724  005704                70$:    TST     R4              ;;;IS IT A '/CT' ?
956 004726  001003                        BNE     8$              ;;;BRANCH IF YES
957 004730  042713  020000                BIC     #BIT13,(R3)     ;;;SETUP FOR '/NC'-OPTION
958 004734  000724                        BR      30$             ;;;TRY NEXT SWITCH
959 004736  005201                8$:     INC     R1              ;;;ADVANCE TO NEXT CHARACTER
960 004740  122127  000124                CMPB    (R1)+,#'T       ;;;IS IT A'/CT' OPTION ?
961 004744  001022                        BNE     77$             ;;;IF NO: SYNTAX ERROR
962 004746  052713  020000                BIS     #BIT13,(R3)     ;;;SETUP FOR '/CT' OPTION
963 004752  000660                        BR      1$              ;;;CHECK NEXT SWITCH
964 004754  005704                9$:     TST     R4              ;;;IS IT '/HT' ?
965 004756  001003                        BNE     10$             ;;;BRANCH IF YES
966 004760  052713  010000                BIS     #BIT12,(R3)     ;;;SETUP FOR '/NP'-OPTION
967 004764  000710                        BR      30$             ;;;TRY NEXT SWITCH
968 004766  005201                10$:    INC     R1              ;;;ADVANCE TO NEXT CHARACTER
969 004770  122127  000124                CMPB    (R1)+,#'T       ;;;IS IT A '/HT' ?
970 004774  001006                        BNE     77$             ;;;IF NO: SYNTAX ERROR
971 004776  042713  010000                BIC     #BIT12,(R3)     ;;;SETUP FOR '/HT'-OPTION
972 005002  000644                        BR      1$              ;;;TRY NEXT SWITCH
973 005004  124127  000015        11$:    CMPB     (R1),#15       ;;;WAS LAST BYTE A <CR>?
974 005010  001402                        BEQ     IAE..E          ;;;BRANCH IF YES
975 005012  004737  006262        77$:    CALL    $..STX          ;;;SYNTAX ERROR
976 005016  000207                IAE..E: RETURN                  ;;;RETURN TO CALLER
977                                       .DSABL LSB
```

```
                        .SBTTL  $..RED
                ;+
                ;SUBROUTINE     $..RED
                ;**********     **.***
                ;
                ;
                ;**** ABOO
                ;       THE CARRY BIT IS SET IF A ZERO OCTAL INPUT (ONLY CR,NO DATA)
                ;       HAPPENS
                ;
                ;
                ;THIS SUBROUTINE HANDLES ALL TYPES OF SPECIAL INPUT REQUESTS
                ;SETUP BY THE PROGRAMMER THROUGH A ' READ ' MACRO.
                ;LEGAL INPUT TYPES THAT CAN BE HANDLED ARE:
                ;
                ;BINARY FORMAT (EX: 1001101101010001)
                ;-----------------------------------
                ;INPUT CAN BE UP TO 16-DIGITS OF THE FORM 0 OR 1.
                ;LESS THAN 16 DIGIT INPUT IS FILLED UWITH LEADING ZEROES
                ;INPUT IS RETURNED IN R0.
                ;THE PREVIOUS CONTEND OF R0 WILL BE OVERWRITTEN.
                ;
                ;OCTAL FORMAT (EX:176523)
                ;-----------------------
                ;INPUT CAN BE UP TO 6 DIGITS OF THE FORM 0-7.
                ;IN CASE OF A 6-DIGIT-WIDE INPUT,THE LEFTMOST(HIGH ORDER)
                ;DIGIT IS TRUCTATED TO THE RIGHTMOST (LOW ORDER)BIT.
                ;IN CASE OF LESS THAN 6 DIGITS INPUT, LEADING ZERO DIGITS
                ;ARE INSERTED.
                ;THE RESULT IS RETURNED IN R0,HEREBY OVERWRITING
                ;THE PREVIOUS CONTENT OF R0.
                ;
                ;DECIMAL FORMAT (EX: 96003)
                ;-------------------------
                ;INPUT CAN BE UP TO 5 DIGITS OF THE FORM 0-9.
                ;LESS THEN FIVE DIGITS INPUT ARE FILLED UP WITH
                ;LEADING ZEROE DIGITS.
                ;MAXIMUM NUMBER ALLOWED IS 32656.
                ;INPUT IS RETURNED IN R0, HEREBY OVERWRITING THE
                ;PREVIOUS CONTENT OF R0.
                ;
                ;ASCI FORMAT (EX: B6T*&L)
                ;----------------------- --
                ;
                ;INPUT CAN BE ANY LEGAL ASCII CHARACTER.
                ;IF INPUT INTO A BUFFER IS REQUESTED, UP TO 80
                ;CHARACTERS CAN BE INPUT.
                ;IF INPUT RETURN IN R0 IS REQUESTED,ONLY THE FIRST
                ;CHARACTER WILL BE RETURNED IN THE LOW ORDER BYTE
                ;OF R0.
                ;OLD CONTEND OF R0 IS OVERWRITTEN IN THIS CASE.
                ;IF THE ONLY INPUT IS <CR>, ZERO WILL BE RETURNED IN R0
                ;AS WELL AS IN THE FIRST BYTE OF THE SPECIFIED BUFFER.
                ;
                ;AUTHOR:         BERT HUBER, CSS/DP MUNICH       29-JUN-76
                ;
005020  004737  006136  $..RED: CALL    $..PRO          ;PRINT PROMPT STRING
```

```
$..RED

1036 005024  010637  007156                    MOV     SP,INPREQ       ;SET SPECIAL REQUST FLAG
1037 005030  026627  000002  000004            CMP     2(SP),#4        ;DETECT SPECIAL ASCII INPUT RUNNING
1038 005036  003402                            BLE     100$            ;NO ASCII INPUT
1039 005040  010637  007174                    MOV     SP,$.$ASF       ;SET SPECIAL ASCII INPUT FLAG
1040 005044                          100$:
1041 005044  005737  007150           1$:      TST     INFLAG          ;WAIT FOR INPUT COMPLETE
1042 005050  001375                            BNE     1$              ;
1043 005052                                    PUSH    R1,R2,R3,R4,R5  ;SAVE REGS
1044 005064  012701  007026                    MOV     #KBBUFF,R1      ;GET INPUTPIONTER
1045 005070  016602  000014                    MOV     14(SP),R2       ;GET INPUTMOFE INDICATOR
1046 005074  000172  005100                    JMP     @2$(R2)         ;SELECT REQ. INPUT
1047 005100  005112           2$:              $..OCT                  ;HANDLE OCTAL INPUT
1048 005102  005270                            $..DEC                  ;HANDLE DECIMAL INPUT
1049 005104  005504                            $..BIN                  ;HANDLE BINARY INPUT
1050 005106  005444                            $..BUF                  ;HANDLE ASCII INPUT INTO BUFFER
1051 005110  005464                            $..ASC                  ;HANDLE ASCII INPUT INTO BUFFER
1052 005112  005000           $..OCT:  CLR     R0                      ;CLEAR INPUT POINTER
1053 005114  005037  005244                    CLR     6$              ;RESET COUNTER
1054 005120  121127  000015                    CMPB    @R1,#15         ;ZERO INPUT ?
1055 005124  001014                            BNE     3$              ;IF NE, SKIP
1056 005126                                    POP     R5,R4,R3,R2,R1  ;***ABOO: RESTORE GEN. REGISTERS
1057 005140  011666  000002                    MOV     (SP),2(SP)      ;***ABOO: REAJUST RETURN ADDRESS
1058 005144  005726                            TST     (SP)+           ;***ABOO:
1059 005146  005037  007156                    CLR     INPREQ          ;***ABOO: RESET SPECIAL REQUEST COUNTER
1060 005152  000261                            SEC                     ;***ABOO: SET CARRY AS ZERO INPUT INDICATOR
1061 005154  000207                            RETURN                  ;***ABOO: AND GO BACK TO CALLER
1062 005156  006300           3$:      ASL     R0                      ;ADJUST INPUT CHARACTER
1063 005160  006300                            ASL     R0              ;       "
1064 005162  006300                            ASL     R0              ;       "
1065 005164  023727  005244  000006            CMP     6$,#6           ;ALL INPUT HANDLED ?
1066 005172  001406                            BEQ     4$              ;MORE THAN 6 DIGITS INPUT
1067 005174  121127  000060                    CMPB    @R1,#60         ;LEGAL OCTAL DIGIT ?
1068 005200  103403                            BLO     4$              ;IF LO, NO
1069 005202  121127  000067                    CMPB    @R1,#67         ;LEGAL OCTAL DIGIT ?
1070 005206  101412                            BLOS    5$              ;IF LOS, YES
1071 005210                          4$:       WRITE   MILOCT          ;ILLEGAL OCTAL INPUT
1072 005220                                    POP     R5,R4,R3,R2,R1  ;RESTORE REGS
1073 005232  000672                            BR      $..RED          ;RESTART INPUT SEQUENCE
1074 005234  142711  000060           5$:      BICB    #60,@R1         ;BUILD BINARY
1075 005240  152100                            BISB    (R1)+,R0        ;INSERT INTO BUFFER
1076 005242  005227                            INC     (PC)+           ;COUNT INPUTS HANDLED
1077 005244  000000           6$:      .WORD   0                       ;DIGIT COUNTER
1078 005246  121127  000015                    CMPB    @R1,#15         ;END OF INPUT DETECTED?
1079 005252  001341                            BNE     3$              ;IF NE , NO
1080 005254  023727  005244  000001            CMP     6$,#1           ;SET CARRY IF ONLY CR WAS THE INPUT
1081 005262  001001                            BNE     7$              ;
1082 005264  000261                            SEC                     ;SET CARRY
1083 005266                          7$:
1084 005266  000557                            BR      RED..F          ;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$:     INC     R1              ;POINT TO NEXT INPUT
1150 005606  005237  005624              INC     6$              ;COUNT INPUTS
1151 005612  000742                      BR      1$              ;CHECK NEXT
1152 005614  121127  000015      4$:     CMPB    @R1,#15         ;END OF INPUT ?
1153 005620  001343                      BNE     5$              ;NO, ILLEGAL CHARACTER
1154 005622  000401                      BR      RED..E          ;
1155 005624  000000              6$:     .WORD   0               ;COUNTER
1156 005626                      RED..E: POP     R5,R4,R3,R2,R1  ;REST. REGS
1157 005640  011666  000002              MOV     (SP),2(SP)      ;READJUST STACK RETURN ADDRESS
1158 005644  005726                      TST     (SP)+           ;       "
1159 005646  005037  007156              CLR     INPREQ          ;RESET SPECIAL REQUEST POINTER
1160 005652  000207                      RETURN                  ;RETURN TO CALLER
```

```
$..WRT

                        .SBTTL  $..WRT
                ;+
                ;SUBROUTINE:    $..WRT
                ;***********    ******
                ;
                ;THIS ROUTINE IS CALLED TO SETTUP A MESSAGE PRINTOUT ON CONSOLE
                ;ALL PRINTING IS CONTROLLED BY INTERRUPT EXEPT <CR>, <LF>.
                ;
                ;INPUT:         2(SP)   -       ADDRESS OF MESSAGE TEXT
                ;               (SP)    -       CALLER PC
                ;
                ;OUTPUT:                MESSAGE ON TERMINAL
                ;
                ;AUTHOR:        BERT HUBER CSS/DP MUNICH        28 JUN-76
                ;
                ;MODIFIED BY:   JOHN LEVESQUE   CSS ISG   23-JAN-83
                ;
                ;               MODIFICATION REQUIRED TO HANDLE XON AND XOFF
                ;
                ;-
                        ;.PSECT CSSMON
                        .ENABL  LSB
005654  005737  007152  $..WRT: TST     OUTFLG          ;OUTPUT RUNNING ?
005660  001375                  BNE     $..WRT          ;IF NE, YES - WAIT
005662  005737  007150          TST     INFLAG          ;INPUT RUNNING ?
005666  001372                  BNE     $..WRT          ;IF NE, YES - WAIT
005670  010637  007152          MOV     SP,OUTFLG       ;SET OUTPUTFLAG
005674  016627  000002          MOV     2(SP),(PC)+             ;GET MESSAGE POINTER
005700  000000          $..MSP: .WORD   0               ;MESSAGE POINTER
005702  012737  005736  000064  MOV     #1$,@#64        ;SETUP VECTOR
005710  012737  000340  000066  MOV     #340,@#66       ;;;     "
005716  012616                  MOV     (SP)+,(SP)      ;UPDATE STACK
005720  052737  000100  177564  BIS     #100,@#TPS      ;ENABLE INTERRUPTS
005726  005737  007152          TST     OUTFLG          ;OUTPUT RUNNING?        ;**GS0001**
005732  001375                  BNE     .-4             ;IF NE ,YES->WAIT       ;**GS0001**
005734  000207                  RETURN                  ;RETURN TO CALLER
005736  122737  000023  177562  1$:     CMPB    #23,@#TKB       ;;;  CHECK FOR XOFF
005744  001401                  BEQ     3$              ;;;  XOFF CONDITION WAIT TO CLEAR
005746  000404                  BR      20$             ;;;  NO XOFF THEN CONTINUE
005750  122737  000021  177562  3$:     CMPB    #21,@#TKB       ;;;  CHECK FOR XON
005756  001367                  BNE     1$              ;;;  NO XON GO WAIT FOR IT
005760                  20$:
005760  127727  177714  000135          CMPB    @$..MSP,#'\     ;END OF MESSAGE ?
005766  001434                  BEQ     WRT..E          ;;;IF EQ, YES
005770  127727  177704  000133          CMPB    @$..MSP,#'[     ;;;<CR>,<LF>?
005776  001022                  BNE     30$             ;;;IF NE, NO
006000  042737  000100  177564          BIC     #100,@#TPS      ;;;DISABLE INTERRUPTS
006006  012737  000015  177566          MOV     #15,@#TPB       ;;;SEND <CR>
006014  105737  177564                  TSTB    @#TPS           ;;;WAIT FOR DONE
006020  100375                  BPL     .-4             ;;;
006022  005237  005700                  INC     $..MSP          ;;;POINT TO NEXT CHARACTER
006026  012737  000012  177566          MOV     #12,@#TPB       ;;;SEND <LF>
006034  052737  000100  177564          BIS     #100,@#TPS      ;;;RE ENABLE INTERRUPT
006042  000002                  RTI                     ;;;RETURN TO MAIN PROGRAM
006044  117737  177650  177566  30$:    MOVB    @$..MSP,@#TPB   ;;;SEND NEXT CHARACTER
006052  005237  005700                  INC     $..MSP          ;;;POINT TO NEXT
006056  000002                  RTI                     ;;;RETURN TO CALLER
```

```
IEC11-B TEST    MACRO M1200  30-MAR-84 16:11  PAGE 20-1                                    SEQ 80
$..WRT

 1219 006060  042737  000100  177564  WRT..E: BIC     #100,@#TPS      ;;;DISABLE OUTPUT INTERRUPT
 1220 006066  005037  007152                  CLR     OUTFLG          ;;;RESET OUTPUT FLAG
 1221 006072  005737  007170                  TST     SPEFLG          ;;;THIS FLAG IS SET ONLY
 1222 006076  001410                          BEQ     4$              ;;;WHEN DURING A PRINTOUT
 1223 006100  005037  007170                  CLR     SPEFLG          ;;;THE <CNTRL C> COMMAND WAS
 1224 006104  005037  007156                  CLR     INPREQ          ;;;NECCESARY TO INTERRUPT THE
 1225 006110  004737  003360                  CALL    $..RSV          ;;;RUNNING PRINTOUT
 1226 006114  000137  003520                  JMP     $$SPE           ;;;NOW NEW SPEC. INPUT IS PROCESSED
 1227 006120  000002                  4$:     RTI                     ;;;RETURN TO MAIN PROGRAM
 1228                                         .DSABL LSB
```

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

```
                                .SBTTL  $..PRO
                              ;*
                              ;SUBROUTINE:    $..PRO
                              ;***********    ******
                              ;
                              ;THIS ROUTINE IS CALLED TO PRINT THE DEFAULT KEYBOARD
                              ;PROMPT STRING ON THE TERMINAL, AND TO SET THE INPUT FLAG.
                              ;
                              ;INPUT:         (SP)    *       CALLER PC
                              ;
                              ;OUTPUT:        PROMPT STING ON TERMINAL
                              ;
                              ;AUTHOR:        BERT HUBER CSS/DP MUNICH        28-JUN-76
                              ;       MZ001
                              ;       *****
                              ;       THE USE OF THIS MACRO'S ALLOWS TO USE ONE VERSION OF CDM
                              ;       ON LSI AND PDP.
                              ;
                              ;-
                                .IDENT  /V1.0/
                                ;.PSECT CSSMON
                                .ENABL  LSB
006136  005037  007150        $..PRO: CLR     INFLAG          ;;;RESET INPUT FLAG
006142                                PREAD   STATUS  ;SAVE STATUS    ;MZ001
006166                                PSET    #0      ;CLEAR PS       ;MZ001
006212                                WRITE   PROMES          ;;;WRITE PROMPT STRING
006222  005737  007152        1$:     TST     OUTFLG          ;WAIT FOR OUTPUT COMPLETE
006226  001375                        BNE     1$              ;
006230                                PSET    STATUS          ;RELOAD STATUS  ;MZ001
006254  010637  007150                MOV     SP,INFLAG               ;;;SET INPUT FLAG
006260  000207                        RETURN                  ;RETURN TO CALLING PROGRAM
                                .DSABL  LSB
```

```
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 COM
1302                                ;       ON LSI AND PDP.
1303                                ;
1304                                ;-
1305                                        ;.PSECT CSSMON
1306                                        .ENABL  LSB
1307 006262  000240                $..STX: NOP                             ;;;DUMMY
1308 006264                                PREAD   STATUS  ;SAVE STATUS    ;MZ001
1309 006310                                PSET    #0      ;CLEAR PS       ;MZ001
1310 006334                                WRITE   MSYERR                  ;WRITE SYNTAX ERROR MESSAGE
1311 006344  005737  007152          1$:     TST     OUTFLG  ;WAIT FOR OUTPUT COMPLETE
1312 006350  001375                        BNE     1$                      ;
1313 006352                                PSET    STATUS                  ;RELOAD PS      ;MZ001
1314 006376  000207                        RETURN                          ;;;RETURN TO CALLER
1315                                        .DSABL  LSB
```

```
                        .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                                    ;INPUT:         4(SP)   =       VALUE TO DUMP
1333                                    ;               2(SP)   =TYPE OF DUMP   :  0 = OCTAL
1334                                    ;                                          2 = DECIMAL
1335                                    ;                                          4 = BINARY
1336                                    ;               (SP)    =RETURN PC
1337                                    ;
1338                                    ;OUTPUT:        DUMP ON TERMINAL
1339                                    ;
1340                                    ;AUTHOR:        BERT HUBER CSS/DP MUNICH        28-JUN-76
1341                                    ;-
1342                                            ;.PSECT CSSMON
1343                                            .ENABL  LSB
1344 006400                            $..DMP: PUSH    R0              ;SAVE R0
1345 006402  016600  000004                    MOV     4(SP),R0                ;GET TYPE ARGUMENT
1346 006406  005737  007152                    TST     OUTFLG          ;OUTPUT RUNNING ?
1347 006412  001375                            BNE     .-4             ;IF NE, WAIT
1348 006414  000170  006420                    JMP     @D..TYP(R0)     ;SELECT FUNCTION ROUTINE
1349 006420  006426                    D..TYP: $..DOC
1350 006422  006520                            $..DDC
1351 006424  006630                            $..DBN
1352 006426  012746  000006            $..DOC: MOV     #6,-(SP)        ;SETUP DIGIT COUNT
1353 006432  016646  000010                    MOV     10(SP),-(SP)    ;GET VALUE TO DUMP
1354 006436  012700  000060                    MOV     #'0,R0          ;SETUP BASIC ASCII VALUE
1355 006442  006316                            ASL     (SP)            ;BUILD FIRST DIGIT
1356 006444  103001                            BCC     1$              ;IF CC, DIGIT IS A '0'
1357 006446  005200                            INC     R0              ;OTHERWISE DIGIT IS A ?
1358 006450  004737  006122            1$:     CALL    $..KBO          ;DUMP DIGIT
1359 006454  005366  000002                    DEC     2(SP)           ;COUNT DIGITS DUMPED
1360 006460  001415                            BEQ     4$              ;IF EQ, ALL DONE
1361 006462  012700  000060                    MOV     #'0,R0          ;SETUP NEXT DIGIT BASE
1362 006466  006316                            ASL     (SP)            ;BUILD NEXT DIGIT
1363 006470  103002                            BCC     2$              ;IF CC, DIGIT INCLUDES ?????
1364 006472  062700  000004                    ADD     #4,R0           ;DIGIT = BASE+4
1365 006476  006316                    2$:     ASL     (SP)            ;ANALYZE NEXT BIT
1366 006500  103001                            BCC     3$              ;IF CC, DIGIT INCLUDES ????
1367 006502  122020                            CMPB    (R0)+,(R0)+     ;DIGIT = DIGIT + 2
1368 006504  006316                    3$:     ASL     (SP)            ;ANALIZE LAST BIT OF DIGIT
1369 006506  103360                            BCC     1$              ;IF CC, DIGIT MUST BE EVEN
1370 006510  005200                            INC     R0              ;OTHERWISE DIGIT IS ODD
1371 006512  000756                            BR      1$              ;GO AND DUMP
1372 006514  022626                    4$:     CMP     (SP)+,(SP)+     ;READJUST STACK
1373 006516  000466                            BR      DMP..E          ;END OF DUMP
1374                                            .DSABL  LSB
1375                                            .ENABL  LSB
1376 006520  012746  000005            $..DDC: MOV     #5,-(SP)        ;SETUP A DIGIT COUNT
1377 006524  016646  000010                    MOV     10(SP),-(SP)    ;GET VALUE TO DUMP
1378 006530  012700  000060                    MOV     #'0,R0  ;BUILD LEADING ZERO
1379 006534  004737  006122                    CALL    $..KBO          ;AND PRINT
1380 006540  012700  000060            1$:     MOV     #'0,R0  ;SETUP BASIC ASCII VALUE
1381 006544  027716  000044            2$:     CMP     @$POINT,(SP)    ;CMP VALUE WITH ACT. DEC. DIGIT
1382 006550  101004                            BHI     3$              ;VALUE IS SMALLER
1383 006552  005200                            INC     R0              ;COUNT NUMBER OF DIVISIONS
1384 006554  167716  000034                    SUB     @$POINT,(SP)    ;DIVIDE BY ACT. DEC. DIGIT
1385 006560  000771                            BR      2$              ;DO NEXT
1386 006562  004737  006122            3$:     CALL    $..KBO          ;PRINT THIS DEC. DIGIT
1387 006566  062737  000002  006614            ADD     #2,$POINT       ;POINT TO NEXT DEC. VALUE
```

```
$..DMP

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

```
                                        .SBTTL  $..ADP
                                ;*
                                ; THIS SUBROUTINE IS USED TO PRINT OUT ADDITIONAL INFORMATION
                                ; ABOUT AN ERROR. MOSTTIMES IT WILL BE USED TO PRINT THE
                                ; DATA IN CASE OF DATA ERROR.
                                ;
                                ;
                                ;       INPUTS  :  (SP) =ADDRESS OF FIRST ARGUMENT
                                ;
                                ;
                                ;       OUTPUT:    NONE
                                ;
                                ; AUTHOR:        MICHAEL ZILLER  CSS DP  MUNICH MAY 1977
006706                          $..ADP: WRITE   MER1    ;"SHOULD BE"
006716                                  DUMP    OCT,$SHLD       ;DUMP DATA
006730                                  WRITE   MER2            ;"WAS"
006740                                  DUMP    OCT,$WAS        ;DUMP   DATA
006752  000207                          RETURN                  ;RETURN TO MAIN
```

```
                                        .SBTTL  $..TOT
                                ;+
                                ;THIS ROUTINE IS ENTERED AFTER THE MACRO "ADRTST" HAS BEEN EVOCED
                                ;AND A BUS TIMEOUT HAS HAPPENED
                                ;
                                ;INPUT: 4(SP) = PC FOR  PROGRAM CONTINUATION
                                ;        R0   = REGISTER ADDRESS (E.G. 164000) WHICH SHOULD BE TESTED
                                ;
                                ;OUTPUT:        THE ERROR MESSAGE:
                                ;               THIS ADDRESS IS NOT AVAILABLE ON UNIBUS:  164000
                                ;AUTHOR:        ALBERT BREM,CSS MUC     FEB 1979
                                ;-
                                ;
                                ;
006754  062706  000004          $..TOT: ADD     #4,SP           ;CORRECT SP
006760                                  WRITE   $MNOAV          ;SAY ADDRESS NOT AVAILABLE ON UNIBUS
006770                                  DUMP    OCT,R0          ;DUMP REGISTER ADDRESS
007000  005737  007152                  TST     OUTFLG          ;SYNCHRONIZE PRINTOUT
007004  001375                          BNE     .-4             ;"
007006  012737  000006  000004          MOV     #6,@#4          ;REBUILD TRAP CATCHER
007014  000136                          JMP     @(SP)+          ;JUMP TO CONTINUATION LABEL
                                ;
                                ;
                                ;
                                ;
```

```
                                        .SBTTL  $..BUF
                                ;+
                                ;SUBSECTION TO CONTAIN ALL COM-BUFFERS, CONSTANTS AND
                                ;'NOT MACRO-DEFINED' PARAMETERS.
                                ;
                                ;AUTHOR:        BERT HUBER CSS/DP MUNICH
                                ;-
                                        ;.PSECT CSSMON
                                        .ENABL  LSB
                                ;MEMORY MANAGEMENT DEFINITIONS
        172360                  KDSAR0=172360
        172320                  KDSDR0=172320
        172340                  KISAR0=172340
        172352                  KISAR5=172352
        172354                  KISAR6=172354
        172356                  KISAR7=172356
        172300                  KISDR0=172300
        172314                  KISDR6=172314
        172316                  KISDR7=172316
        172200                  SISDR0=172200
        177660                  UDSAR0=177660
        177620                  UDSDR0=177620
        177640                  UISAR0=177640
        177650                  UISAR4=177650
        177652                  UISAR5=177652
        177654                  UISAR6=177654
        177656                  UISAR7=177656
        177600                  UISDR0=177600
        177610                  UISDR4=177610
        177612                  UISDR5=177612
        177614                  UISDR6=177614
        177616                  UISDR7=177616
        170200                  UBMPR=170200
        140000                  CMODE=140000
        030000                  PMODE=30000
        177572                  SR0=177572
        172516                  SR3=172516
007016  000000                  $$PTYP: .WORD   0               ;IF 0:=LSI; IF -1:=PDP
007020  000000                  $$MTYP: .WORD   0               ;IF 0:NO MM; IF 1:WITH MM
007022  000000  000000            MEMEND: .WORD   0,0             ;TOP OF CORE AND CORE EXTENSION
                                                                ;STARTING FROM HERE TILL PROMES
                                                                ;THE CONTENTS WILL BE CLEARED
                                                                ;WITH "ABO"
007026                          KBBUFF: .BLKB   80.             ;TERMINAL INPUT BUFFER
007146                          KBBEND:                         ;REFERENCE LABEL
007146  007026                  KBBPNT: .WORD   KBBUFF  ;INPUT BUFFER POINTER
007150  000000                  INFLAG: .WORD   0       ;INPUT 'RUNNING" FLAG
007152  000000                  OUTFLG:  WORD   0       ;OUTPUT 'RUNNING" FLAG
007154  000000                  ERRFLG: .WORD   0               ;'ERROR HANDLING REQ.' FLAG
007156  000000                  INPREQ: .WORD   0               ;'WAIT FOR SPECIAL INPUT" FLAG
007160  000000                  PSDSWR: .WORD   0               ;SOFTWARE  SWITCHREGISTER
007162  000000                  RUBFLG: .WORD   0               ;RUBOUTMODE-INDICATOR FLAG
007164  000000                  STATUS: .WORD   0               ;BUFFER PS-CONTENT IF SAVED
007166  000000                  $..ER:  .WORD   0               ;HOLDS CURRENT ERROR NUMBER
007170  000000                  SPEFLG: .WORD   0               ;USED IN $..KBI AND $..WRT
007172  001000                  SCOPAD: 1000            ;SAVE AREA FOR SCOPE LOOP ADDRESS
007174  000000                  $.$ASF: .WORD   0               ;SPECIAL ASCII INPUT FLAG
```

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

```
                                .TITLE          IEC11-B TEST
                                .SBTTL  LOCAL MACRO DEFINITIONS
                        ;
                        ;       THIS SECTION CONTAINS ONLY MACROS WHICH ARE
                        ;       USED SEPARATE FROM THE MACRO LIBRARY (NEWMAC).
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                                .MACRO  MCLEAR
                                MOV     #BIT5,@CIR              ; CLEAR REGISTERS IN IEC11-A
                                MOV     #BIT3,@CSR              ; CLEAR REGISTERS IN IEC11-B
                                .ENDM   MCLEAR
                        ;
                        ;
                        ;
                        ;
                                .MACRO  DATACC
                                BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
                                BEQ     .-6                     ; IF YES,CONTINUE
                                BIC     #BIT15,@CIR             ; CLEAR "DATA ACC" IN CIR
                                .ENDM   DATACC
                        ;
                        ;
                        ;
                                .MACRO  STALL
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                NOP
                                .ENDM
                        ;
                        ;
                        ;
                        ;       THE MACRO "CACS" SETS THE IEC11 A
                        ;       TO CONTROLLER ACTIVE STATE (CONTROLLER IN CHARGE)
                        ;
                                .MACRO  CACS,?A,?B
                                BIS     #BIT0,@CIR              ; SET SACS
```

LOCAL MACRO DEFINITIONS

```
                                        BIS     #BIT6,@SMR              ; SET SIC
                                A:      BIT     #BIT13,@SMR             ; IS SIAS STILL SET ?
                                        BNE     A                       ; IF NOT,TEST AGAIN
                                B:      BIT     #BIT8,@SMR              ; IS CACS SET ?
                                        BEQ     B                       ; IF NOT,WAIT
                                        BIC     #177400,@CIR            ;
                                        .ENDM   CACS
```

```
 77                                             .SBTTL          INPUT PROCEDURE
 78 010614  032737  000003  007024  BUFPRP: BIT     #3,MEMEND+2             ; MORE THAN 32 K ?
 79 010622  001012                          BNE     1$                      ;
 80 010624                                  IF MEMEND HI #157776 GOTO 1$    ;
 81 010640  013737  007022  026772          MOV     MEMEND,MEMSAV           ;
 82 010646  000403                          BR      2$                      ;
 83 010650  012737  157776  026772  1$:     MOV     #157776,MEMSAV          ; USE 28K FOR BUFFER
 84 010656  013700  026772          2$:     MOV     MEMSAV,R0               ;
 85 010662  162700  000277                  SUB     #277,R0                 ; SAVE ABSOLUTE LOADER AREA
 86 010666  010037  027004                  MOV     R0,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,R0                ; BUILD LENGTH
 89 010704  162700  031623                  SUB     #PRGEND-1,R0            ; IN BYTES
 90 010710  006200                          ASR     R0                      ; OF
 91 010712  042700  100000                  BIC     #BIT15,R0               ;
 92 010716  010037  027000                  MOV     R0,BUFFL                ; BUFFER A OR B
 93 010722  062700  031624                  ADD     #PRGEND,R0              ; BUILD FIRST BYTE ADDRESS OF
 94 010726  010037  027002                  MOV     R0,BUFBB                ; BUFFER B
 95 010732  162700  000001                  SUB     #1,R0                   ; BUILD LAST BYTE ADDRESS
 96 010736  010037  026776                  MOV     R0,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,R0                  ; INPUT REQUEST
100 010770                                  IF R0 NE #0 GOTO 1$             ;
101 011000  012700  160010                  MOV     #160010,R0              ; SAVE DEFAULT VALUE
102 011004  032700  000007          1$:     BIT     #7,R0                   ; CHECK THREE LSB OF INPUT
103 011010  001360                          BNE     RGAIN                   ; IF NOT CLEAR,REQUEST AGAIN
104 011012                                  IF R0 LO #160010 GOTO RGAIN     ;
105 011022                                  IF R0 HI #177776 GOTO RGAIN     ;
106 011032                                  ADRTST  RGAIN                   ;
107 011056  010037  026732                  MOV     R0,CIR                  ; SAVE CIR ADDRESS
108 011062  062700  000002                  ADD     #2,R0                   ; BUILD NEXT ADDRESS
109 011066                                  ADRTST  RGAIN                   ;
110 011112  010037  026734                  MOV     R0,SMR                  ; SAVE SMR REG ADDRESS
111 011116  062700  000002                  ADD     #2,R0                   ; BUILD NEXT ADDRESS
112 011122                                  ADRTST  RGAIN                   ;
113 011146  010037  026736                  MOV     R0,IOR                  ; SAVE IOR REG ADDRESS
114 011152  062700  000001                  ADD     #1,R0                   ; BUILD HIGH-BYTE ADDRESS
115 011156  010037  026740                  MOV     R0,IORHB                ; OF IOR
116 011162  062700  000001                  ADD     #1,R0                   ; BUILD NEXT ADDRESS
117 011166                                  ADRTST  RGAIN                   ;
118 011212  010037  026742                  MOV     R0,VSR                  ; SAVE VSR REG ADDRESS
119 011216                          VECAIN: WRITE   MES2                    ; REQUEST VECTOR ADDRESS OF IEC11-A
120 011226                                  READ    OCT,R0                  ; INPUT REQUEST
121 011234                                  IF R0 NE #0 GOTO 2$             ;
122 011244  012700  000270                  MOV     #270,R0                 ; SAVE DEFAULT VALUE
123 011250                          2$:     IF R0 LO #40 GOTO VECAIN        ;
124 011260                                  IF R0 HI #774 GOTO VECAIN       ;
125 011270  020077  015446                  CMP     R0,@VSR                 ; 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 R0,VECA                     ; SAVE IEC11-A VECTOR ADDRESS
130 011316                          ADDRIN: WRITE   MES5                    ; REQUEST IEC11-A BUS ADDRESS
131 011326                                  READ    OCT,R0                  ; INPUT
132 011334                                  IF R0 HI #36 GOTO ADDRIN        ;
133 011344                                  IF R0 EQ #0 GOTO DEFA           ;
```

```
134 011354                                      IF RO EQ #35 GOTO DEFA          ;
135 011366  010037  026762                      MOV     RO,IECAAD               ; SAVE IEC11-A BUS ADDRESS
136 011372  000403                              BR      RGBIN                   ; GOTO GET NEXT REGISTER ADDRESS
137 011374  012737  000035  026762  DEFA:       MOV     #35,IECAAD              ; SAVE DEFAULT VALUE
138 011402                          RGBIN:      WRITE   MES3                    ; ENTER FIRST REG OF IEC11-B
139 011412                                      READ OCT,RO                     ; INPUT REQUEST
140 011420                                      IF RO NE #0 GOTO 4$             ;
141 011430  012700  160020                      MOV     #160020,RO              ; SAVE DEFAULT VALUE
142 011434  032700  000007          4$:         BIT     #7,RO                   ; CHECK THREE LSB OF INPUT
143 011440  001360                              BNE     RGBIN                   ; ARE ZERO
144 011442                                      IF RO LO #160020 GOTO RGBIN     ;
145 011452                                      IF RO HI #177776 GOTO RGBIN     ;
146 011462                                      ADRTST  RGBIN                   ;
147 011506  010037  026744                      MOV     RO,CSR                  ; SAVE CSR REG ADDRESS
148 011512  062700  000002                      ADD     #2,RO                   ; BUILD NEXT REG ADDRESS
149 011516                                      ADRTST  RGBIN                   ;
150 011542  010037  026746                      MOV     RO,BCR                  ; SAVE BCR REG ADDRESS
151 011546  062700  000002                      ADD     #2,RO                   ; BUILD NEXT REG ADDRESS
152 011552                                      ADRTST  RGBIN                   ;
153 011576  010037  026750                      MOV     RO,BAR                  ; SAVE BAR REG ADDRESS
154 011602  062700  000002                      ADD     #2,RO                   ; BUILD NEXT REG ADDRESS
155 011606                                      ADRTST  RGBIN                   ;
156 011632  010037  026752                      MOV     RO,MCR                  ; SAVE MCR REG ADDRESS
157 011636  062700  000001                      ADD     #1,RO                   ; BUILD MCR-HIGHBYTE
158 011642  010037  026754                      MOV     RO,MCRHB                ;
159 011646                          VECBIN:     WRITE   MES4                    ; REQUEST IEC11-B VECTOR
160 011656                                      READ    OCT,RO                  ; INPUT REQUEST
161 011664                                      IF RO NE #0 GOTO 5$             ;
162 011674  012700  000274                      MOV     #274,RO                 ; SAVE DEFAULT VALUE
163 011700                          5$:         IF RO LO #40 GOTO VECBIN        ;
164 011710                                      IF RO HI #774 GOTO VECBIN       ;
165 011720  010037  026760                      MOV     RO,VECB                 ; SAVE IEC11-B VECTOR ADDRESS
166 011724                          BDDRIN:     WRITE   MES7                    ; REQUEST IEC11-B BUS ADDRESS
167 011734                                      READ    OCT,RO                  ; INPUT
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               ; SAVE IEC11-B BUS ADDRESS
172 012000  000403                              BR      $..SPC                  ; GOTO TEST SELECT
173 012002  012737  000036  026764  DEFB:       MOV     #36,IECBAD              ; SAVE DEFAULT VALUE
```

```
                                                .SBTTL  TEST SELECT ROUTINE
012010                                  $..SPC: REMVEC  @VECA                   ;
012016                                          REMVEC  @VECB                   ;
012024  005737  007152                          TST     OUTFLG                  ; WAIT FOR ANY OUTPUT COMPLETE
012030  001375                                  BNE     .-4                     ; ...
012032  000005                                  RESET                           ; CLEAR THE WORLD
012034                                          SET6    @@TKS                   ; RE-ENABLE TTY
012042                                  LAB11:  WRITE   MESNEX                  ; REQUEST NEXT TEST TO RUN
012052                                          READ    OCT,R0                  ; WAIT FOR INPUT
012060                                          IF R0 HI @7 GOTO LAB11          ;
012070                                          IF R0 EQ @0 GOTO TESTAL         ;
012100  012737  177777  027024                  MOV     @177777,TSTREP          ; LOOP ON SINGLE TESTS
012106  006300                                  ASL     R0                      ; BUILD WORD OFFSET INTO TABLE
012110  010027                                  MOV     R0,(PC)+                ; AND SAVE
012112  000000                          OFFSET: .WORD   0                       ; HOLDS CURRENT TEST NUMBER
012114  013737  012112  012240          SINGLE: MOV     OFFSET,OFFCNT           ;
012122  062737  012220  012240                  ADD     @TT,OFFCNT              ;
012130  017737  000104  012240                  MOV     @@OFFCNT,OFFCNT         ;
012136  004777  000076                          JSR     PC,@@OFFCNT             ; GOTO SELECTED TEST
012142  000764                                  BR      SINGLE                  ; REPEAT TEST
012144  012737  000000  027024          TESTAL: MOV     @0,TSTREP               ; NO LOOP ON SINGLE TESTS
012152  012727  012222                          MOV     @TT+2,(PC)+             ; GET ADDRESS OF FIRST TEST
012156  000000                          SEQPNT: .WORD   0                       ; POINTER FOR TEST IN PROGRESS
012160  013737  012156  012240          SEQ:    MOV     SEQPNT,OFFCNT           ; LOAD ADDRESS OF NEXT TEST
012166  017737  000046  012240                  MOV     @@OFFCNT,OFFCNT         ;
012174  004777  000040                          JSR     PC,@@OFFCNT             ; GOTO EXECUTE TEST
012200  023727  012156  012236                  CMP     SEQPNT,@TTE-4           ; ALL DONE ?
012206  001756                                  BEQ     TESTAL                  ; IF EQ, YES
012210  062737  000002  012156                  ADD     @2,SEQPNT               ; POINT TO NEXT TEST
012216  000760                                  BR      SEQ                     ; AND DO NEXT
012220  000000  012242  013074          TT:     .WORD   0,TEST1,TEST2,TEST3,TEST4,TEST5,TEST6,TEST7
012226  014352  017350  020442
012234  021350  022206
012240  000000                          OFFCNT: .WORD   0                       ;
012242                                  TTE:
```

```
209                                             .SBTTL   TEST 1: REGISTER STATIC TEST
210 012242                              TEST1:  WRITE   MES8                    ; IDENTIFY TEST 1
211 012252  104000                      REP1:   SCOPE                           ;
212 012254                                      MCLEAR                          ;
213 012270  052777  000161  014446              BIS     #161,@CSR               ; 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,@CIR                 ; ALL BITS RESET IN CIR ?
219 012342  001417                              BEQ     1$                      ; IF YES,CONTINUE
220 012344                                      ADP     #0,@CIR                 ;
221 012376                                      ERROR   1.                      ;
222 012402  022777  000000  014324  1$:         CMP     #0,@SMR                 ; ALL BITS RESET IN SMR ?
223 012410  001417                              BEQ     2$                      ; IF YES,CONTINUE
224 012412                                      ADP     #0,@SMR                 ;
225 012444                                      ERROR   2.                      ;
226 012450  022777  000000  014260  2$:         CMP     #0,@IOR                 ; ALL BITS RESET IN IOR ?
227 012456  001417                              BEQ     3$                      ; IF YES,CONTINUE
228 012460                                      ADP     #0,@IOR                 ;
229 012512                                      ERROR   3.                      ;
230 012516  022777  000000  014220  3$:         CMP     #0,@CSR                 ; ALL BITS RESET IN CSR ?
231 012524  001417                              BEQ     4$                      ; IF YES,CONTINUE
232 012526                                      ADP     #0,@CSR                 ;
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$:         CMP     #0,@MCR                 ; ALL BITS RESET IN MCR ?
243 012706  001417                              BEQ     7$                      ; IF YES,CONTINUE
244 012710                                      ADP     #0,@MCR                 ;
245 012742                                      ERROR   7.                      ;
246 012746  012746  000060          7$:         MOV     #60,-(SP)               ; SAVE BIT MASK
247 012752  012746  026744                      MOV     #CSR,-(SP)              ; SAVE REGISTER
248 012756  004737  031100                      CALL    REGTST                  ; TEST R/W BITS
249 012762  012746  000100                      MOV     #100,-(SP)              ; SAVE BIT MASK
250 012766  012746  026744                      MOV     #CSR,-(SP)              ; SAVE  REGISTER
251 012772  004737  031100                      CALL    REGTST                  ; TEST R/W BITS
252 012776  012777  000010  013740              MOV     #BIT3,@CSR              ; CLEAR REGISTERS IN IEC11-B
253 013004  012746  177777                      MOV     #177777,-(SP)           ; SAVE BIT MASK
254 013010  012746  026746                      MOV     #BCR,-(SP)              ; SAVE  REGISTER
255 013014  004737  031100                      CALL    REGTST                  ; TEST R/W BITS
256 013020  012746  177777                      MOV     #177777,-(SP)           ; SAVE BIT MASK
257 013024  012746  026750                      MOV     #BAR,-(SP)              ; SAVE REGISTER
258 013030  004737  031100                      CALL    REGTST                  ; TEST R/W BITS
259 013034  012746  137777                      MOV     #137777,-(SP)           ; SAVE BIT MASK
260 013040  012746  026752                      MOV     #MCR,-(SP)              ; SAVE REGISTER
261 013044  004737  031100                      CALL    REGTST                  ; TEST R/W BITS
262 013050  005737  027024                      TST     TSTREP                  ; TEST LOOP SELECTED ?
263 013054  001402                              BEQ     8$                      ; IF NOT,LEAVE THE TEST
264 013056  000137  012252                      JMP     REP1                    ;
265 013062                              8$:     WRITE   MESEND                  ; END OF TEST
```

```
    266 013072  000207                        RETURN                          ;
```

```
                                                .SBTTL  TEST 2: TALKER AND LISTENER FUNCTION TEST
013074                                  TEST2:  WRITE   MES9                    ; IDENTIFY TEST 2
013104  104000                                 SCOPE                           ; LOOP FOR ERROR 3
013106                                  REP2:   MCLEAR                          ; TEST LOOP ENTRY
013122  052777  000001  013602                  BIS     #BIT0,@CIR              ; SET SACS
013130  052777  000100  013576                  BIS     #BIT6,@SMR              ; SET SIC
013136  005037  027006                          CLR     CNT1                    ;
013142  032777  020000  013564  1$:             BIT     #BIT13,@SMR             ; IS SIAS STILL SET ?
013150  001405                                  BEQ     2$                      ; IF NOT,CONTINUE
013152  005237  027006                          INC     CNT1                    ; IF YES,WAIT A WHILE
013156  001371                                  BNE     1$                      ;
013160                                          ERROR   8.                      ;
013164  032777  000400  013542  2$:             BIT     #BIT8,@SMR              ; IS CACS SET ?
013172  001002                                  BNE     3$                      ;
013174                                          ERROR   9.                      ;
013200  042777  177400  013524  3$:             BIC     #177400,@CIR            ;
013206  013700  026766                          MOV     TAIND,R0                ; BUILD MTA B
013212  063700  026764                          ADD     IECBAD,R0               ;       "
013216  110077  013514                          MOVB    R0,@IOR                 ; SET B TO TADS
013222  005037  027006                          CLR     CNT1                    ;
013226  032777  100000  013476  4$:             BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
013234  001007                                  BNE     5$                      ; IF YES,CONTINUE
013236  005237  027006                          INC     CNT1                    ; IF NOT,WAIT A WHILE
013242  005737  027006                          TST     CNT1                    ; WAITLOOP FINISHED ?
013246  001367                                  BNE     4$                      ;
013250                                          ERROR   10.                     ;
013254  042777  100000  013450  5$:             BIC     #BIT15,@CIR             ; CLEAR "DATA ACC" IN CIR
013262  052777  000004  013444                  BIS     #BIT2,@SMR              ; GOTO STANDBY STATE,B TO TACS
013270  022777  001204  013446                  CMP     #1204,@CSR              ; IS TACS,STATE CH AND INT IN B SET?
013276  001420                                  BEQ     LAB1                    ; IF YES CONTINUE
013300                                          ADP     #1204,@CSR              ;
013332                                          ERROR   11.                     ;
013336  104000                                  SCOPE                           ; LOOP FOR ERROR 4
013340  042777  001000  013376  LAB1:           BIC     #BIT9,@CSR              ; CLEAR STATE CHANGE IN CSR
013346  022777  000004  013370                  CMP     #4,@CSR                 ; TACS SET,STATE CH AND INT CLEAR?
013354  001420                                  BEQ     LAB2                    ; IF YES,CONTINUE
013356                                          ADP     #4,@CSR                 ;
013410                                          ERROR   12.                     ;
013414  104000                                  SCOPE                           ; LOOP FOR ERROR 5
013416  052777  000002  013310  LAB2:           BIS     #BIT1,@SMR              ; LEAVE STANDBY STATE
013424  022777  001200  013312                  CMP     #1200,@CSR              ; STATE CH AND INT SET,TACS CLEAR?
013432  001420                                  BEQ     LAB3                    ; IF YES CONTINUE
013434                                          ADP     #1200,@CSR              ;
013466                                          ERROR   13.                     ;
013472  104000                                  SCOPE                           ;
013474  042777  001000  013242  LAB3:           BIC     #BIT9,@CSR              ; CLEAR STATE CHANGE IN CSR
013502  013700  026766                          MOV     TAIND,R0                ; BUILD MTA A
013506  063700  026762                          ADD     IECAAD,R0               ;       "
013512  110077  013220                          MOVB    R0,@IOR                 ; A SETS ITSELF TO TACS,UNADDR, OTHER
013516                                          DATACC                          ;
013534  052777  000004  013172                  BIS     #BIT2,@SMR              ; GO TO STANDBY STATE
013542  022777  000000  013174                  CMP     #0,@CSR                 ; TACS CLEAR ?
013550  001420                                  BEQ     LAB4                    ; IF YES,CONTINUE
013552                                          ADP     #0,@CSR                 ;
013604                                          ERROR   14.                     ;
013610  104000                                  SCOPE                           ; LOOP FOR ERROR 7
013612  052777  000002  013114  LAB4:           BIS     #BIT1,@SMR              ; LEAVE STANDBY STATE
```

```
325 013620  013700  026766              MOV     TAIND,R0        ; BUILD MTA B
326 013624  063700  026764              ADD     IECBAD,R0       ;       "
327 013630  110077  013102              MOVB    R0,@IOR         ; SET B TO TADS
328 013634                              DATACC                  ;
329 013652  013700  026770              MOV     LIIND,R0        ; BUILD MLA B
330 013656  063700  026764              ADD     IECBAD,R0       ;       "
331 013662  110077  013050              MOVB    R0,@IOR         ; SET B TO LADS
332 013666                              DATACC                  ;
333 013704  052777  000004  013022      BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,B TO LACS
334 013712  022777  001202  013024      CMP     #1202,@CSR      ; LACS,STATE CH AND INT SET ?
335 013720  001417                      BEQ     LAB5            ; IF YES,CONTINUE
336 013722                              ADD     #1202,@CSR      ; ELSE ERROR
337 013754                              ERROR   15.             ;
338 013760  104000              LAB5:   SCOPE                   ; LOOP FOR ERROR 8
339 013762  042777  001000  012754      BIC     #BIT9,@CSR      ; CLEAR STATE CHANGE IN CSR
340 013770  022777  000002  012746      CMP     #2,@CSR         ; LACS SET,STATE CHANGE AND INT RESET ?
341 013776  001417                      BEQ     LAB6            ; IF TRUE,CONTINUE
342 014000                              ADD     #2,@CSR         ;
343 014032                              ERROR   16.             ;
344 014036  104000              LAB6:   SCOPE                   ; LOOP FOR ERROR 9
345 014040  052777  000002  012666      BIS     #BIT1,@SMR      ; LEAVE STANDBY STATE
346 014046  022777  001200  012670      CMP     #1200,@CSR      ; STATE CHGE AND INT SET,LACS CLEAR ?
347 014054  001417                      BEQ     LAB7            ; IF YES,CONTINUE
348 014056                              ADD     #1200,@CSR      ;
349 014110                              ERROR   17.             ;
350 014114  042777  001000  012622  LAB7: BIC     #BIT9,@CSR      ; CLEAR STATE CHGE AND INT
351 014122  104000                      SCOPE                   ; LOOP FOR ERROR 10
352 014124  012700  000077              MOV     #77,R0          ;
353 014130  110077  012602              MOVB    R0,@IOR         ; UNLISTEN IEC11-B
354 014134                              DATACC                  ;
355 014152  052777  000004  012554      BIS     #BIT2,@SMR      ; GOTO STANDBY STATE
356 014160  032777  000002  012556      BIT     #BIT1,@CSR      ; IS LACS RESET ?
357 014166  001403                      BEQ     LAB8            ; IF YES,CONTINUE
358 014170                              ERROR   18.             ;
359 014174  104000                      SCOPE                   ; LOOP FOR ERROR 11
360 014176  052777  000002  012530  LAB8: BIS     #BIT1,@SMR      ; LEAVE STANDBY STATE
361 014204  013700  026770              MOV     LITND,R0        ; BUILD MLA B
362 014210  063700  026764              ADD     IECBAD,R0       ;       "
363 014214  110077  012516              MOVB    R0,@IOR         ; SET B TO LADS
364 014220                              DATACC                  ;
365 014236  013700  026766              MOV     TAIND,R0        ; BUILD MTA B
366 014242  063700  026764              ADD     IECBAD,R0       ;       "
367 014246  110077  012464              MOVB    R0,@IOR         ; SET B TO TADS
368 014252  052777  000004  012454      BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,SET B TO TACS
369 014260  022777  001204  012456      CMP     #1204,@CSR      ; TACS,STATE CHGE AND INT SET ?
370 014266  001417                      BEQ     LAB9            ; IF YES,CONTINUE
371 014270                              ADD     #1204,@CSR      ;
372 014322                              ERROR   19.             ;
373 014326  005737  027024      LAB9:   TST     ISTREP          ; TEST LOOP SELECTED ?
374 014332  001402                      BEQ     1$              ; IF NOT,LEAVE THE TEST
375 014334  000137  013106              JMP     REP2            ;
376 014340                      1$:     WRITE   MESEND          ; END OF TEST
377 014350  000207                      RETURN
```

```
                                                .SBTTL TEST 3: GENERAL INTERRUPT AND DMA FUNCTION TEST
014352                                  TEST3:  WRITE   MES10                   ;
014362  005737  007152                        TST     OUTFLG                  ; WAIT FOR ANY OUTPUT COMPLETE
014366  001375                                BNE     .-4                     ; ...
014370  104000                                SCOPE                           ;
014372                                  REP3:   MCLEAR                          ;
014406                                          CACS                            ;
014450                                          SETVEC  #VEC8,#INTTST,#340      ;
014466  013700  026770                        MOV     LIIND,R0                ; BUILD MLA B
014472  063700  026764                        ADD     IEC8AD,R0               ;       "
014476  110077  012234                        MOVB    R0,@IOR                 ; SET B TO LADS
014502                                          DATACC                          ;
014520  013700  026766                        MOV     TAIND,R0                ; BUILD MTA A
014524  063700  026762                        ADD     IECAAD,R0               ;       "
014530  110077  012202                        MOVB    R0,@IOR                 ; SET A TO TADS
014534                                          DATACC                          ;
014552  052777  000004  012154                BIS     #BIT2,@SMR              ; GOTO STANDBY ,SET A TO TACS,B TO LACS
014560  005037  027022                        CLR     INTFLG                  ;
014564  052777  000100  012152  STCHGE: BIS     #BIT6,@CSR              ; ENABLE INTERRUPT
014572  012737  177000  027006                MOV     #177000,CNT1            ; PREPARE DELAY
014600  005737  027022          1$:     TST     INTFLG                  ; HAS INTERRUPT OCCURRED ?
014604  001010                                BNE     2$                      ; IF YES,CONTINUE
014606  005237  027006                        INC     CNT1                    ;
014612  001372                                BNE     1$                      ;
014614  042777  000100  012122                BIC     #BIT6,@CSR              ; DISABLE INTERRUPTS
014622                                          ERROR   20.                     ;
014626  032777  000100  012110  2$:      BIT     #100,@CSR               ; IS INTERRUPT ENABLE BIT RESET ?
014634  001402                                BEQ     3$                      ; IF YES,CONTINUE
014636                                          ERROR   21.                     ;
014642  022777  001202  012074  3$:      CMP     #1202,@CSR              ; CAUSED STATE CHGE THE INTERRUPT ?
014650  001420                                BEQ     NEXBIT                  ; IF YES,CONTINUE
014652                                          ADP     #1202,@CSR              ;
014704                                          ERROR   22.                     ;
014710  104000                                SCOPE                           ;
014712                                  NEXBIT: SETVEC  #VEC8,#INTSRV,#340      ;
014730  005037  027016                        CLR     NEXFLG                  ;
014734  042777  177400  012002                BIC     #177400,@CSR            ; CLEAR INTERRUPT CAUSING BITS
014742  022737  157776  007022                CMP     #157776,MEMEND          ;
014750  001121                                BNE     NEX1                    ;
014752  022737  000003  007024                CMP     #3,MEMEND+2             ; 128K MEMORY ?
014760  001115                                BNE     NEX1                    ;
014762  012737  026722  000004                MOV     #NEXMEM,@#4             ; SETUP VECTOR
014770  052777  000060  011746                BIS     #60,@CSR                ; ENABLE ADDRESSING OVER 32K
014776  012700  160000                        MOV     #160000,R0              ; TRY FIRST ADDRESS IN I/O PAGE
015002  005710                          1$:     TST     (R0)                    ; NON-EXISTANT ADDRESS?
015004  005737  027016                        TST     NEXFLG                  ;
015010  001003                                BNE     2$                      ;
015012  062700  000002                        ADD     #2,R0                   ; IF NOT,TRY NEXT ONE
015016  000771                                BR      1$                      ;
015020  012737  000006  000004  2$:      MOV     #6,@#4                  ; REBUILD VECTOR
015026  010077  011716                        MOV     R0,@BAR                 ; NEX-ADDRESS TO BAR
015032  012777  000001  011706                MOV     #1,@BCR                 ; PREPARE BCR
015040  052777  000101  011676                BIS     #101,@CSR               ; ENABLE INTERRUPT AND FUNCTION (DMA)
015046  112777  000252  011662                MOVB    #252,@IOR               ; DUMMY VALUE TO OUTPUT REGISTER
015054  012737  177000  027006                MOV     #177000,CNT1            ; PREPARE DELAY
015062  032777  100000  011642  6$:      BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
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,@CSR          ; DISABLE INTERRUPTS
 497 015552                                ERROR   30.                 ;
 498 015556  022777  100222  011160  3$:   CMP     #100222,@CSR        ; CAUSED NXM THE INTERRUPT ?
 499 015564  001420                        BEQ     NEX3                ; IF YES,CONTINUE
 500 015566                                ADP     #100222,@CSR        ;
 501 015620                                ERROR   31.                 ;
 502 015624  104000                        SCOPE                       ;
 503 015626  042777  177760  011110  NEX3: BIC     #177760,@CSR        ;
 504 015634  032737  100000  007022        BIT     #BIT15,MEMEND       ; MEMORY EXISTS ?
 505 015642  001067                        BNE     NEX4                ;
 506 015644  012777  100000  011076        MOV     #100000,@BAR        ; SELECT BUS ADDRESS 100000
 507 015652  052777  000101  011064        BIS     #101,@CSR           ; ENABLE INTERRUPT AND FUNCTION (DMA)
 508 015660  112777  000252  011050        MOVB    #252,@IOR           ; DUMMY VALUE TO OUTPUT REGISTER
 509 015666  012737  177000  027006        MOV     #177000,CNT1        ; PREPARE DELAY
 510 015674  032777  100000  011030  5$:   BIT     #BIT15,@CIR         ; 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$:   BIT     #BIT6,@CSR          ; HAS INTERRUPT OCCURRED ?
 517 015730  001410                        BEQ     3$                  ; IF YES,CONTINUE
 518 015732  005237  027006                INC     CNT1                ;
 519 015736  001371                        BNE     2$                  ;
 520 015740  042777  000100  010776        BIC     #BIT6,@CSR          ; DISABLE INTERRUPTS
 521 015746                                ERROR   33.                 ;
 522 015752  022777  100202  010764  3$:   CMP     #100202,@CSR        ; CAUSED NXM THE INTERRUPT ?
 523 015760  001420                        BEQ     NEX4                ; IF YES,CONTINUE
 524 015762                                ADP     #100202,@CSR        ;
 525 016014                                ERROR   34.                 ;
 526 016020  104000                        SCOPE                       ;
 527 016022  042777  177400  010714  NEX4: BIC     #177400,@CSR        ; CLEAR INTERRUPT CAUSING BITS
 528 016030  032737  040000  007022        BIT     #BIT14,MEMEND       ; MEMORY EXISTS ?
 529 016036  001067                        BNE     BCOVFL              ;
 530 016040  012777  040000  010702        MOV     #40000,@BAR         ; SELECT BUS ADDRESS 40000
 531 016046  052777  000101  010670        BIS     #101,@CSR           ; ENABLE INTERRUPT AND FUNCTION (DMA)
 532 016054  112777  000252  010654        MOVB    #252,@IOR           ; DUMMY VALUE TO OUTPUT REGISTER
 533 016062  012737  177000  027006        MOV     #177000,CNT1        ; PREPARE DELAY
 534 016070  032777  100000  010634  5$:   BIT     #BIT15,@CIR         ; 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$:   BIT     #BIT6,@CSR          ; HAS INTERRUPT OCCURRED ?
 541 016124  001410                        BEQ     2$                  ; IF YES,CONTINUE
 542 016126  005237  027006                INC     CNT1                ;
 543 016132  001371                        BNE     1$                  ;
 544 016134  042777  000100  010602        BIC     #BIT6,@CSR          ; DISABLE INTERRUPTS
 545 016142                                ERROR   36.                 ;
 546 016146  022777  100202  010570  2$:   CMP     #100202,@CSR        ; CAUSED NXM THE INTERRUPT ?
 547 016154  001420                        BEQ     BCOVFL              ; IF YES,CONTINUE
 548 016156                                ADP     #100202,@CSR        ;
 549 016210                                ERROR   37.                 ;
```

```
550 016214  104000                        SCOPE
551 016216  042777  177760  010520  BCOVFL: BIC     #177760,@CSR    ; CLEAR INTERRUPT CAUSING BITS
552 016224  012777  177777  010514          MOV     #177777,@BCR    ; PREPARE BCR WITH HIGHEST VALUE
553 016232  013777  027002  010510          MOV     BUFBB,@BAR      ; FIRST BUFFER ADDRESS
554 016240  052777  000101  010476          BIS     #101,@CSR       ; SET INTERRUPT ENABLE AND FUNCTION BIT
555 016246  112777  000252  010462          MOVB    #252,@IOR       ; DUMMY VALUE TO I/O REGISTER
556 016254  012737  177000  027006          MOV     #177000,CNT1    ; PREPARE DELAY
557 016262  032777  100000  010442  5$:     BIT     #BIT15,@CIR     ; DATA ACCEPTED ?
558 016270  001007                          BNE     1$              ; 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$:     BIT     #BIT6,@CSR      ; HAS INTERRUPT OCCURRED ?
564 016316  001410                          BEQ     2$              ; IF YES,CONTINUE
565 016320  005237  027006                  INC     CNT1            ;
566 016324  001371                          BNE     1$              ;
567 016326  042777  000100  010410          BIC     #BIT6,@CSR      ; DISABLE INTERRUPTS
568 016334                                   ERROR   39.             ;
569 016340  022777  004202  010376  2$:     CMP     #4202,@CSR      ; CAUSED BC OVFL THE INTERRUPT ?
570 016346  001420                          BEQ     ENDBIT          ; IF YES,CONTINUE
571 016350                                   ADP     #4202,@CSR      ;
572 016402                                   ERROR   40.             ;
573 016406  104000                          SCOPE                   ;
574 016410  042777  177400  010326  ENDBIT: BIC     #177400,@CSR    ; CLEAR INTERRUPT CAUSING BITS
575 016416  013777  027002  010324          MOV     BUFBB,@BAR      ; FIRST BUFFER ADDRESS
576 016424  012777  000001  010314          MOV     #1,@BCR         ; PREPARE BYTE COUNT REGISTER
577 016432  012777  100777  010312          MOV     #100777,@MCR    ; PREPARE MATCH CHARACTER
578 016440  052777  000101  010276          BIS     #101,@CSR       ; ENABLE FUNCTION (DMA) AND INTERRUPT
579 016446  112777  000377  010262          MOVB    #377,@IOR       ; FIRST BYTE IN I/O REGISTER
580 016454  012737  177000  027006          MOV     #177000,CNT1    ; PREPARE DELAY
581 016462  032777  100000  010242  5$:     BIT     #BIT15,@CIR     ; 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$:     BIT     #BIT6,@CSR      ; HAS INTERRUPT OCCURRED ?
588 016516  001410                          BEQ     2$              ; IF YES,CONTINUE
589 016520  005237  027006                  INC     CNT1            ;
590 016524  001371                          BNE     1$              ;
591 016526  042777  000100  010210          BIC     #BIT6,@CSR      ; DISABLE INTERRUPTS
592 016534                                   ERROR   42.             ;
593 016540  022777  002202  010176  2$:     CMP     #2202,@CSR      ; CAUSED "END" THE INTERRUPT ?
594 016546  001417                          BEQ     3$              ; IF YES,CONTINUE
595 016550                                   ADP     #2202,@CSR      ;
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:     REMVEC  #VECB           ;
602 016630                                   MCLEAR                  ;
603 016644                                   CACS                    ;
604 016706  013700  026766                  MOV     TAIND,R0        ; BUILD MTA A
605 016712  063700  026762                  ADD     IECAAD,R0       ;     "
606 016716  110077  010014                  MOVB    R0,@IOR         ; SET B TO TADS
```

```
 607 016722                                    DATACC                  ;
 608 016740  013700  026770                    MOV     LIIND,R0        ; BUILD MLA B
 609 016744  063700  026764                    ADD     IECBAD,R0       ;       "
 610 016750  110077  007762                    MOVB    R0,@IOR         ; SET A TO LADS
 611 016754                                    DATACC                  ;
 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                    PR1:    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                            PRI7:   WRITE   MES13           ;
 624 017122  000137  017330                    JMP     PRIEX           ; LEAVE TEST 3
 625 017126  012703  017164            PR2:    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                            PRI6:   WRITE   MES14           ;
 631 017174  000137  017330                    JMP     PRIEX           ; LEAVE TEST 3
 632 017200  012703  017236            PR3:    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                            PRI5:   WRITE   MES15           ;
 638 017246  000137  017330                    JMP     PRIEX           ; LEAVE TEST 3
 639 017252  012703  017310            PR4:    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                            PRI4:   WRITE   MES16           ;
 645 017320  000137  017330                    JMP     PRIEX           ; LEAVE TEST 3
 646 017324                            PRERR:  ERROR   44.             ; NO INTERRUPT OCCURED
 647 017330                            PRIEX:  REMVEC  #VECB           ;
 648 017336                                    WRITE   MESEND          ; END OF TEST
 649 017346  000207                            RETURN                  ;
```

```
                                                .SBTTL TEST 4: DMA-TRANSFER FROM B TO A (B IS TALKER)
017350                                  TEST4:  WRITE   MES11
017360  005737  027024                        TST     TSTREP          ; TEST LOOP SELECTED ?
017364  001004                                BNE     BCIN4           ;
017366  013737  027000  027020                MOV     BUFFL,BCINP     ; USE MAXIMAL BUFFER LENGTH
017374  000425                                BR      REP4            ;
017376                                  BCIN4:  WRITE   MES17           ; REQUEST BYTE COUNT
017406                                          DUMP    OCT,BUFFL       ; MAXIMAL INPUT
017420                                          READ    OCT,R0          ;
017426                                          IF R0 HI BUFFL GOTO BCIN4       ;
017436                                          IF R0 EQ #0 GOTO BCIN4          ;
017444  010037  027020                        MOV     R0,BCINP        ; SAVE BYTE COUNT
017450  005737  007152                  REP4:   TST     OUTFLG          ; WAIT FOR ANY OUTPUT COMPLETE
017454  001375                                BNE     .-4             ; ...
017456  104000                                SCOPE                   ;
017460                                          REMVEC  #VEC8           ;
017466  005037  027006                        CLR     CNT1            ;
017472  013700  027002                        MOV     BUFBB,R0        ; PREPARE
017476  005001                          1$:   CLR     R1              ; BUFFER B
017500  110120                          2$:   MOVB    R1,(R0)+        ; WITH
017502  005237  027006                        INC     CNT1            ;
017506  023737  027006  027000                CMP     CNT1,BUFFL      ; CONSECUTIVE
017514  001405                                BEQ     3$              ; VALUES
```

```
675 017516  005201                          INC     R1                      ; FROM
676 017520  022701  000400                  CMP     #400,R1                 ; 0 377
677 017524  001365                          BNE     2$                      ; OCTAL
678 017526  000763                          BR      1$                      ;
679 017530  013700  026774          3$:     MOV     BUFAB,R0                ; SET ALL
680 017534  112720  000377          4$:     MOVB    #377,(R0)+              ; BITS
681 017540  020037  027002                  CMP     R0,BUFBB                ; IN
682 017544  001373                          BNE     4$                      ; BUFFER A
683 017546                                  MCLEAR                          ;
684 017562                                  CACS                            ;
685 017624  013700  026766                  MOV     TAIND,R0                ; BUILD MTA B
686 017630  063700  026764                  ADD     IECBAD,R0               ;       "
687 017634  110077  007076                  MOVB    R0,@IOR                 ; SET B TO TADS
688 017640                                  DATACC                          ;
689 017656  013700  026770                  MOV     LIIND,R0                ; BUILD MLA A
690 017662  063700  026762                  ADD     IECAAD,R0               ;       "
691 017666  110077  007044                  MOVB    R0,@IOR                 ; SET A TO LADS
692 017672                                  DATACC                          ;
693 017710  052777  000004  007016          BIS     #BIT2,@SMR              ; GOTO STANDBY STATE
694 017716  104000                          SCOPE                           ;
695 017720  042777  177400  007016          BIC     #177400,@CSR            ; CLEAR INTERRUPT CAUSING BITS
696 017726  013777  027002  007014          MOV     BUFBB,@BAR              ; 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,@BCR                 ;       "
700 017746  013701  026774                  MOV     BUFAB,R1                ; POINTER TO BUFFER A
701 017752                                  SETVEC  #VECB,#INTSRV,#340      ;
702 017770  052777  000101  006746          BIS     #101,@CSR               ; 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  5$:     BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
707 020020  001013                          BNE     DMA4                    ; IF YES,CONTINUE
708 020022  005237  027006                  INC     CNT1                    ; IF NOT,WAIT A WHILE
709 020026  005737  027006                  TST     CNT1                    ; WAITLOOP FINISHED ?
710 020032  001367                          BNE     5$                      ;
711 020034                                  ERROR   45.                     ;
712 020040  032777  100000  006664  ACC4:   BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
713 020046  001774                          BEQ     ACC4                    ; IF YES,CONTINUE
714 020050  032777  000100  006666  DMA4:   BIT     #100,@CSR               ; HAS INTERRUPT OCCURRED ?
715 020056  001410                          BEQ     INTB4                   ; IF YES,GET LAST BYTE
716 020060  042777  100000  006644          BIC     #BIT15,@CIR             ;
717 020066  117721  006646                  MOVB    @IORHB,(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     10$                     ;
724 020120  032777  100000  006604          BIT     #BIT15,@CIR             ; DATA ACCEPTED ?
725 020126  001767                          BEQ     LAST4                   ; IF NOT,WAIT
726 020130  005237  027010          9$:     INC     CNT2                    ; IF YES,CHECK FOR
727 020134  005737  027014                  TST     DONE                    ; CORRECT BYTE COUNT
728 020140  001023                          BNE     11$                     ;
729 020142  042777  100000  006562          BIC     #BIT15,@CIR             ;
730 020150  117721  006564                  MOVB    @IORHB,(R1)+            ; BYTE TO BUFFER A
731 020154  023737  027010  027020          CMP     CNT2,BCINP              ; ALL BYTES DONE ?
```

```
732 020162  001351                        BNE     LAST4           ; IF YES,
733 020164  052737  177777  027014        BIS     #177777,DONE    ; SET FLAG
734 020172  000745                        BR      LAST4           ;
735 020174  005737  027014          10$:  TST     DONE            ; ALL DONE ?
736 020200  001005                        BNE     12$             ;
737 020202                                ERROR   46.             ; NOT ALL BYTES TRANSFERRED
738 020206  000402                        BR      12$             ;
739 020210                          11$:  ERROR   47.             ; TO MANY BYTES TRANSFERRED
740 020214  022777  004204  006522  12$:  CMP     #4204,@CSR      ; CAUSED "BC OVFL" THE INTERRUPT
741 020222  001417                        BEQ     13$             ; IF YES,CONTINUE
742 020224                                ADP     #4204,@CSR      ;
743 020256                                ERROR   48.             ;
744 020262  013700  026774          13$:  MOV     BUFAB,R0        ; PROVIDE FIRST BYTE OF BUFFER A
745 020266  013701  027002                MOV     BUFBB,R1        ; PROVIDE FIRST BYTE OF BUFFER B
746 020272  005037  027006                CLR     CNT1            ;
747 020276  005237  027006          CMP4: INC     CNT1            ; BYTE COUNT
748 020302  122021                        CMPB    (R0)+,(R1)+     ; BUFFER A EQUAL TO BUFFER B
749 020304  001440                        BEQ     14$             ; IF YES,CONTINUE
750 020306  114037  027026                MOVB    -(R0),ASAVE     ; IF NOT,SAVE DATA
751 020312  114137  027030                MOVB    -(R1),BSAVE     ;
752 020316  032737  040000  007160        BIT     #BIT14,PSDSWR   ; PRINTOUT ALLOWED ?
753 020324  001030                        BNE     14$             ;
754 020326                                WRITE   MES18           ; WRITE BYTE COUNT
755 020336                                DUMP    OCT,CNT1        ; DUMP NUMBER
756 020350                                ADP     BSAVE,ASAVE     ;
757 020402                                ERROR   49.             ;
758 020406  023737  027006  027020  14$:  CMP     CNT1,BCINP      ; ALL BYTES COMPARED ?
759 020414  001330                        BNE     CMP4            ; IF NOT,GET NEXT ONE
760 020416  005737  027024                TST     TSTREP          ; TEST LOOP SELECTED ?
761 020422  001402                        BEQ     15$             ; IF NOT,LEAVE THE TEST
762 020424  000137  017450                JMP     REP4            ;
763 020430                          15$:  WRITE   MESEND          ; END OF TEST
764 020440  000207                        RETURN
```

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

```
766                                             .SBTTL TEST 5: DMA-TRANSFER FROM A TO B (B IS LISTENER)
767 020442                              TEST5:  WRITE   MES12
768 020452  005737  027024                      TST     TSTREP          ; TEST LOOP SELECTED ?
769 020456  001004                              BNE     BCIN5           ;
770 020460  013737  027000  027020              MOV     BUFFL,BCINP     ; USE MAXIMAL BUFFER LENGTH
771 020466  000425                              BR      REP5            ;
772 020470                              BCIN5:  WRITE   MES17           ; REQUEST BYTE COUNT
773 020500                                      DUMP    OCT,BUFFL       ; MAXIMAL INPUT
774 020512                                      READ    OCT,R0          ;
775 020520                                      IF R0 HI BUFFL GOTO BCIN5       ;
776 020530                                      IF R0 EQ #0 GOTO BCIN5          ;
777 020536  010037  027020                      MOV     R0,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     BUFAB,R0        ; PREPARE
783 020564  005001                      1$:     CLR     R1              ; BUFFER A
784 020566  110120                      2$:     MOVB    R1,(R0)+        ; WITH
785 020570  020037  027002                      CMP     R0,BUFBB        ; 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$:     CLR     CNT1            ; SET
792 020614  013700  027002                      MOV     BUFBB,R0        ; ALL
793 020620  112720  000377              4$:     MOVB    #377,(R0)+      ; BITS
794 020624  005237  027006                      INC     CNT1            ; IN
795 020630  023737  027000  027006              CMP     BUFFL,CNT1      ; BUFFER
796 020636  001370                              BNE     4$              ; A
797 020640                                      MCLEAR                  ;
798 020654                                      CACS                    ;
799 020716  013700  026766                      MOV     TAIND,R0        ; BUILD MTA A
800 020722  063700  026762                      ADD     IECAAD,R0       ;       "
801 020726  110077  006004                      MOVB    R0,@IOR         ; SET A TO TADS
802 020732                                      DATACC                  ;
803 020750  013700  026770                      MOV     LIIND,R0        ; BUILD MLA B
804 020754  063700  026764                      ADD     IECBAD,R0       ;       "
805 020760  110077  005752                      MOVB    R0,@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     BUFBB,@BAR      ; FIRST BUFFER ADDRESS
811 021026  013700  027020                      MOV     BCINP,R0        ; PREPARE BYTE COUNT REGISTER
812 021032  005400                              NEG     R0              ;               "
813 021034  010077  005706                      MOV     R0,@BCR         ;               "
814 021040  013701  026774                      MOV     BUFAB,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
```

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

```
823 021132                                      ADP     #4202,@CSR      ;
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$:         CMP     CNT1,BCINP      ; ALL BYTES COMPARED ?
840 021322  001330                              BNE     CMP5            ; IF NOT,GET NEXT ONE
841 021324  005737  027024                      TST     TSTREP          ; TEST LOOP SELECTED ?
842 021330  001402                              BEQ     8$              ; IF NOT,LEAVE THE TEST
843 021332  000137  020542                      JMP     REP5            ;
844 021336                          8$:         WRITE   MESEND          ; END OF TEST
845 021346  000207                              RETURN
```

```
                                                .SBTTL TEST 6: MATCH CHARACTER REGISTER TEST (B IS LISTENER)
021350                                  TEST6:  WRITE   MES19
021360  005737  007152                        TST     OUTFLG          ; WAIT FOR ANY OUTPUT COMPLETE
021364  001375                                BNE     .-4             ; ...
021366                                          MCLEAR                  ;
021402                                          CACS                    ;
021444  013700  026766                        MOV     TAIND,R0        ; BUILD MTA A
021450  063700  026762                        ADD     IECAAD,R0       ;       "
021454  110077  005256                        MOVB    R0,@IOR         ; SET A TO TADS
021460                                          DATACC                  ;
021476  013700  026770                        MOV     LIIND,R0        ; BUILD MLA B
021502  063700  026764                        ADD     IECBAD,R0       ;       "
021506  110077  005224                        MOVB    R0,@IOR         ; SET B TO LADS
021512                                          DATACC                  ;
021530  052777  000004  005176                BIS     #BIT2,@SMR      ; GOTO STANDBY STATE
021536  042777  177400  005200          1$:   BIC     #177400,@CSR    ; CLEAR HIGH-BYTE IN CSR
021544                                          SETVEC  #VECB,#INTSRV,#340      ;
021562  104000                                  SCOPE                   ;
021564  012737  000001  027010          REP6:   MOV     #1,CNT2         ; FIRST COUNT OF MCR
021572  005037  027012                        CLR     CHAR            ; FIRST CHARACTER OF MCR
021576  042777  177400  005140          BEG6:   BIC     #177400,@CSR    ; CLEAR INTERRUPT CAUSING BITS
021604  005001                                CLR     R1              ;
021606  013700  026774                        MOV     BUFAB,R0        ; POINTER TO BUFFER A
021612  113720  027012                  1$:   MOVB    CHAR,(R0)+      ; FILL BUFFER A
021616  105201                                INCB    R1              ; WITH ONE CHARACTER
021620  123701  027010                        CMPB    CNT2,R1         ; LIMITED BY MATCH COUNT
021624  001372                                BNE     1$              ;
021626  013701  026774                        MOV     BUFAB,R1        ; PROVIDE FIRST BUFFER ADDRESS A
021632  013777  027002  005110                MOV     BUFBB,@BAR      ; PREPARE BAR
021640  012777  000001  005100                MOV     #1,@BCR         ; DUMMY VALUE TO BCR
021646  113777  027010  005100                MOVB    CNT2,@MCRHB     ; COUNT TO REGISTER
021654  113777  027012  005070                MOVB    CHAR,@MCR       ; CHARACTER TO REGISTER
021662  152777  000200  005064                BISB    #BIT7,@MCRHB    ; ENABLE MATCH
021670  052777  000101  005046                BIS     #101,@CSR       ; ENABLE INTERRUPT AND DMA
021676  112177  005034              DMA6:     MOVB    (R1)+,@IOR      ; ONE BYTE TO IOR
021702                                          DATACC                  ;
021720  032777  000100  005016                BIT     #100,@CSR       ; HAS INTERRUPT OCCURRED ?
021726  001363                                BNE     DMA6            ; IF NOT,CONTINUE DMA
021730  022777  002202  005006                CMP     #2202,@CSR      ; CAUSED "END" THE INTERRUPT
021736  001417                                BEQ     3$              ; IF YES,CONTINUE
021740                                          ADP     #2202,@CSR      ;
021772                                          ERROR   52.             ;
021776  013700  026774              3$:       MOV     BUFAB,R0        ; PROVIDE FIRST BYTE OF BUFFER A
022002  013701  027002                        MOV     BUFBB,R1        ; PROVIDE FIRST BYTE OF BUFFER B
022006  005002                                CLR     R2              ;
022010  020237  027010              CMP6:     CMP     R2,CNT2         ; END OF BYTE COUNT ?
022014  001442                                BEQ     6$              ;
022016  005202                                INC     R2              ; BYTE COUNT
022020  122021                                CMPB    (R0)+,(R1)+     ; BUFFER A EQUAL TO BUFFER B
022022  001772                                BEQ     CMP6            ; IF YES,CONTINUE
022024  114037  027026                        MOVB    -(R0),ASAVE     ; IF NOT,SAVE DATA
022030  114137  027030                        MOVB    -(R1),BSAVE     ;
022034  032737  040000  007160                BIT     #BIT14,PSDSWR   ; PRINTOUT ALLOWED ?
022042  001010                                BNE     5$              ;
022044                                          WRITE   MES18           ; WRITE BYTE COUNT
022054                                          DUMP    OCT,R2          ; DUMP NUMBER
022064                              5$:         ADP     ASAVE,BSAVE     ;
```

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

```
 904 022116                                      ERROR   53.                     ;
 905 022122  005237  027012          6$:         INC     CHAR                    ; NEXT MATCH CHARACTER
 906 022126  122737  000400  027012              CMPB    #400,CHAR               ; LAST CHARACTER ?
 907 022134  001220                              BNE     BEG6                    ;
 908 022136  005037  027012                      CLR     CHAR                    ; FIRST CHARACTER
 909 022142  005237  027010                      INC     CNT2                    ; NEXT MATCH COUNT
 910 022146  122737  000100  027010              CMPB    #100,CNT2               ; ALL COUNTS DONE ?
 911 022154  001402                              BEQ     7$                      ;
 912 022156  000137  021576                      JMP     BEG6                    ;
 913 022162  005737  027024          7$:         TST     TSTREP                  ; TEST LOOP SELECTED ?
 914 022166  001402                              BEQ     8$                      ; IF NOT,LEAVE THE TEST
 915 022170  000137  021564                      JMP     REP6                    ;
 916 022174                          8$:         WRITE   MESEND                  ; END OF TEST
 917 022204  000207                              RETURN
```

```
                                                .SBTTL TEST 7: SERIAL POLL PROCEDURE TEST
022206                                  TEST7:  WRITE   MES20
022216  005737  007152                         TST     OUTFLG          ; WAIT FOR ANY OUTPUT COMPLETE
022222  001375                                 BNE     .-4             ; ...
022224                                         SETVEC  #VECB,#INTSRV,#340      ;
022242                                  SETVEC  #VECA,#INTA,#340        ;
022260                                  REP7:   MCLEAR                  ;
022274                                          CACS                    ;
022336  104000                                  SCOPE                   ;
022340  052777  000002  004366          POLL0:  BIS     #BIT1,@SMR      ; GOTO CACS
022346  112777  000030  004362                  MOVB    #30,@IOR        ; SERIAL POLL ENABLE
022354  013700  026770                         MOV     LIIND,R0        ; BUILD MLA A
022360  063700  026762                         ADD     IECADD,R0       ;       "
022364  110077  004346                         MOVB    R0,@IOR         ; SET A TO LADS
022370                                         DATACC                  ;
022406  012701  000000                         MOV     #0,R1           ; BUILD FIRST TALKER ADDRESS
022412  013700  026766                  P0:    MOV     TAIND,R0        ;
022416  060100                                 ADD     R1,R0           ; BUILD TALKER ADDRESS
022420  110077  004312                         MOVB    R0,@IOR         ;
022424                                         DATACC                  ;
022442  052777  000004  004264                 BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
022450                                         STALL                   ;
022500  122777  000000  004232                 CMPB    #0,@IORHB       ; STATUS BYTE IS ZERO ?
022506  001416                                 BEQ     5$              ; IF YES,TRY THE NEXT ONE
022510  032737  040000  007160                 BIT     #BIT14,PSDSWR   ; PRINTOUT SELECTED ?
022516  001012                                 BNE     5$              ;
022520                                         WRITE   MES21           ; GIVE INFORMATION ABOUT
022530                                         DUMP    OCT,R1          ; ILLEGAL IEC-BUS ADDRESS
022540                                         ERROR   54.             ;
022544  005201                          5$:    INC     R1              ; GET NEXT TALKER ADDRESS
022546  022701  000037                         CMP     #37,R1          ; ALL DONE ?
022552  001405                                 BEQ     7$              ;
022554  052777  000002  004152                 BIS     #BIT1,@SMR      ; IF NOT,GOTO CACS
022562  000137  022412                         JMP     P0              ; AND SELECT THE NEXT ONE
022566  052777  000002  004140          7$:    BIS     #BIT1,@SMR      ; GOTO CACS
022574  112777  000031  004134                 MOVB    #31,@IOR        ; DISABLE SERIAL POLL
022602  052777  000004  004124                 BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
022610  000240                          DONE:  NOP                     ; JUST ENTRY-POINT
022612  042777  177400  004112                 BIC     #177400,@CIR    ; CLEAR INTERRUPT CAUSING BITS
022620  012737  177700  027006                 MOV     #177700,CNT1    ; PREPARE DELAY
022626  052777  000400  004110                 BIS     #BIT8,@CSR      ; REQUEST SERVICE
022634  052777  000100  004070                 BIS     #100,@CIR       ; ENABLE INTERRUPT A
022642  032777  000100  004062          1$:    BIT     #100,@CIR       ; HAS INTERRUPT OCCURRED ?
022650  001407                                 BEQ     2$              ; IF YES,CONTINUE
022652  005237  027006                         INC     CNT1            ;
022656  005737  027006                         TST     CNT1            ;
022662  001367                                 BNE     1$              ;
022664                                         ERROR   55.             ;
022670  022777  040201  004034          2$:    CMP     #040201,@CIR    ; CAUSED SRQ THE INTERRUPT ?
022676  001417                                 BEQ     POLL1           ; IF YES,CONTINUE
022700                                         ADP     #040201,@CIR    ;
022732                                         ERROR   56.             ;
022736  104000                          POLL1: SCOPE                   ;
022740  005037  027014                         CLR     DONE            ; CLEAR DONE FLAG
022744  052777  000002  003762                 BIS     #BIT1,@SMR      ; GOTO CACS
022752  112777  000030  003756                 MOVB    #30,@IOR        ; SERIAL POLL ENABLE
022760  013700  026770                         MOV     LIIND,R0        ; BUILD MLA A
```

```
 976 022764  063700  026762                ADD     IECAAD,R0          ;       "
 977 022770  110077  003742                MOVB    R0,@IOR            ; SET A TO LADS
 978 022774                                DATACC                     ;
 979 023012  012701  000000                MOV     #0,R1              ; BUILD FIRST TALKER ADDRESS
 980 023016  013700  026766          P1:   MOV     TAIND,R0           ;
 981 023022  060100                        ADD     R1,R0              ; BUILD TALKER ADDRESS
 982 023024  110077  003706                MOVB    R0,@IOR            ;
 983 023030                                DATACC                     ;
 984 023046  052777  000004  003660        BIS     #BIT2,@SMR         ; GOTO STANDBY STATE,A TO LACS
 985 023054                                STALL
 986 023104  122777  000000  003626        CMPB    #0,@IORHB          ; POLLING ADDRESS ?
 987 023112  001452                        BEQ     5$                 ; IF NOT,TRY THE NEXT ONE
 988 023114  120137  026764                CMPB    R1,IECBAD          ; LEGAL ADDRESS ?
 989 023120  001416                        BEQ     3$                 ; IF YES,CONTINUE
 990 023122  032737  040000  007160        BIT     #BIT14,PSDSWR      ; PRINTOUT SELECTED ?
 991 023130  001012                        BNE     3$                 ;
 992 023132                                WRITE   MES21              ; GIVE INFORMATION ABOUT
 993 023142                                DUMP    OCT,R1             ; ILLEGAL IEC-BUS ADDRESS
 994 023152                                ERROR   57.                ;
 995 023156  117702  003556          3$:   MOVB    @IORHB,R2          ; SAVE STATUS BYTE
 996 023162  042702  177400                BIC     #177400,R2         ;
 997 023166  022702  000100                CMP     #100,R2            ; STATUS BYTE OK ?
 998 023172  001416                        BEQ     4$                 ; IF YES,CONTINUE
 999 023174                                ADP     #100,R2            ;
1000 023224                                ERROR   58.                ;
1001 023230  012737  177777  027014  4$:   MOV     #177777,DONE       ; SET DONE FLAG
1002 023236  000411                        BR      PEND1              ; GOTO FINISH THIS TESTPART
1003 023240  005201                  5$:   INC     R1                 ; GET NEXT TALKER ADDRESS
1004 023242  022701  000037                CMP     #37,R1             ; ALL DONE ?
1005 023246  001405                        BEQ     PEND1              ;
1006 023250  052777  000002  003456        BIS     #BIT1,@SMR         ; 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     7$                 ; IF YES,GOTO NEXT PART OF THE TEST
1010 023270                                ERROR   59.                ;
1011 023274  104000                  7$:   SCOPE                      ;
1012 023276  052777  000002  003430        BIS     #BIT1,@SMR         ; GOTO CACS
1013 023304  112777  000031  003424        MOVB    #31,@IOR           ; DISABLE SERIAL POLL
1014 023312  052777  000004  003414        BIS     #BIT2,@SMR         ; GOTO STANDBY STATE,A TO LACS
1015 023320  032777  040000  003404        BIT     #BIT14,@CIR        ; 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,R0           ; BUILD MLA B
1022 023420  063700  026764                ADD     IECBAD,R0          ;       "
1023 023424  110077  003306                MOVB    R0,@IOR            ; SET B TO LADS
1024 023430                                DATACC                     ;
1025 023446  013700  026766                MOV     TAIND,R0           ; BUILD MTA A
1026 023452  063700  026762                ADD     IECAAD,R0          ;       "
1027 023456  110077  003254                MOVB    R0,@IOR            ; SET A TO TADS
1028 023462                                DATACC                     ;
1029 023500  052777  000004  003226        BIS     #BIT2,@SMR         ; GOTO STANDBY ,SET A TO TACS,B TO LACS
1030 023506  042777  177760  003230  SPOVFL: BIC   #177760,@CSR      ; CLEAR INTERRUPT CAUSING BITS
1031 023514  012777  177777  003224        MOV     #177777,@BCR       ; PREPARE BCR WITH HIGHEST VALUE
1032 023522  013777  027002  003220        MOV     BUFBB,@BAR         ; 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                                  DATACC                  ;
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                                  DATACC                  ;
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     IECAAD,R0       ;       "
1047 023656  110077  003054                  MOVB    R0,@IOR         ; SET A TO LADS
1048 023662                                  DATACC                  ;
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                                  DATACC                  ;
1054 023734  052777  000004  002772          BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
1055 023742                                  STALL                   ;
1056 023772  122777  000000  002740          CMPB    #0,@IORHB       ; 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$:     MOVB    @IORHB,R2       ; SAVE STATUS BYTE
1066 024050  042702  177400                  BIC     #177400,R2      ;
1067 024054  022702  000104                  CMP     #104,R2         ; STATUS BYTE OK ?
1068 024060  001416                          BEQ     4$              ; 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 TEST
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     IAIND,R0        ; BUILD MTA A
1092 024322  063700  026762                       ADD     IECAAD,R0       ;       "
1093 024326  110077  002404                       MOVB    R0,@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,R0        ; BUILD MLA A
1113 024530  063700  026762                       ADD     IECAAD,R0       ;       "
1114 024534  110077  002176                       MOVB    R0,@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     IAIND,R0        ;
1118 024566  060100                               ADD     R1,R0           ; BUILD TALKER ADDRESS
1119 024570  110077  002142                       MOVB    R0,@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,@IORHB       ; 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$:        MOVB    @IORHB,R2       ; SAVE STATUS BYTE
1133 024726  042702  177100                       BIC     #177400,R2      ;
1134 024732  022702  000110                       CMP     #110,R2         ; STATUS BYTE OK ?
1135 024736  001416                               BEQ     4$              ; 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 025006  022701  000037                       CMP     #37,R1          ; ALL DONE ?
1142 025012  001405                               BEQ     PEND3           ;
1143 025014  052777  000002  001712               BIS     #BIT1,@SMR      ; IF NOT,GOTO CACS
1144 025022  000137  024562                       JMP     P3              ; AND SELECT THE NEXT ONE
1145 025026  005737  027014        PEND3:     TST     DONE            ; ONE POLLING ADDRESS FOUND AT LEAST ?
1146 025032  001002                               BNE     7$              ; IF YES,GOTO NEXT PART OF THE TEST
```

```
1147 025034                                    ERROR   66.                     ;
1148 025040  052777  000002  001666  7$:      BIS     #BIT1 @SMR              ; GOTO CACS
1149 025046  112777  000031  001662           MOVB    #31,@IOR                ; DISABLE SERIAL POLL
1150 025054  052777  000004  001652           BIS     #BIT2,@SMR              ; GOTO STANDBY STATE,A TO LACS
1151 025062  000240                 DON3:     NOP                             ; JUST ENTRY-POINT
1152 025064                                    MCLEAR                          ;
1153 025100                                    CACS                            ;
1154 025142  013700  026770                    MOV     LIIND,R0                ; BUILD MLA B
1155 025146  063700  026764                    ADD     IECBAD,R0               ;       "
1156 025152  110077  001560                    MOVB    R0,@IOR                 ; SET B TO LADS
1157 025156                                    DATACC                          ;
1158 025174  013700  026766                    MOV     TAIND,R0                ; BUILD MTA A
1159 025200  063700  026762                    ADD     IECAAD,R0               ;       "
1160 025204  110077  001526                    MOVB    R0,@IOR                 ; SET A TO TADS
1161 025210                                    DATACC                          ;
1162 025226  052777  000004  001500           BIS     #BIT2,@SMR              ; GOTO STANDBY ,SET A TO TACS,B TO LACS
1163 025234  005037  027016         SPNEX:    CLR     NEXFLG                  ;
1164 025240  042777  177760  001476           BIC     #177760,@CSR            ; CLEAR INTERRUPT CAUSING BITS
1165 025246  012737  026722  000004           MOV     #NEXMEM,@#4             ;
1166 025254  052777  000060  001462           BIS     #60,@CSR                ;
1167 025262  012700  160000                    MOV     #160000,R0              ; FIRST LOC IN I/O PAGE
1168 025266  005710                 1$:       TST     (R0)                    ; 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$:      MOV     #6,@#4                  ;
1173 025306  010077  001436                    MOV     R0,@#BAR                ;
1174 025312  012777  000001  001426           MOV     #1,@#BCR                ; PREPARE BCR
1175 025320  052777  000101  001416           BIS     #101,@CSR               ; ENABLE INTERRUPT AND FUNCTION (DMA)
1176 025326  112777  000252  001402           MOVB    #252,@IOR               ; DUMMY VALUE TO OUTPUT REGISTER
1177 025334                                    DATACC                          ;
1178 025352  032777  000100  001364  3$:      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                                    DATACC                          ;
1186 025430  112777  000030  001300           MOVB    #30,@IOR                ; SERIAL POLL ENABLE
1187 025436  013700  026770                    MOV     LIIND,R0                ; BUILD MLA A
1188 025442  063700  026762                    ADD     IECAAD,R0               ;       "
1189 025446  110077  001264                    MOVB    R0,@IOR                 ; SET A TO LADS
1190 025452                                    DATACC                          ;
1191 025470  012701  000000                    MOV     #0,R1                   ; BUILD FIRST TALKER ADDRESS
1192 025474  013700  026766         P4:       MOV     TAIND,R0                ;
1193 025500  060100                           ADD     R1,R0                   ; BUILD TALKER ADDRESS
1194 025502  110077  001230                    MOVB    R0,@IOR                 ;
1195 025506                                    DATACC                          ;
1196 025524  052777  000004  001202           BIS     #BIT2,@SMR              ; GOTO STANDBY STATE,A TO LACS
1197 025532                                    STALL                           ;
1198 025562  122777  000000  001150           CMPB    #0,@IORHB               ; 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$                      ;
```

```
1204 025610                                        WRITE   MES21           ; GIVE INFORMATION ABOUT
1205 025620                                        DUMP    OCT,R1          ; ILLEGAL IEC-BUS ADDRESS
1206 025630                                        ERROR   67.             ;
1207 025634  117702  001100            3$:        MOVB    @IORHB,R2       ; SAVE STATUS BYTE
1208 025640  042702  177400                        BIC     #177400,R2      ;
1209 025644  022702  000102                        CMP     #102,R2         ; STATUS BYTE OK ?
1210 025650  001416                                BEQ     4$              ; IF YES,CONTINUE
1211 025652                                        ADP     #102,R2         ;
1212 025702                                        ERROR   68.             ;
1213 025706  012737  177777  027014    4$:        MOV     #177777,DONE    ; SET DONE FLAG
1214 025714  000411                                BR      PEND4           ; GOTO FINISH THIS TESTPART
1215 025716  005201                    5$:        INC     R1              ; GET NEXT TALKER ADDRESS
1216 025720  022701  000037                        CMP     #37,R1          ; ALL DONE ?
1217 025724  001405                                BEQ     PEND4           ;
1218 025726  052777  000002  001000                BIS     #BIT1,@SMR      ; IF NOT,GOTO CACS
1219 025734  000137  025474                        JMP     P4              ; AND SELECT THE NEXT ONE
1220 025740  005737  027014            PEND4:     TST     DONE            ; ONE POLLING ADDRESS FOUND AT LEAST ?
1221 025744  001002                                BNE     7$              ; IF YES,GOTO NEXT PART OF THE TEST
1222 025746                                        ERROR   69.             ;
1223 025752  052777  000002  000754    7$:        BIS     #BIT1,@SMR      ; GOTO CACS
1224 025760  112777  000031  000750                MOVB    #31,@IOR        ; DISABLE SERIAL POLL
1225 025766  052777  000004  000740                BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
1226 025774  000240            DON4:               NOP                     ; JUST ENTRY-POINT
1227 025776                                        MCLEAR                  ;
1228 026012                                        CACS                    ;
1229 026054  013700  026766                        MOV     TAIND,R0        ; BUILD MTA B
1230 026060  063700  026764                        ADD     IECBAD,R0       ;       "
1231 026064  110077  000646                        MOVB    R0,@IOR         ; SET B TO TADS
1232 026070                                        DATACC                  ;
1233 026106  013700  026770                        MOV     LIIND,R0        ; BUILD MLA A
1234 026112  063700  026762                        ADD     IECAAD,R0       ;       "
1235 026116  110077  000614                        MOVB    R0,@IOR         ; SET A TO LADS
1236 026122                                        DATACC                  ;
1237 026140  052777  000004  000566                BIS     #BIT2,@SMR      ; GOTO STANDBY STATE
1238 026146  104000                                SCOPE                   ;
1239 026150  042777  177400  000566    SPB10A:    BIC     #177400,@CSR    ; CLEAR INTERRUPT CAUSING BITS
1240 026156  013777  027002  000564                MOV     BUFBB,@BAR      ; FIRST BUFFER ADDRESS
1241 026164  013700  027002                        MOV     BUFBB,R0        ; PREPARE
1242 026170  112710  000000                        MOVB    #0,(R0)         ; TO TRANSFER "0"
1243 026174  012700  000001                        MOV     #1,R0           ; PREPARE BYTE COUNT REGISTER
1244 026200  005400                                NEG     R0              ;       "
1245 026202  010077  000540                        MOV     R0,@BCR         ;       "
1246 026206  013701  026774                        MOV     BUFAB,R1        ; POINTER TO BUFFER A
1247 026212  052777  000101  000524                BIS     #101,@CSR       ; ENABLE FUNCTION (DMA) AND INTERRUPT
1248 026220                                        DATACC                  ;
1249 026236  117721  000476                        MOVB    @IORHB,(R1)+    ; DATA TO BUFFER A
1250 026242  032777  000100  000474    1$:        BIT     #100,@CSR       ; HAS INTERRUPT OCCURRED ?
1251 026250  001374                                BNE     1$              ; IF YES,CONTINUE
1252 026252  052777  000400  000464                BIS     #BIT8,@CSR      ; REQUEST SERVICE
1253 026260  104000                                SCOPE                   ;
1254 026262  005037  027014            POLL5:     CLR     DONE            ; CLEAR DONE FLAG
1255 026266  052777  000002  000440                BIS     #BIT1,@SMR      ; GOTO CACS
1256 026274  112777  000030  000434                MOVB    #30,@IOR        ; SERIAL POLL ENABLE
1257 026302  012701  000000                        MOV     #0,R1           ; BUILD FIRST TALKER ADDRESS
1258 026306  013700  026766            P5:        MOV     TAIND,R0        ;
1259 026312  060100                                ADD     R1,R0           ; BUILD TALKER ADDRESS
1260 026314  110077  000416                        MOVB    R0,@IOR         ;
```

```
1261 026320                                    DATACC
1262 026336  052777  000004  000370            BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
1263 026344                                    STALL
1264 026374  122777  000000  000336            CMPB    #0,@IORHB       ; POLLING ADDRESS ?
1265 026402  001452                            BEQ     5$              ; IF NOT,TRY THE NEXT ONE
1266 026404  120137  026764                    CMPB    R1,IECBAD       ; LEGAL ADDRESS ?
1267 026410  001416                            BEQ     3$              ; IF YES,CONTINUE
1268 026412  032737  040000  007160            BIT     #BIT14,PSDSWR   ; PRINTOUT SELECTED ?
1269 026420  001012                            BNE     3$              ;
1270 026422                                    WRITE   MES21           ; GIVE INFORMATION ABOUT
1271 026432                                    DUMP    OCT,R1          ; ILLEGAL IEC-BUS ADDRESS
1272 026442                                    ERROR   70.             ;
1273 026446  117702  000366            3$:     MOVB    @IORHB,R2       ; SAVE STATUS BYTE
1274 026452  042702  177400                    BIC     #177400,R2      ;
1275 026456  022702  000104                    CMP     #104,R2         ; STATUS BYTE OK ?
1276 026462  001416                            BEQ     4$              ; IF YES,CONTINUE
1277 026464                                    ADD     #104,R2         ;
1278 026514                                    ERROR   71.             ;
1279 026520  012737  177777  027014  4$:       MOV     #177777,DONE    ; SET DONE FLAG
1280 026526  000411                            BR      PEND5           ; GOTO FINISH THIS TESTPART
1281 026530  005201                    5$:     INC     R1              ; GET NEXT TALKER ADDRESS
1282 026534  022701  000037                    CMP     #37,R1          ; ALL DONE ?
1283 026540  001405                            BEQ     PEND5           ;
1284 026542  052777  000002  000166            BIS     #BIT1,@SMR      ; IF NOT,GOTO CACS
1285 026550  000137  026306                    JMP     P5              ; AND SELECT THE NEXT ONE
1286 026556  005737  027014            PEND5:  TST     DONE            ; ONE POLLING ADDRESS FOUND AT LEAST ?
1287 026562  001002                            BNE     7$              ; IF YES,GOTO NEXT PART OF THE TEST
1288 026564                                    ERROR   72.             ;
1289 026570  052777  000002  000142  7$:       BIS     #BIT1,@SMR      ; GOTO CACS
1290 026576  112777  000031  000136            MOVB    #31,@IOR        ; DISABLE SERIAL POLL
1291 026604  052777  000004  000126            BIS     #BIT2,@SMR      ; GOTO STANDBY STATE,A TO LACS
1292 026612  000240                    DON5:   NOP                     ; JUST ENTRY POINT
1293 026614  005737  027024                    TST     ISTREP          ; TEST LOOP SELECTED ?
1294 026620  001402                            BEQ     8$              ; IF NOT,LEAVE THE TEST
1295 026622  000137  022260                    JMP     REP7            ;
1296 026626                            8$:     WRITE   MESEND          ; END OF TEST
1297 026636  000207                            RETURN
```

```
                                        .SBTTL  INTERRUPT SERVICE ROUTINES
                                ;
                                ;
                                ;
026634  052737  177777  027022  INTTST: BIS     #177777,INTFLG          ; SET INTERRUPT FLAG
026642  000002                  RTI                                     ;
                                ;
                                ;
                                ;
026644  000240                  INTSRV: NOP                             ; JUST ENTRY POINT
026646  000002                  RTI                                     ;
                                ;
                                ;
                                ;
026650                          PRIOR:  PSET    #0                      ; CLR PSW
026674  062706  000004                  ADD     #4,SP                   ; INCREASE SP BY 4
026700  000113                  JMP     (R3)                            ; GOTO PRINTOUT
                                ;
                                ;
                                ;
026702  032777  040000  000022  INTA:   BIT     #BIT14,@CIR             ; CAUSED SRQ THE INTERRUPT ?
026710  001003                  BNE     1$                              ;
026712  052777  000100  000012          BIS     #100,@CIR               ; REENABLE INTERRUPT
026720  000002                  1$:     RTI                             ;
                                ;
                                ;
                                ;
026722  052737  177777  027016  NEXMEM: BIS     #177777,NEXFLG          ; SET FLAG FOR NEX INDICATION
026730  000002                  RTI                                     ;
                                ;
                                ;
                                ;
```

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

```
   1336 026736  160014                IOR:    160014                ; IOR      "
```

```
1338 026740  160015              IORHB:  160015          ; IOR HIGH-BYTE ADDRESS DEFAULT VALUE
1339 026742  160016              VSR:    160016          ; VSR      "
1340 026744  160020              CSR:    160020          ; CSR      "
1341 026746  160022              BCR:    160022          ; BCR      "
1342 026750  160024              BAR:    160024          ; BAR      "
1343 026752  160026              MCR:    160026          ; MCR      "
1344 026754  160027              MCRHB:  160027          ; MCR HIGH-BYTE ADDRESS DEFAULT VALUE
1345 026756  000270              VECA:     270           ; DEFAULT OF IEC11-A VECTOR ADDRESS
1346 026760  000274              VECB:     274           ; DEFAULT OF IEC11-B VECTOR ADDRESS
1347 026762  000035              IECAAD:    35           ; IEC11-A BUS ADDRESS DEFAULT VALUE
1348 026764  000036              IECBAD:    36           ; IEC11-B BUS ADDRESS DEFAULT VALUE
1349 026766  000100              TAIND:    100           ; TALKER INDICATION
1350 026770  000040              LIIND:     40           ; LISTENER INDICATION
1351                             ;
1352                             ;
1353 026772  000000              MEMSAV: .WORD 0         ; MEMORY AREA FOR BUFFERS
1354 026774  000000              BUFAB:  .WORD 0         ; FIRST ADDRESS IN BUFFER A
1355 026776  000000              BUFAE:  .WORD 0         ; LAST ADDRESS IN BUFFER A
1356 027000  000000              BUFFL:  .WORD 0         ; LENGTH OF BUFFERS IN BYTES
1357 027002  000000              BUFBB:  .WORD 0         ; FIRST ADDRESS IN BUFFER B
1358 027004  000000              BUFBE:  .WORD 0         ; LAST ADDRESS IN BUFFER B
1359                             ;
1360                             ;
1361 027006  000000              CNT1:   .WORD 0         ; DELAY OR COUNTER
1362 027010  000000              CNT2:   .WORD 0         ; DELAY OR COUNTER
1363 027012  000000              CHAR:   .WORD 0         ; CHARACTER FOR MCR
1364 027014  000000              DONE:   .WORD 0         ;
1365 027016  000000              NEXFLG: .WORD 0         ; NEX-PROCEDURE (TEST3) TIMEOUT FLAG
1366 027020  000000              BCINP:  .WORD 0         ; REQUESTED BYTE COUNT FOR DMA
1367 027022  000000              INTFLG: .WORD 0         ; INTERRUPT FLAG
1368 027024  000000              TSTREP: .WORD 0         ; FLAG FOR LOOP ON SINGLE TESTS
1369 027026  000000              ASAVE:  .WORD 0         ;
1370 027030  000000              BSAVE:  .WORD 0         ;
```

```
1372                                            .SBTTL  MESSAGES
1373                                            .NLIST  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 SDC 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
```

```
                                        .SBTTL  STANDARD CSS REGISTER TEST
                                        ;       GR001           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,#CSR
                                        ;        CALL   REGTST
                                        ;
                                        ;
                                        ;
031100                          REGTST: PUSH    R0,R1,R2,R3             ;  GR001
031110  017601  000012                  MOV     @12(SP),R1              ; GET ADDRESS OF REGISTER;GR001
031114  016602  000014                  MOV     14(SP),R2               ; GET BITMASK ;GR001
031120  104000                  1$:     SCOPE                           ; PREPARE SCOPE LOOP
031122  010211                          MOV     R2,@R1                  ; TRY TO SET ALL BITS
031124  011100                          MOV     @R1,R0                  ; TRY TO READ IT BACK
031126  005102                          COM     R2                      ; BUILD MASK COMPLEMENT
031130  040200                          BIC     R2,R0                   ; CLEAR UNUSED BITS
031132  005102                          COM     R2                      ; REBUILD MASK
031134  020200                          CMP     R2,R0                   ; ALL BITS SET ?
031136  001432                          BEQ     2$                      ; BRANCH IF YES
031140                                  TST14   PSDSWR                  ; IF ADP AFTER ERROR IS NOT ALLOWED,
031146  001010                          BNE     5$                      ; GOTO 5$,ELSE WRITE
031150                                  WRITE   RG$MS$                  ; WRITE AT REGISTER NR:
031160                                  DUMP    OCT,R1                  ; DMP REGISTER NUMBER
031170                          5$:     ADP     R2,R0
031216                                  ERROR   255.                    ; NOT ALL BITS SET
031222  000736                          BR      1$                      ; RESTART TEST
031224  104000                  2$:     SCOPE                           ; SAVE SCOPE LOOP ADRESS
031226  040211                          BIC     R2,@R1                  ; CLEAR BITS UNDER MASK
031230  011100                          MOV     @R1,R0                  ; READ BITS
031232  005102                          COM     R2                      ; BUILD MASK COMPLEMENT
031234  040200                          BIC     R2,R0                   ; MASK OUT UNUSED BITS
031236  005102                          COM     R2                      ; REBUILD MASK
031240  005700                          TST     R0                      ; ARE ALL BITS RESET
031242  001433                          BEQ     3$                      ; BRANCH IF YES
031244                                  TST14   PSDSWR                  ; IF ADP IS NOT ALLOWED
031252  001010                          BNE     6$                      ; GOTO 6$
031254                                  WRITE   RG$MS$                  ; AT REGISTER
```

```
1468 031264                                    DUMP    OCT,R1          ; DMP REGISTER NR
1469 031274                            6$:     ADP     #0,R0
1470 031324                                    ERROR   254.            ; NOT ALL BITS RESET
1471 031330  000673                            BR      1$              ; RESTART TEST
1472 031332  005102                    3$:     COM     R2              ; BUILD COMPLEMENT OF MASK
1473 031334  004737  031506                    CALL    RANDOM          ; GET A RANDOM #
1474 031340  040200                            BIC     R2,R0           ; MASK OUT UNUSED BITS
1475 031342  104000                            SCOPE                   ; SAVE SCOPE LOOP ADDRESS
1476 031344  010011                            MOV     R0,(R1)         ; TRY TO SET RANDOM BITS
1477 031346  011103                            MOV     @R1,R3          ; READ IT BACK  ; GR001
1478 031350  040203                            BIC     R2,R3           ; CLEAR UNUSED BITS     ; GR001
1479 031352  020300                            CMP     R3,R0           ; ALL BITS OK ? ; GR001
1480 031354  001432                            BEQ     4$              ; 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$:     COM     R2              ; REBUILD MASK
1489 031444                                    $$LOOP  1000,1$
1490 031466                                    POP     R3,R2,R1,R0     ;  GR001
1491 031476  062706  000006                    ADD     #6,SP           ; REBUILD STACK
1492 031502  000176  177772                    JMP     @-6(SP)         ; RETURN TO CALLER
1493                                    ;
1494                                    ;       RANDOM NUMBER GENERATOR USED FOR REGTST
1495                                    ;
1496 031506  013746  031544            RANDOM: MOV     R4,-(SP)
1497 031512  013700  031546                    MOV     R5,R0
1498 031516  006316                            ASL     @SP
1499 031520  005500                            ADC     R0
1500 031522  006200                            ASR     R0
1501 031524  005516                            ADC     @SP
1502 031526  061600                            ADD     @SP,R0
1503 031530  005600                            SBC     R0
1504 031532  012637  031544                    MOV     (SP)+,R4
1505 031536  010037  031546                    MOV     R0,R5
1506 031542  000207                            RETURN
1507 031544  135753                    R4:     135753
1508 031546  024624                    R5:     024624
1509                                            .NLIST  BEX
1510 031550     133     102     111    RG$MS$: .ASCII  /[BIT MASK OF R AND W BITS IN REGISTER NR: ]/
1511                                            .LIST   BEX
1512                                            .EVEN
1513 031624                            PRGEND:
1514         001204                             .END    START
```

SYMBOL TABLE

```
ACC4     020040          D..TYP  006420          MERROR  007652          NOR.ER  004376          SAV     011312
ADDRIN   011316          EMTTBL  002416          MERRP2  007662          NOR..E  004372          SCOPAD  007172
ASAVE    027026          EMTTND  002424          MER1    010304          NO6     010612          SCOPE = 104000
BAR      026750          EMT0    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          SISDRO= 172200
BCOVFL   016216          ERRFLG  007154          MES10   030123          PEND2   024150          SMR     026734
BCR      026746          IAD..E  004302          MES11   030205          PEND3   025026          SPBTOA  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          SR0   = 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          PRGEND  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          KDSARO= 172360          MILADR  007445          PR1     017050          TKS   = 177560
BUFAE    026776          KDSDRO= 172320          MILBIN  007360          PR2     017126          TPB   = 177566
BUFBB    027002          KISARO= 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
COM..E   010614          KISDRO= 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          UDSARO= 177660
CMODE  = 140000          LAB1    013340          MNOSUB  010032          P3      024562          UDSDRO= 177620
CMP4     020276          LAB11   012042          MPWRFL  007704          P4      025474          UISARO= 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          UISDRO= 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
DON0     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          RG$MS$  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
```

K10

SYMBOL TABLE

```
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,CDM,ZIECAO
.
```