

IBV11-A

IBV11A DIAG
CNIBAAO

AH-T462A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 82-83
MADE IN USA



Table with multiple columns and rows of technical data, including various alphanumeric codes and symbols.



IDENTIFICATION

PRODUCT NAME: CNIBAA0 IBV11A DIAG
PRODUCT CODE: AC-T461A-MC
PRODUCT DATE: DECEMBER, 1982
MAINTAINER: DIAGNOSTICS SERVICES/ISS
AUTHOR: E. BADGER

COPYRIGHT (C) 1982, 1983
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

5675
 5676
 5677
 5678
 5684 167400
 5685
 5686
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 5687 000001
 5688
 5689
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 5690
 5691
 5692
 5693 000000
 5694
 5695
 5696
 5697
 5698
 5708 000004
 5709 000004 012066 000200
 5710 000174 000000
 5711 000174 000000
 5712 000176 000000
 5713 000100 000100
 5714 000100 000104 000200 000002
 5715 000200 000200
 5716 000200 000137 001422
 5717
 5718
 5719
 (1)
 (1)
 (1) 001100
 (1)
 (1)

```

.NLIST MC,MD,CND
.LIST ME
.ENABL ABS
.ENABL AMA
$SWR=i67400

.TITLE MAINDEC-11-CNIBA-A
;*COPYRIGHT (C) 1982
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY EDWARD C. BADGER
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
$TN=1
;THIS VERSION LAST EDITED - DEC, 1982

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
SWITCH USE
-----
15 HALT ON ERROR
14 LOOP ON TEST
13 INHIBIT ERROR TYPEOUTS
11 INHIBIT ITERATIONS
10 BELL ON ERROR
9 LOOP ON ERROR
8 LOOP ON TEST IN SWR<7:0>

.SBTTL TRAP CATCHER
.=0
;*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2"
;*AND "JSR PC,RO" SEQUENCE TO CATCH ILLEGAL INTERRUPTS,
;*AND INTERRUPTS TO THE WRONG VECTOR.
;*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
;*VECTORS
.=4
.WORD IOTRD,200 ;HANDLE BUSS ERROR.
.=174
DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER.
SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER.
.=100
.WORD 104,200,2 ;IF 'B EVENT' ON Q-BUS IS
;CONNECTED, WE NEED A WAY OF
;IGNORING ITS INTERRUPTS.
.=200
JMP START

.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
  
```



```

(1) .EQUIV SW06,SW6
(1) .EQUIV SW05,SW5
(1) .EQUIV SW04,SW4
(1) .EQUIV SW03,SW3
(1) .EQUIV SW02,SW2
(1) .EQUIV SW01,SW1
(1) .EQUIV SW00,SW0

```

```

(1) ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

```

```

(1) 100000 BIT15= 100000
(1) 040000 BIT14= 40000
(1) 020000 BIT13= 20000
(1) 010000 BIT12= 10000
(1) 004000 BIT11= 4000
(1) 002000 BIT10= 2000
(1) 001000 BIT09= 1000
(1) 000400 BIT08= 400
(1) 000200 BIT07= 200
(1) 000100 BIT06= 100
(1) 000040 BIT05= 40
(1) 000020 BIT04= 20
(1) 000010 BIT03= 10
(1) 000004 BIT02= 4
(1) 000002 BIT01= 2
(1) 000001 BIT00= 1
(1) .EQUIV BIT09,BIT9
(1) .EQUIV BIT08,BIT8
(1) .EQUIV BIT07,BIT7
(1) .EQUIV BIT06,BIT6
(1) .EQUIV BIT05,BIT5
(1) .EQUIV BIT04,BIT4
(1) .EQUIV BIT03,BIT3
(1) .EQUIV BIT02,BIT2
(1) .EQUIV BIT01,BIT1
(1) .EQUIV BIT00,BIT0

```

```

(1) ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1) 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(1) 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
(1) 000014 TBITVEC=14 ;: "T" BIT
(1) 000014 TRTVEC= 14 ;:TRACE TRAP
(1) 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
(1) 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1) 000024 PWRVEC= 24 ;:POWER FAIL
(1) 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
(1) 000034 TRAPVEC=34 ;: "TRAP" TRAP
(1) 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
(1) 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
(1) ;***** THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(1) 000100 LKVEC= 100 ;:LINE CLOCK VECTOR
(1) 000140 BRKVEC= 140 ;:BREAK VECTOR
(1) 000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR

```

```

5720
5721 174100 ABASE= 174100 ;VER:0
5722 000320 AVECT1= 320 ;VER:0
5723 000200 APRIOR= 200

```

```

5724          000001          $TN=1
5725
5726          .SBTTL ACT11 HOOKS
(1)
(2)          ::*****
(1)          :HOOKS REQUIRED BY ACT11
(1)          000204          $SVPC=.          ;SAVE PC
(1)          000046          .=46
(1) 000046 007052          $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
(1)          000052          .=52
(1) 000052 000000          .WORD 0          ;;2)SET LOC.52 TO ZERO
(1)          000204          .=$SVPC          ;; RESTORE PC
5727
5728          001000          .=1000
5729
5730          .SBTTL APT PARAMETER BLOCK
(1)
(2)          ::*****
(1)          :SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)          ::*****
(1)          001000          .$X=.          ;;SAVE CURRENT LOCATION
(1)          000024          .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1) 000024 000200          200          ;;FOR APT START UP
(1)          000044          .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1) 000044 001000          $APTHDR          ;;POINT TO APT HEADER BLOCK
(1)          001000          .=$X          ;;RESET LOCATION COUNTER
(2)          ::*****
(1)          :SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)          :INTERFACE SPEC.
(1)
(1) 001000          $APTHD:
(1) 0C1000 000000          $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1) 001002 001174          $MBADR: .WORD $MAIL          ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1) 001004 000074          $TSTM: .WORD 60.          ;;RUN TIM OF LONGEST TEST
(1) 001006 000170          $PASTM: .WORD 120.          ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1) 001010 000170          $UNITM: .WORD 120.          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1) 001012 000031          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
5731
  
```


| | | | | | |
|-----|--------|--------|----------------|-----------|--|
| (2) | 001210 | 000000 | \$MSGAD: .WORD | AMSGAD | ::MESSAGE ADDRESS |
| (2) | 001212 | 000000 | \$MSGLG: .WORD | AMSLG | ::MESSAGE LENGTH |
| (2) | 001214 | | \$ETABLE: | | ::APT ENVIRONMENT TABLE |
| (2) | 001214 | 000 | \$ENV: .BYTE | AENV | ::ENVIRONMENT BYTE |
| (2) | 001215 | 000 | \$ENVM: .BYTE | AENVM | |
| (2) | | | ::ENVIRONMENT | MODE BITS | |
| (2) | 001216 | 000000 | \$SWREG: .WORD | ASWREG | ::APT SWITCH REGISTER |
| (2) | 001220 | 000000 | \$USWR: .WORD | AUSWR | ::USER SWITCHES |
| (2) | 001222 | 000000 | \$CPUOP: .WORD | ACPUOP | ::CPU TYPE, OPTIONS |
| (2) | | | ::* | | BITS 15-11=CPU TYPE |
| (2) | | | ::* | | 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05 |
| (2) | | | ::* | | 11/70=06,PDC=07,Q=10 |
| (2) | | | ::* | | BIT 10=REAL TIME CLOCK |
| (2) | | | ::* | | BIT 9=FLOATING POINT PROCESSOR |
| (2) | | | ::* | | BIT 8=MEMORY MANAGEMENT |
| (2) | 001224 | 000 | \$MAMS1: .BYTE | AMAMS1 | ::HIGH ADDRESS,M.S. BYTE |
| (2) | 001225 | 000 | \$MTYP1: .BYTE | AMTYP1 | ::MEM. TYPE,BLK#1 |
| (2) | | | ::* | | MEM.TYPE BYTE -- (HIGH BYTE) |
| (2) | | | ::* | | 900 NSEC CORE=001 |
| (2) | | | ::* | | 300 NSEC BIPOLAR=002 |
| (2) | | | ::* | | 500 NSEC MOS=003 |
| (2) | 001226 | 000000 | \$MADR1: .WORD | AMADR1 | ::HIGH ADDRESS,BLK#1 |
| (2) | | | ::* | | MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE |
| (2) | 001230 | 000 | \$MAMS2: .BYTE | AMAMS2 | ::HIGH ADDRESS,M.S. BYTE |
| (2) | 001231 | 000 | \$MTYP2: .BYTE | AMTYP2 | ::MEM. TYPE,BLK#2 |
| (2) | 001232 | 000000 | \$MADR2: .WORD | AMADR2 | ::MEM.LAST ADDRESS,BLK#2 |
| (2) | 001234 | 000 | \$MAMS3: .BYTE | AMAMS3 | ::HIGH ADDRESS,M.S.BYTE |
| (2) | 001235 | 000 | \$MTYP3: .BYTE | AMTYP3 | ::MEM. TYPE,BLK#3 |
| (2) | 001236 | 000000 | \$MADR3: .WORD | AMADR3 | ::MEM.LAST ADDRESS,BLK#3 |
| (2) | 001240 | 000 | \$MAMS4: .BYTE | AMAMS4 | ::HIGH ADDRESS,M.S.BYTE |
| (2) | 001241 | 000 | \$MTYP4: .BYTE | AMTYP4 | ::MEM. TYPE,BLK#4 |
| (2) | 001242 | 000000 | \$MADR4: .WORD | AMADR4 | ::MEM.LAST ADDRESS,BLK#4 |
| (2) | 001244 | 000320 | \$VECT1: .WORD | AVECT1 | ::INTERRUPT VECTOR#1,BUS PRIORITY#1 |
| (2) | 001246 | 000000 | \$VECT2: .WORD | AVECT2 | ::INTERRUPT VECTOR#2BUS PRIORITY#2 |
| (2) | 001250 | 174100 | \$BASE: .WORD | ABASE | |
| (2) | | | ::BASE | ADDRESS | OF EQUIPMENT UNDER TEST |
| (2) | 001252 | 000000 | \$DEV: .WORD | ADEV | ::DEVICE MAP |
| (2) | 001254 | 000000 | \$CDW1: .WORD | ACDW1 | ::CONTROLLER DESCRIPTION WORD#1 |
| (2) | 001256 | | \$ETEND: | | |
| (2) | | | .MEXIT | | |


```

(1) .SBTTL ERROR POINTER TABLE
(1)
(1) : *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1) : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1) : *LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1) : *NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1) : *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1) : * EM ::POINTS TO THE ERROR MESSAGE
(1) : * DH ::POINTS TO THE DATA HEADER
(1) : * DT ::POINTS TO THE DATA
(1) : * DF ::POINTS TO THE DATA FORMAT
(1)
(1) $ERRTB:
(1) 001256
5733
5737
5738 :ITEM 1
5739
5740 001256 012216 EM1 :IBS FUNCTION ERROR
5741 001260 012437 DH1 :TEST ERRPC IB ADDR
5742 001262 012640 DT1 :$TESTN,$ERRPC,IBS
5743 001264 012710 DFO :ALL NUMBERS ARE IN OCTAL FORM.
5744
5745 :ITEM 2
5746
5747 001266 012244 EM2 :IBD FUNCTION ERROR
5748 001270 012437 DH1 :TEST ERRPC IB ADDR
5749 001272 012640 DT1 :$TESTN,$ERRPC,IBS
5750 001274 012710 DFO :ALL NUMBERS ARE IN OCTAL FORM.
5751
5752 :ITEM 3
5753
5754 001276 012272 EM3 :IBS DATA ERROR
5755 001300 012505 DH3 :TEST ERRPC GOOD BAD
5756 001302 012654 DT3 :$TESTN,$ERRPC,$GDDAT,$BDDAT
5757 001304 012710 DFO :ALL NUMBERS ARE IN OCTAL FORM.
5758
5759 :ITEM 4
5760
5761 001306 012314 EM4 :IBD DATA ERROR
5762 001310 012505 DH3 :TEST ERRPC GOOD BAD
5763 001312 012654 DT3 :$TESTN,$ERRPC,$GDDAT,$BDDAT
5764 001314 012710 DFO :ALL NUMBERS ARE IN OCTAL FORM.
5765
5766 :ITEM 5
5767
5768 001316 012336 EM5 :IBS/IBD ADDRESS ERROR
5769 001320 012542 DH5 :TEST ERROR PC ADDRESS
5770 001322 012666 DT5 :$STNM,$ERRPC,IBS
5771 001324 012710 DFO :ALL NUMBERS ARE IN OCTAL FORM.
5772
5773 :ITEM 6
5774
5775 001326 012366 EM6 :IBWC/IBCA DATA ERROR
5776 001330 012505 DH3 :TEST ERRPC GOOD BAD
  
```

5777 001332 012654
5778 001334 012710
5779
5780
5781 001336 012415
5782 001340 012573
5783 001342 012676
5784 001344 012710
5785
5800
5811
5815
5823
5829
5849
5850
5851
5852
5853

DT3
DF0
:ITEM 7
EM7
DH7
DT7
DF0

:\$TESTN,\$ERRPC,\$GDDAT,\$BDDAT
:ALL NUMBERS ARE IN OCTAL FORM.
:INTERRUPT ERROR
:TEST ERRPC TO FROM ADDR.
:TSTNM,#ERRPC,TRTO,TRFRO
:ALL NUMBERS ARE IN OCTAL FORM.

5854 001346 174100
5855 001350 174102
5856 001352 174104
5857 001354 174106
5858 001356 000320
5859 001360 000324
5860 001362 000330
5861 001364 000334
5862 001366 174110
5863 001370 174112
5864 001372 000340
5865 001374 000344
5866 001376 000350
5867 001400 000354
5868
5869
5870
5871 001402 000322
5872 001404 000326
5873 001406 000332
5874 001410 000336
5875
5876 001412 000342
5877 001414 000346
5878 001416 000352
5879 001420 000356
5880
5881
5882
5883
5885
5886 001422

.SBTTL REG ADDRESS AND COMMON TAGS
:WARNING IF DEVICE # IS AT DIFFERENT ADDRESS OR VECTOR
:DO NOT PATCH THESE LOCATIONS - SEE PROGRAM DOCUMENTATION.
IBS: .WORD ABASE ;>NO <:CONTROL AND STATUS REGISTER.
IBD: .WORD ABASE+2 ;>PATCHES <:DATA REGISTER.
IBWC: .WORD ABASE+4 ;ADDRESS RESERVED FOR
IBCA: .WORD ABASE+6 ;FUTURE USE
VECTA: .WORD AVECT1 ;>ALLOWED <:VECTOR ADDRESS.
VECTB: .WORD AVECT1+4 ;>HERE! <:VECTOR ADDR. +4.
VECTC: .WORD AVECT1+10
VECTD: .WORD AVECT1+14
IBS2: .WORD ABASE+10
IBD2: .WORD ABASE+12
VECTA2: .WORD AVECT1+20
VECTB2: .WORD AVECT1+24
VECTC2: .WORD AVECT1+30
VECTD2: .WORD AVECT1+34

:VECTOR ADDRESSES +2 LOCATIONS.

PRA: .WORD AVECT1+2 ;NOTE: DO NOT ATTEMPT TO PATCH
PRB: .WORD AVECT1+6 ; THESE LOCATIONS IF A VECTOR
PRC: .WORD AVECT1+12 ; VARYIES . ALTER LOCATION
PRD: .WORD AVECT1+16 ; '\$VECT1:'.
PRA2: .WORD AVECT1+22 ; IF TEST MODULE VECTOR IS
PRB2: .WORD AVECT1+26 ; DIFFERENT, YOU MUST CHANGE
PRC2: .WORD AVECT1+32 ; LOCATION 'VECTA2:'.
PRD2: .WORD AVECT1+36 ;

.SBTTL PROGRAM START

(1)
(1)
(1) 001422 012706 001100
(1) 001426 005026
(1) 001430 022706 001140

START:
:CLEAR THE COMMON TAGS (\$CMTAG) AREA
MOV # \$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;:CLEAR MEMORY LOCATION
CMP #SWR,R6 ;:DONE?

```

(1) 001434 001374          BNE      -6          ;;LOOP BACK IF NO
(1) 001436 012706 001100  MOV      #STACK,SP   ;;SETUP THE STACK POINTER
(1)          ;;INITIALIZE A FEW VECTORS
(1) 001442 012737 007450 000020  MOV      #$$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1) 001450 012737 000300 000022  MOV      #PR6,@#IOTVEC+2 ;;LEVEL 6
(1) 001456 012737 007126 000030  MOV      #$$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1) 001464 012737 000300 000032  MOV      #PR6,@#EMTVEC+2 ;;LEVEL 6
(1)          ;;BIT02
(1) 001472 012737 012136 000034  MOV      #$$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1) 001500 012737 000300 000036  MOV      #PR6,@#TRAPVEC+2;LEVEL 6
(1) 001506 012737 011710 000024  MOV      #$$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
(1) 001514 012737 000300 000026  MOV      #PR6,@#PWRVEC+2 ;;LEVEL 6
(1) 001522 005037 001160          CLR      $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
(1) 001526 005037 001162          CLR      $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1) 001532 112737 000001 001115  MOV     #1,$ERMAX     ;;ALLOW ONE ERROR PER TEST
(1) 001540 012737 001540 001106  MOV     #.,$LPADR     ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1) 001546 012737 001546 001110  MOV     #.,$LPERR     ;;SETUP THE ERROR LOOP ADDRESS
(2)          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
(2) 001554 013746 000004          MOV     @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
(2) 001560 012737 001614 000004  MOV     #64,$@#ERRVEC ;;SET UP ERROR VECTOR
(2) 001566 012737 177570 001140  MOV     #DSWR,SWR     ;;SETUP FOR A HARDWARE SWICH REGISTER
(2) 001574 012737 177570 001142  MOV     #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2) 001602 022777 177777 177330  CMP     #-1,@SWR     ;;TRY TO REFERENCE HARDWARE SWR
(2) 001610 001012          BNE     66$         ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)          ;;AND THE HARDWARE SWR IS NOT = -1
(2) 001612 000403          BR     65$         ;;BRANCH IF NO TIMEOUT
(2) 001614 012716 001622          64$: MOV     #65$,(SP)   ;;SET UP FOR TRAP RETURN
(2) 001620 000002          RTI
(2) 001622 012737 000176 001140  65$: MOV     #SWREG,SWR ;;POINT TO SOFTWARE SWR
(2) 001630 012737 000174 001142  MOV     #DISPREG,DISPLAY
(2) 001636 012637 000004          66$: MOV     (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
(1)
(2) 001642 005037 001202          CLR     $PASS        ;;CLEAR PASS COUNT
(2) 001646 132737 000200 001215  BITB   #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2) 001654 001403          BEQ    67$         ;;YES,USE NON-APT SWITCH
(2) 001656 012737 001216 001140  MOV     #$$SWREG,SWR ;;NO,USE APT SWITCH REGISTER
(2) 001664          67$:
5887 001664 012737 012066 000004  MOV     #IOTRD,ERRVEC ;;SET TO HANDLE BUS ERRORS.
5888 001672 012737 000200 000006  MOV     #200,ERRVEC+2
5889
5890 001700 013737 001250 001346  MOV     $BASE,IBS    ;;GET BASE ADDR.
5891 001706 013737 001346 001350  MOV     IBS,IBD      ;;FIX DATA BUFFER=
5892 001714 062737 000002 001350  ADD     #2,IBD       ;;CSR+2
5893 001722 013737 001350 001352  MOV     IBD,IBWC
5894 001730 062737 000002 001352  ADD     #2,IBWC
5895 001736 013737 001352 001354  MOV     IBWC,IBCA
5896 001744 062737 000002 001354  ADD     #2,IBCA
5897 001752 013737 001244 001356  MOV     $VECT1,VECTA ;;GET VECTOR ADDR.
5898 001760 042737 170000 001356  BIC     #170000,VECTA ;;STRIP JUNK
5899 001766 013737 001346 012646  MOV     IBS,IBSA
5900 001774 013737 001350 012650  MOV     IBD,IBDA
5901 002002 013737 001366 001370  MOV     IBS2,IBD2
5902 002010 062737 000002 001370  ADD     #2,IBD2
5903 002016 013737 001356 001360  MOV     VECTA,VECTB
5904 002024 062737 000004 001360  ADD     #4,VECTB

```

```

5905 002032 013737 001360 001362      MOV      VECTB,VECTC
5906 002040 062737 000004 001362      ADD      #4,VECTC
5907 002046 013737 001362 001364      MOV      VECTC,VECTD
5908 002054 062737 000004 001364      ADD      #4,VECTD
5909 002062 013737 001372 001374      MOV      VECTA2,VECTB2
5910 002070 062737 000004 001374      ADD      #4,VECTB2
5911 002076 013737 001374 001376      MOV      VECTB2,VECTC2
5912 002104 062737 000004 001376      ADD      #4,VECTC2
5913 002112 013737 001376 001400      MOV      VECTC2,VECTD2
5914 002120 062737 000004 001400      ADD      #4,VECTD2
5915
5916 002126 013737 001356 001402      MOV      VECTA,PRA          ;SET UP VECTOR+2 ADDRESSES.
5917 002134 062737 000002 001402      ADD      #2,PRA
5918 002142 013737 001402 001404      MOV      PRA,PRB
5919 002150 062737 000004 001404      ADD      #4,PRB
5920 002156 013737 001404 001406      MOV      PRB,PRC
5921 002164 062737 000004 001406      ADD      #4,PRC
5922 002172 013737 001406 001410      MOV      PRC,PRD
5923 002200 062737 000004 001410      ADD      #4,PRD
5924 002206 013737 001372 001412      MOV      VECTA2,PRA2
5925 002214 062737 000002 001412      ADD      #2,PRA2
5926 002222 013737 001412 001414      MOV      PRA2,PRB2
5927 002230 062737 000004 001414      ADD      #4,PRB2
5928 002236 013737 001414 001416      MOV      PRB2,PRC2
5929 002244 062737 000004 001416      ADD      #4,PRC2
5930 002252 013737 001416 001420      MOV      PRC2,PRD2
5931 002260 062737 000004 001420      ADD      #4,PRD2
5932 002266
5933
(1) 002266 013777 001402 177062      MOV      PRA,@VECTA ;/RESTORE VECTOR FOR
(1) 002274 012777 004700 177100      MOV      #4700,@PRA ;/ILLEGAL INTRO.
5934
(1) 002302 013777 001404 177050      MOV      PRB,@VECTB ;/RESTORE VECTOR FOR
(1) 002310 012777 004700 177066      MOV      #4700,@PRB ;/ILLEGAL INTRO.
5935
(1) 002316 013777 001406 177036      MOV      PRC,@VECTC ;/RESTORE VECTOR FOR
(1) 002324 012777 004700 177054      MOV      #4700,@PRC ;/ILLEGAL INTRO.
5936
(1) 002332 013777 001410 177024      MOV      PRD,@VECTD ;/RESTORE VECTOR FOR
(1) 002340 012777 004700 177042      MOV      #4700,@PRD ;/ILLEGAL INTRO.
5937
(1)
(1) 002346 005227 177777      .SBTTL TYPE PROGRAM NAME
(1) 002352 001033      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1) 002354 104401 002422      INC # -1 ;:FIRST TIME?
(2) 002360 005737 000042      BNE 64$ ;:BRANCH IF NO
(2) 002364 001012      TYPE 65$ ;:TYPE ASCIZ STRING
(2) 002366 123727 001214 000001      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
(2) 002374 001406      TST @#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
(2) 002376 023727 001140 000176      BNE 66$ ;:BRANCH IF YES
(2) 002404 001005      CMPB $ENV,#1 ;:ARE WE RUNNING UNDER APT?
(2) 002406 104406      BEQ 66$ ;:BRANCH IF YES
(2) 002410 000403      CMP SWR,#SWREG ;:SOFTWARE SWITCH REG SELECTED?
(2) 002412 112737 000001 001134 66$: BR 67$ ;:BRANCH IF NO
(2) 002420      GTSWR ;:GET SOFT-SWR SETTINGS
      BR 67$
      MOVB #1,$AUTOB ;:SET AUTO-MODE INDICATOR

```

```

(1) 002420 000410 BR 64$ ::GET OVER THE ASCIZ
(1) 65$: .ASCIZ <CRLF>#MD11-CNIBA-A#<CRLF>
(1) 002442 64$: RESET
5938 002442 000005
5939
5940
(3) *****
(3) *TEST 1 *TEST THE ADDRESSABILITY OF THE IBS, IBD REGISTERS
(2) 002444 000240 TST1: NOP
(2)
(1) 002446 012737 000050 001160 MOV #50,$TIMES ::DO 50 ITERATIONS
(1) 002454 012737 002504 001106 MOV #1$,$LPADR ::SET SCOPE LOOP ADDRESS
5941
5942 002462 012737 000001 001102 MOV #1,$STSTNM ;SET TEST #1.
5943 002470 012737 000001 001200 MOV #1,$TESTN ;DON'T FORGET APT!
5944 002476 012737 002504 001110 MOV #1$,$LPERR
5945
5946 002504 013746 000004 1$: MOV ERRVEC,-(SP) ;SAVE CONTENTS OF ADDR. 4
5947 002510 012737 002536 000004 MOV #2$,ERRVEC ;SET TIME-OUT TRAP VECTOR TO HANDLE
5948 ;IN CASE WE TIME OUT WHEN
5949 ;WE ADDR. THE IBV-11
5950
5951 002516 005777 176624 TST @IBS ;ADDR THE IBS, IF NO RESPONSE,
5952 ;WILL TRAP TO 2$ FROM HERE
5953 002522 012737 002544 000004 MOV #3$,ERRVEC ;CHANGE FOR ADDRESSING THE IBD REG.
5954
5955 002530 005777 176614 TST @IBD ;ADDR THE IBD REG.
5956 ;WE'LL TRAP TO 3$ FROM HERE IF BAD.
5957 002534 000406 BR 4$
5958 002536 2$:
(1) 002536 062706 000004 ADD #4,$SP ;/ADD #4 TO STACK POINTER.
5959

::$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$

(1)
(1) 002542 104005 ERROR 5 ;/MODULE FAULT DETECTED:
5960 ;IBS REGISTER COULD NOT BE
5961 ;ADDRESSED

::$$$$$$$$$$$^^^ ERROR ^^ $$$$$$$$$$$$

5963 002544 3$:
(1) 002544 062706 000004 ADD #4,$SP ;/ADD #4 TO STACK POINTER.
5964

::$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$

(1)
(1) 002550 104005 ERROR 5 ;/MODULE FAULT DETECTED:
5965 ;ADDRESSED

::$$$$$$$$$$$^^^ ERROR ^^ $$$$$$$$$$$$

5967 002552 012637 000004 4$: MOV (SP)+,ERRVEC ;RESTORE CONTENTS OF LOC 4.
5968 ;/PR
(1) 002556 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 002562 012746 002570 MOV #64$,-(SP) ;/SHOW RETURN ADDRESS
  
```

```

(1) 002566 000002
(1) 002570
5969
5977
5978
(3)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 002570 000004
(2)
(1) 002572 012737 000010 001160
5979
5980 002600 013746 000004
5981 002604 012737 002632 000004
5982
5983
5984 002612 005777 176534
5985
5986
5987 002616 012737 002642 000004
5988
5989 002624 005777 176524
5990 002630 000407
5991
5992 002632
(1) 002632 062706 000004
5993

        RTI                ;/CAUSE A RETURN (PUTS NEW STATUS
64$:                ;/IN STATUS REG.)

::*****
:*TEST 2          *TEST THAT BASE ADDRESSES +4,+6 RESPOND WHEN ADDRESSED
:
:*
:*EVEN THOUGH THE BASE ADDRESS +4 AND +6 ARE NOT USED,
:*THE IBV11A SHOULD RESPOND TO THEM
:*
::*****
TST2:  SCOPE
        MOV      #10,$TIMES      ;;DO 10 ITERATIONS
        MOV      ERRVEC,-(SP)    ;SAVE CONTENTS OF ADDR 4.
        MOV      #1$,ERRVEC     ;SET TIME OUT TRAP VECTOR TO HANDLE
                                ;IN CASE WE TIME OUT WHEN WE
                                ;ADDRESS THE IBV-11 ADDRESSES +4,+6.
        TST      @IBWC          ;TEST BASE ADDRESS +4, IF NO RESPONSE
                                ;WILL TRAP TO 1$ FROM HERE
        MOV      #2$,ERRVEC     ;CHANGE FOR ADDRESSING +6 ADDR.
        TST      @IBCA          ;ADDR THE +6 ADDR. - TRAP IF BAD.
        BR       3$            ;CONTINUE IF GOOD.

1$:      ADD      #4,SP          ;/ADD #4 TO STACK POINTER.

        ;:$$$$$$$$$>>> ERROR <<<$$$$$$$$$
        ERROR    5              ;/MODULE FAULT DETECTED:
                                ;BASE ADDR+4 COULD NOT
                                ;BE ADDRESSED.

        ;:$$$$$$$$$^^^ ERROR ^^^$$$$$$$$$
        BR       3$

2$:      ADD      #4,SP          ;/ADD #4 TO STACK POINTER.

        ;:$$$$$$$$$>>> ERROR <<<$$$$$$$$$
        ERROR    5              ;/MODULE FAULT DETECTED:
                                ;BASE ADDR+6 COULD NOT
                                ;BE ADDRESSED.

        ;:$$$$$$$$$^^^ ERROR ^^^$$$$$$$$$
3$:      MOV      (SP)+,ERRVEC  ;RESTORE CONTENTS (IF LOC 4.
  
```

```

6006
6007      ::*****
(3)      ::*TEST 3          *TEST THAT IBS IS CLEAR AT INIT OF TESTING
(3)      ::*****
(2) 002654 000004      TST3:  SCOPE
(2)
(1) 002656 012737 000001 001160      MOV    #1,$TIMES      ;;DO 1 ITERATION
6008
6009 002664 000005      RESET          ;ISSUE SYSTEM INIT.
6010
6011 002666 005037 001124      CLR    $GDDAT        ;EXPECT ZERO CSR.
6012 002672 017737 176450 001126      MOV    @IBS,$BDDAT   ;READ CSR.
6013 002700 001401      BEQ    TST4          ;;
6014
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

```

(1)
(1) 002702 104003      ERROR 3          ;/MODULE FAULT DETECTED:
6015                                     ;IBS NOT CLEAR ON INT.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

```

6017
6018      ::*****
(3)      ::*TEST 4          *TEST THAT IBD IS CLEAR AT INIT OF TESTING
(3)      ::*****
(2) 002704 000004      TST4:  SCOPE
(2)
(1) 002706 012737 000001 001160      MOV    #1,$TIMES      ;;DO 1 ITERATION
6019
6020 002714 000005      RESET          ;ISSUE SYSTEM INITIALIZE.
6021
6022 002716 005037 001124      CLR    $GDDAT        ;EXPECT ZERO CSR.
6023 002722 117737 176422 001126      MOV    @IBD,$BDDAT   ;READ DBR
6024 002730 001401      BEQ    TST5          ;;
6025
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

```

(1)
(1) 002732 104004      ERROR 4          ;/MODULE FAULT DETECTED:
6026                                     ;IBD NOT CLEAR ON INIT.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

```

6028
6035
6036      ::*****
(3)      ::*TEST 5          *TEST THAT BASE ADDRESSES +4,+6 RETURN ZERO WHEN READ
(4)      ;*
(4)      ;*BASE ADDRESS +4 AND +6 SHOULD RETURN A ZERO WHEN
(4)      ;*READ, IN THIS TEST WE WILL TRY THAT.
(4)      ;*
(3)      ::*****
(2) 002734 000004      TST5:  SCOPE
(2)
(1) 002736 012737 000010 001160      MOV    #10,$TIMES    ;;DO 10 ITERATIONS
  
```

```

6037
6038 002744 005037 001124 CLR $GDDAT ;EXPECT ZERO RETURN
6039 002750 017737 176376 001126 MOV @IBWC,$BDDAT ;READ BASE ADDRESS+4
6040 002756 001402 BEQ 1$ ;IF ZERO - GOOD.
6041
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 002760 104006 ERROR 6 ;/MODULE FAULT DETECTED:
6042 ;SHOULD HAVE READ BACK ZERO FROM
6043 ;THIS ADDR.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6045 002762 000405 BR TST6 ;:
6046
6047 002764 017737 176364 001126 1$: MOV @IBCA,$BDDAT ;READ BASE ADDR+6, SHOULD BE ZERO
6048 002772 001401 BEQ TST6 ;:
6049
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 002774 104006 ERROR 6 ;/MODULE FAULT DETECTED:
6050 ;SHOULD HAVE READ BACK ZERO FROM
6051 ;BASE ADDR+6.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6053
6054 *****
(3) *TEST 6 *TEST THAT WE CAN SET TCS, TCS SETS CMD
(3) *****
(2) 002776 000004 TST6: SCOPE
(2)
  
```

```

6055
6056 003000 005077 176342 CLR @IBS ;CLEAR CLR
6057 003004 052777 000001 176334 BIS #BIT0,@IBS ;SET TCS.
6058 003012 052737 002001 001124 BIS #BIT0!BIT10,$GDDAT ;EXPECT ONLY TCS AND CMD TO SET
6059 003020 017737 176322 001126 MOV @IBS,$BDDAT ;READ THE IBS.
6060 003026 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID TCS AND CMD SET?
6061 003034 000402 BR 1$ ;YES - CONTINUE
6062
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003036 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6063 ;TCS AND/OR CMD FAILED TO SET
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6065 003040 000412 BR TST7 ;:
6066
6067 003042 042777 000001 176276 1$: BIC #BIT0,@IBS ;CLEAR TCS.
6068 003050 005037 001124 CLR $GDDAT ;EXPECT TCS AND CMD TO CLEAR
6069 003054 017737 176266 001126 MOV @IBS,$BDDAT ;READ IBS, DID THEY CLEAR?
6070 003062 001401 BEQ TST7 ;:
  
```


6071

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003064 104003
6072

ERROR 3 ;/MODULE FAULT DETECTED:
;TCS AND/OR CMD FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6074
6075
(3)
(3)
(2) 003066 000004
(2)

:::*****
:*TEST 7 *TEST THAT EOP WILL SET
:*****
TST7: SCOPE

6076
6077 003070 005077 176252
6078 003074 052777 000002 176244
6079 003102 012737 000002 001124
6080 003110 017737 176232 001126
6081 003116 023737 001124 001126
6082 003124 001402
6083

CLR @IBS ;CLEAR CSR.
BIS #BIT1,@IBS ;SET EOP
MOV #BIT1,\$GDDAT ;EXPECT ONLY EOP TO SET.
MOV @IBS,\$BDDAT ;READ IBS
CMP \$GDDAT,\$BDDAT ;DID EOP SET?
BEQ 1\$;YES - CONTINUE

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003126 104003
6084

ERROR 3 ;/MODULE FAULT DETECTED:
;EOP BIT SETTING ERROR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6086 003130 000412
6087
6088 003132 042777 000002 176206 1\$:
6089 003140 005037 001124
6090 003144 017737 176176 001126
6091 003152 001401
6092

BR TST10 ;:
BIC #BIT1,@IBS ;CLEAR EOP
CLR \$GDDAT ;EXPECT A ZERO CSR.
MOV @IBS,\$BDDAT ;READ IBS, IS IT CLEAR?
BEQ TST10 ;:

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003154 104003
6093

ERROR 3 ;/MODULE FAULT DETECTED:
;IBS FAILED TO CLEAR

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6095
6096
(3)
(3)
(2) 003156 000004
(2)

:::*****
:*TEST 10 *TEST THAT RE WILL SET + CLEAR
:*****
TST10: SCOPE

6097
6098 003160 005077 176162
6099 003164 052777 000004 176154
6100 003172 012737 000004 001124

CLR @IBS ;CLEAR CSR.
BIS #BIT02,@IBS ;SET REM
MOV #BIT02,\$GDDAT ;EXPECT ONLY REM TO SET.

6101 003200 017737 176142 001126 MOV @IBS,\$BDDAT ;READ IBS.
6102 003206 023737 001126 001124 CMP \$BDDAT,\$GDDAT ;DID REM AND ONLY REM SET?
6103 003214 001402 BEQ 1\$;YES - CONTINUE
6104

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003216 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6105 ;REM BIT SETTING ERROR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
BR TST11 ;:

6107 003220 000412
6108
6109 003222 042777 000004 176116 1\$: BIC #BIT02,@IBS ;CLEAR REM BIT.
6110 003230 005037 001124 CLR \$GDDAT ;EXPECT ZERO CSR.
6111 003234 017737 176106 001126 MOV @IBS,\$BDDAT ;READ IBS - IS IT CLEAR?
6112 003242 001401 BEQ TST11 ;:
6113

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003244 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6114 ;IBS FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6116
6117 :*****
(3) :*TEST 11 *TEST THAT IBC WILL SET AND CLEAR
(3) :*****
(2) 003246 000004 TST11: SCOPE
(2)

6118
6119 003250 005077 176072 CLR @IBS ;CLEAR CSR.
6120 003254 052777 000010 176064 BIS #BIT03,@IBS ;SET IBC
6121 003262 012737 000010 001124 MOV #BIT03,\$GDDAT ;EXPECT ONLY IBC TO BE SET
6122 003270 017737 176052 001126 MOV @IBS,\$BDDAT ;READ IBS.
6123 003276 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;DID IBS SET?
6124 003304 001414 BEQ 1\$;YES CONTINUE
6125 003306 012777 000010 176032 MOV #BIT03,@IBS ;TRY SETTING IBC AGAIN.
6126 003314 017737 176026 001126 MOV @IBS,\$BDDAT ;MEMORY REFRESH MIGHT HAVE
6127 003322 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;GOT IN THE WAY.
6128 003330 001402 BEQ 1\$

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 003332 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6130 ;IBS BIT SETTING ERROR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
BR TST12 ;:

6132 003334 000416
6133
6134 003336 004537 007106 1\$: JSR R5,DEL50 ;DELAY 150 US.

```

6135 003342 000006 .WORD 6
6136
6137 003344 012737 002001 001124 MOV #BIT10!BIT0,$GDDAT ;EXP CMD AND TCS.
6138 003352 017737 175770 001126 MOV @IBS,$BDDAT ;READ IBS - IS IT CLEAR?
6139 003360 023737 001124 001126 CMP $GDDAT,$BDDAT
6140 003366 001401 BEQ TST12 ;;
6141
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003370 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6142 ;IBS NOT CLEAR AFTER IBC
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6144 :*****
(3) :*TEST 12 *TEST THAT TON (BIT05) AND TKR SET AND CLEAR
(3) :*****
(2) 003372 000004 TST12: SCOPE
(2)
  
```

```

6145
6146 003374 005077 175746 CLR @IBS ;CLEAR THE CSR.
6147 003400 052777 000040 175740 BIS #BIT5,@IBS ;SET TON.
6148 003406 012737 001040 001124 MOV #BIT5!BIT9,$GDDAT ;EXPECT ONLY TON AND TKR TO SET.
6149 003414 017737 175726 001126 MOV @IBS,$BDDAT ;READ CSR.
6150 003422 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID THEY BOTH SET?
6151 003430 001402 BEQ 1$ ;YES - CONTINUE.
6152
6153
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003432 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6154 ;ERROR IN SETTING TON BIT.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6156 003434 000412 BR TST13 ;;
6157
6158 003436 042777 000040 175702 1$: BIC #BIT5,@IBS ;WHEN TON CLEARED, TKR SHOULD CLEAR.
6159 003444 005037 001124 CLR $GDDAT ;EXPECT ZERO CSR.
6160 003450 017737 175672 001126 MOV @IBS,$BDDAT ;DID IT CLEAR?
6161 003456 001401 BEQ TST13 ;;
6162
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003460 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6163 ;CSR FAILED TO CLEAR.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6165 :*****
6166 :*TEST 13 *MAKE SURE WE CAN SET AND CLEAR BIT06 (IE)
(3) :*****
(3)
  
```

```

(2) 003462 000004 TST13: SCOPE
(2)
6167
6168
(1) 003464 012746 000300 MOV #300,-(SP) ;/PR
(1) 003470 012746 003476 MOV #64$,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 003474 000002 RTI ;/SHOW RETURN ADDRESS
(1) 003476 64$: ;/CAUSE A RETURN (PUTS NEW STATUS
; /IN STATUS REG.)
6169
6170 003476 005077 175644 CLR @IBS ;CLEAR CSR.
6171
6172 003502 052777 000100 175636 BIS #BIT6,@IBS ;SET IE.
6173 003510 012737 000100 001124 MOV #BIT6,$GDDAT ;EXPECT ONLY BIT 6 TO SET.
6174 003516 017737 175624 001126 MOV @IBS,$BDDAT ;READ IBS.
6175 003524 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID IE SET?
6176 003532 001402 BEQ 1$ ;YES - CONTINUE.
6177

```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003534 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6178 ;ERROR IN SETTING IE BIT.

```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6180 003536 000412 BR TST14 ;:
6181
6182 003540 005037 001124 1$: CLR $GDDAT ;EXPECT ZERO CSR AFTER.
6183 003544 042777 000100 175574 BIC #BIT6,@IBS ;IE IS CLEARED.
6184 003552 017737 175570 001126 MOV @IBS,$BDDAT ;READ CSR - IS IT CLEAR?
6185 003560 001401 BEQ TST14 ;:
6186

```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 003562 104003 ERROR 3 ;/MODULE FAULT DETECTED:
6187 ;FAILED TO CLEAR CSR.

```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6189
6190 ;:*****
(3) ;*TEST 14 *TEST THAT BIT 7 (ACC) CAN BE SET AND CLEARED
(3) ;:*****
(2) 003564 000004 TST14: SCOPE
(2)

```

```

6191
6192 003566 005077 175554 CLR @IBS ;CLEAR CSR.
6193 003572 052777 000200 175546 BIS #BIT7,@IBS ;SET ACC.
6194 003600 012737 000200 001124 MOV #BIT7,$GDDAT ;EXPECT ONLY ACC TO SET.
6195 003606 017737 175534 001126 MOV @IBS,$BDDAT ;READ IBS.
6196 003614 023737 001124 001126 CMP $GDDAT,$BDDAT ;DID ACC SET?
6197 003622 001402 BEQ 1$ ;YES - CONTINUE.
6198

```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
 (1) 003624 104003 ERROR 3 :/MODULE FAULT DETECTED:
 6199 :FAILURE IN SETTING BIT 7 (ACC).

::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$
 BR TST15 ::
 6201 003626 000412
 6202
 6203 003630 042777 000200 175510 1\$: BIC #BIT7,@IBS :TRY CLEARING ACC.
 6204 003636 005037 001124 CLR \$GDDAT :EXPECT ZERO CSR.
 6205 003642 017737 175500 001126 MOV @IBS,\$BDDAT :READ IBS, IS IT CLEAR?
 6206 003650 001401 BEQ TST15 ::
 6207

::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(1)
 (1) 003652 104003 ERROR 3 :/MODULE FAULT DETECTED:
 6208 :IBS FAILED TO CLEAR.

::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

6210
 6237
 6238
 (5)
 (4) :*****
 (4) :*TEST 15 *TEST THAT IBD BIT 0 CAN BE SET + CLEARED
 (3) 003654 000004 TST15: SCOPE :*****

(1)
 (1) 003656 012777 000060 175462 MOV #BIT4!BIT5,@IBS :/MACRO BDT
 (1) 003664 012737 000001 001124 MOV #BIT0,\$GDDAT :/SET TON AND LON.
 (1) 003672 013777 001124 175450 MOV \$GDDAT,@IBD :/WE'RE GONNA TEST BIT 0.
 (1) :/SET THE BIT.
 (1) 003700 117737 175444 001126 MOV @IBD,\$BDDAT :/READ THE IBD.
 (1) 003706 123737 001124 001126 CMPB \$GDDAT,\$BDDAT :/DID IT GET THRU OK?
 (1) 003714 001402 BEQ 1\$:/YES - CONTINUE.

::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(2)
 (2) 003716 104004 ERROR 4 :/MODULE FAULT DETECTED:
 (1) :/ERROR IN SETTING IBD BIT 0.

::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$
 (3) 003720 000412 BR TST16 ::
 (1) 003722 005037 001124 CLR \$GDDAT :/EXPECT ZERO IBD WHEN
 (1) 003726 042777 CPU001 175414 1\$: BIC #BIT0,@IBD :/BIT 0 IS CLEARED.
 (1) 003734 117737 175410 001126 MOV @IBD,\$BDDAT :/READ IBD, IS IT CLEAR?
 (3) 003742 001401 BEQ TST16 ::
 (2)

::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(2) 003744 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;FAILED TO CLEAR IBD.
;:\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

(1) 6239
(5) ;:*****
(4) ;*TEST 16 *TEST THAT IBD BIT 1 CAN BET SET + CLEARED
(4) ;:*****
(3) 003746 000004 TST16: SCOPE

(1) ;/MACRO BDT
(1) 003750 012777 000060 175370 MOV #BIT4!BIT5,@IBS ;/SET TON AND LON.
(1) 003756 012737 000002 001124 MOV #BIT1,\$GDDAT ;/WE'RE GONNA TEST BIT 1.
(1) 003764 013777 001124 175356 MOV \$GDDAT,@IBD ;/SET THE BIT.
(1) 003772 117737 175352 001126 MOVB @IBD,\$BDDAT ;/READ THE IBD.
(1) 004000 123737 001124 001126 CMPB \$GDDAT,\$BDDAT ;/DID IT GET THRU OK?
(1) 004006 001402 BEQ 1\$;/YES - CONTINUE.
(1)
(2)

;:\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(2) 004010 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/ERROR IN SETTING IBD BIT 1.

;:\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$
(3) 004012 000412 BR TST17 ;:
(1) 004014 005037 001124 1\$: CLR \$GDDAT ;/EXPECT ZERO IBD WHEN
(1) 004020 042777 000002 175322 BIC #BIT1,@IBD ;/BIT 1 IS CLEARED.
(1) 004026 117737 175316 001126 MOVB @IBD,\$BDDAT ;/READ IBD, IS IT CLEAR?
(3) 004034 001401 BEQ TST17 ;:
(2)

;:\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(2) 004036 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;FAILED TO CLEAR IBD.

;:\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

(1) 6240
(5) ;:*****
(4) ;*TEST 17 *TEST THAT IBD BIT 2 CAN BET SET + CLEARED
(4) ;:*****
(3) 004040 000004 TST17: SCOPE

(1) ;/MACRO BDT
(1) 004042 012777 000060 175276 MOV #BIT4!BIT5,@IBS ;/SET TON AND LON.

```

(1) 004050 012737 000004 001124 MOV #BIT2,$GDDAT ;/WE'RE GONNA TEST BIT 2.
(1) 004056 013777 001124 175264 MOV $GDDAT,@IBD ;/SET THE BIT.
(1) 004064 117737 175260 001126 MOVB @IBD,$BDDAT ;/READ THE IBD.
(1) 004072 123737 001124 001126 CMPB $GDDAT,$BDDAT ;/DID IT GET THRU OK?
(1) 004100 001402 BEQ 1$ ;/YES - CONTINUE.
(1)
(2)
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004102 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/ERROR IN SETTING IBD BIT 2.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

(3) 004104 000412 BR TST20 ;/
(1) 004106 005037 001124 175230 1$: CLR $GDDAT ;/EXPECT ZERO IBD WHEN
(1) 004112 042777 000004 175230 BIC #BIT2,@IBD ;/BIT 2 IS CLEARED.
(1) 004120 117737 175224 001126 MOVB @IBD,$BDDAT ;/READ IBD, IS IT CLEAR?
(3) 004126 001401 BEQ TST20 ;/
(2)
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004130 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/FAILED TO CLEAR IBD.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

(1) 6241
(5)
(4) :::*****
(4) *TEST 20 *TEST THAT IBD BIT 3 CAN BET SET + CLEARED
(3) 004132 000004 TST20: SCOPE
(3)
(1)
  
```

```

(1) 004134 012777 000060 175204 MOV #BIT4!BIT5,@IBS ;/MACRO BDT
(1) 004142 012737 000010 001124 MOV #BIT3,$GDDAT ;/SET TON AND LON.
(1) 004150 013777 001124 175172 MOV $GDDAT,@IBD ;/WE'RE GONNA TEST BIT 3.
(1) ;/SET THE BIT.
(1) 004156 117737 175166 001126 MOVB @IBD,$BDDAT ;/READ THE IBD.
(1) 004164 123737 001124 001126 CMPB $GDDAT,$BDDAT ;/DID IT GET THRU OK?
(1) 004172 001402 BEQ 1$ ;/YES - CONTINUE.
(1)
(2)
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004174 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/ERROR IN SETTING IBD BIT 3.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

(3) 004176 000412
(1) 004200 005037 001124 1$: BR TST21 ;;
(1) 004204 042777 000010 175136 CLR $GDDAT ;;/EXPECT ZERO IBD WHEN
(1) 004212 117737 175132 001126 BIC #BIT3,@IBD ;;/BIT 3 IS CLEARED.
(3) 004220 001401 MOV @IBD,$BDDAT ;;/READ IBD, IS IT CLEAR?
(2) BEQ TST21 ;;
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004222 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/FAILED TO CLEAR IBD.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6242

```

(5) *****
(4) *TEST 21 *TEST THAT IBD BIT 4 CAN BET SET + CLEARED
(4) *****
(3) 004224 000004 TST21: SCOPE
(3)
(1)
  
```

```

(1) 004226 012777 000060 175112 MOV #BIT4!BIT5,@IBS ;/MACRO BDT
(1) 004234 012737 000020 001124 MOV #BIT4,$GDDAT ;/SET TON AND LON.
(1) 004242 013777 001124 175100 MOV $GDDAT,@IBD ;/WE'RE GONNA TEST BIT 4.
(1) ;/SET THE BIT.
(1) 004250 117737 175074 001126 MOV @IBD,$BDDAT ;/READ THE IBD.
(1) 004256 123737 001124 001126 CMPB $GDDAT,$BDDAT ;/DID IT GET THRU OK?
(1) 004264 001402 BEQ 1$ ;/YES - CONTINUE.
(1)
(2)
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004266 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/ERROR IN SETTING IBD BIT 4.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

(3) 004270 000412
(1) 004272 005037 001124 1$: BR TST22 ;;
(1) 004276 042777 000020 175044 CLR $GDDAT ;;/EXPECT ZERO IBD WHEN
(1) 004304 117737 175040 001126 BIC #BIT4,@IBD ;;/BIT 4 IS CLEARED.
(3) 004312 001401 MOV @IBD,$BDDAT ;;/READ IBD, IS IT CLEAR?
(2) BEQ TST22 ;;
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(2) 004314 104004 ERROR 4 ;/MODULE FAULT DETECTED:
(1) ;/FAILED TO CLEAR IBD.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

6243


```

    (2)
    (2) 004452 104004          ERROR 4          :/MODULE FAULT DETECTED:
    (1)                                     :/ERROR IN SETTING IBD BIT 6.
  
```

```

    (3) 004454 000412          :/MODULE FAULT DETECTED:
    (1) 004456 005037 001124 1$: BR TST24          :/ERROR IN SETTING IBD BIT 6.
    (1) 004462 042777 000100 174660 CLR $GDDAT          :/EXPECT ZERO IBD WHEN
    (1) 004470 117737 174654 001126 BIC #BIT6,@IBD      :/BIT 6 IS CLEARED.
    (3) 004476 001401          MOVB @IBD,$BDDAT    :/READ IBD, IS IT CLEAR?
    (2) BEQ TST24          :/
  
```

```

    (2)
    (2) 004500 104004          ERROR 4          :/MODULE FAULT DETECTED:
    (1)                                     :/FAILED TO CLEAR IBD.
  
```

```

    (1)
    6245
    (5)
    (4)
    (4)
    (3) 004502 000004          :/MODULE FAULT DETECTED:
    (3)
    (1)
    (1)
    (1) 004504 012777 000060 174634 MOV #BIT4!BIT5,@IBS :/MACRO BDT
    (1) 004512 012737 000200 001124 MOV #BIT7,$GDDAT    :/SET TON AND LON.
    (1) 004520 013777 001124 174622 MOV $GDDAT,@IBD     :/WE'RE GONNA TEST BIT 7.
    (1)
    (1) 004526 117737 174616 001126 MOVB @IBD,$BDDAT    :/SET THE BIT.
    (1) 004534 123737 001124 001126 CMPB $GDDAT,$BDDAT  :/READ THE IBD.
    (1) 004542 001402          BEQ 1$             :/DID IT GET THRU OK?
    (1)
    (2)
  
```

```

    (2)
    (2) 004544 104004          ERROR 4          :/MODULE FAULT DETECTED:
    (1)                                     :/ERROR IN SETTING IBD BIT 7.
  
```

```

    (3) 004546 000412          :/MODULE FAULT DETECTED:
    (1) 004550 005037 001124 1$: BR TST25          :/ERROR IN SETTING IBD BIT 7.
    (1) 004554 042777 000200 174566 CLR $GDDAT          :/EXPECT ZERO IBD WHEN
    (1) 004562 117737 174562 001126 BIC #BIT7,@IBD      :/BIT 7 IS CLEARED.
    (3) 004570 001401          MOVB @IBD,$BDDAT    :/READ IBD, IS IT CLEAR?
    (2) BEQ TST25          :/
  
```

```

    :/MODULE FAULT DETECTED:
    :/ERROR IN SETTING IBD BIT 7.
  
```

(2) 004572 104004 ERROR 4 :/MODULE FAULT DETECTED:
 (1) :FAILED TO CLEAR IBD.

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

(1) 6246
 6247
 (3) :*****
 (3) :*TEST 25 *TEST THAT NO DATA GETS XFERRERD, IF NOT ENABLED
 (2) 004574 000004 :*****
 (2) TST25: SCOPE

6248
 6249 004576 005077 174544 CLR @IBS :CLEAR CSR
 6250 004602 112777 000252 174540 MOVB #252,@IBD :TRY XFERRING DATA
 6251 004610 005037 001124 CLR \$BDDAT :NO DATA SHOULD XFERR
 6252 004614 117737 174530 001126 MOVB @IBD,\$BDDAT :READ BUFFER REG.
 6253 004622 001401 BEQ TST26 :
 6254

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(1) 004624 104002 ERROR 2 :/MODULE FAULT DETECTED:
 (1) :DATA WAS XFERRERD THROUGH IBD
 6255 :EVEN THOUGH TON AND LON CLEARED.
 6256 :SIGNAL 'ENB XFER L' PROBABLY
 6257 :STUCK LOW.
 6258

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

6260
 6261
 (3) :*****
 (3) :*TEST 26 *TEST IBD BITS DAC, AND DAV
 (2) 004626 000004 :*****
 (2) TST26: SCOPE

6262
 6263 004630 005077 174512 CLR @IBS :CLEAR CSR.
 6264 004634 005077 174510 CLR @IBD :CLEAR DATA REG.
 6265 004640 032777 000400 174502 BIT #BIT8,@IBD :IS DAC SET?
 6266 004646 001002 BNE 4\$:YES (GOOD)
 6267

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(1) 004650 104002 ERROR 2 :/MODULE FAULT DETECTED:
 (1) :DAC NOT SET.
 6268

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

6270 004652 000437 BR TST27 :
 6271 004654 4\$:
 6272 004654 052777 000260 174464 BJS #BITS!BIT4!BIT7,@IBS :SET TON AND LON
 6273 004662 012777 000252 174460 MOV #252,@IBD :PUT DATA IN IBD.
 6274 004670 017737 174454 001126 MOV @IBD,\$BDDAT :READ IBD.

6275 004676 032737 001000 001126 BIT #BIT9,\$BDDAT ;DID DAV SET?
6276 004704 001002 BNE 1\$;YES - CONTINUE.
6277

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 004706 104002 ERROR 2 ;/MODULE FAULT DETECTED:
6278 ;DAV FAILED TO SET.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
BR TST27 ;:

6280 004710 000420
6281
6282 004712 012777 000060 174426 1\$: MOV #BIT4!BIT5,@IBS ;CLEAR ACC.
6283 004720 105777 174424 2\$: TSTB @IBD ;READ LOW BYTE OF IBD.
6284 004724 032777 000400 174416 BIT #BIT8,@IBD ;DID DAC CLEAR?
6285 004732 001402 BEQ 3\$;YES - CONTINUE.
6286

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 004734 104002 ERROR 2 ;/MODULE FAULT DETECTED:
6287 ;DAC FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
BR TST27 ;:
6289 004736 000405 BIT #BIT9,@IBD ;DID DAV CLEAR?
6290 004740 032777 001000 174402 3\$: BEQ TST27 ;:
6291 004746 001401
6292

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)
(1) 004750 104002 ERROR 2 ;/MODULE FAULT DETECTED:
6293 ;DAV FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
;/MACRO -SIGC-

6336
6337
(1)
(5) :*****
(4) :*TEST 27 *TEST THAT REN SETS WHEN REM SETS, ALSO TEST CLEAR
(4) :*****
(3) 004752 000004 TST27: SCOPE

(1) 004754 005077 174366 CLR @IBS ;/CLEAR CSR.
(1) 004760 052777 000004 174360 BIS #BIT2,@IBS ;/SET REM, SHOULD SET REN.
(1) 004766 032777 010000 174354 BIT #BIT12,@IBD ;/DID REN SET?
(1) 004774 001011 BNE 1\$;/YES - LETS TRY CLEARING IT.
(1) 004776 052777 000004 174342 BIS #BIT2,@IBS ;/SET REM, MEMORY
(1) ;/REFRESH COULD HAVE
(1) ;/INTERRUPTED US.
(1) 005004 032777 010000 174336 BIT #BIT12,@IBD ;/DID REN SET THIS TIME?
(1) 005012 001002 BNE 1\$

(2) ::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$
 (2) 005014 104002 ERROR 2 :/MODULE FAULT DETECTED:
 (1) :/REN FILED TO SET WHEN REM SET.

(3) 005016 000410 ::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
 (1) 005020 042777 000004 174320 1\$: BR TST30 :/
 (1) 005026 032777 010000 174314 BIC #BIT2,@IBS :/CLEAR REM, SHOULD CLEAR REN.
 (3) 005034 001401 BIT #BIT12,@IBD :/DID REN CLEAR?
 (2) BEQ TST30 :/

(2) ::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$
 (2) 005036 104002 ERROR 2 :/MODULE FAULT DETECTED:
 (1) :/REN FAILED TO CLEAR.

(1) 6338
 6339
 (1) :/MACRO -SIGC-
 (5)

(4) ::*****
 (4) *TEST 30 *TEST THAT IFC SETS WHEN IBS SETS, ALSO TEST CLEAR
 (3) 005040 000004 TST30: SCOPE
 (3)

(1) 005042 005077 174300 CLR @IBS :/CLEAR CSR.
 (1) 005046 052777 000010 174272 BIS #BIT3,@IBS :/SET IBS, SHOULD SET IFC.
 (1) 005054 032777 020000 174266 BIT #BIT13,@IBD :/DID IFC SET?
 (1) 005062 001011 BNE 1\$:/YES - LETS TRY CLEARING IT.
 (1) 005064 052777 000010 174254 BIS #BIT3,@IBS :/SET IBS, MEMORY
 (1) :/REFRESH COULD HAVE
 (1) 005072 032777 020000 174250 BIT #BIT13,@IBD :/INTERRUPTED US.
 (1) 005100 001002 BNE 1\$:/DID IFC SET THIS TIME?
 (2)

(2) ::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$
 (2) 005102 104002 ERROR 2 :/MODULE FAULT DETECTED:
 (1) :/IFC FILED TO SET WHEN IBS SET.

(3) 005104 000411 ::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$
 (1) 005106 032777 000010 174232 1\$: BR TST31 :/
 (1) 005114 001374 BIT #BIT3,@IBS :/WAIT FOR IBS TO CLEAR.
 (1) BNE 1\$:/
 (1) 005116 032777 020000 174224 BIT #BIT13,@IBD :/IBS CLEAR, DID IFC CLEAR?

(3) 005124 001401
(2)

BEG TST31 ::

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

(2)
(2) 005126 104002
(1)

ERROR 2 :/MODULE FAULT DETECTED:
:/IFC FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

(1)
6340
6341

:/MACRO -SIGC-

(1)
(5)
(4)
(4)

:::*****
:*TEST 31 *TEST THAT ATN SETS WHEN TCS SETS, ALSO TEST CLEAR
:::*****
TST31: SCOPE

(3) 005130 000004
(3)
(1)

(1) 005132 005077 174210
(1) 005136 052777 000001 174202
(1) 005144 032777 040000 174176
(1) 005152 001011
(1) 005154 052777 000001 174164
(1)
(1)
(1) 005162 032777 040000 174160
(1) 005170 001002
(2)

CLR @IBS :/CLEAR CSR.
BIS #BIT0,@IBS :/SET TCS, SHOULD SET ATN.
BIT #BIT14,@IBD :/DID ATN SET?
BNE 1\$:/YES - LETS TRY CLEARING IT.
BIS #BIT0,@IBS :/SET TCS, MEMORY
:/REFRESH COULD HAVE
:/INTERRUPTED US.
BIT #BIT14,@IBD :/DID ATN SET THIS TIME?
BNE 1\$

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

(2)
(2) 005172 104002
(1)

ERROR 2 :/MODULE FAULT DETECTED:
:/ATN FILED TO SET WHEN TCS SET.

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

(3) 005174 000410
(1) 005176 042777 000001 174142 1\$:
(1) 005204 032777 040000 174136
(3) 005212 001401
(2)

BR TST32 ::
BIC #BIT0,@IBS :/CLEAR TCS, SHOULD CLEAR ATN.
BIT #BIT14,@IBD :/DID ATN CLEAR?
BEG TST32 ::

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

(2)
(2) 005214 104002
(1)

ERROR 2 :/MODULE FAULT DETECTED:
:/ATN FAILED TO CLEAR.

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

(1)
6342
6343
(1)
(5)

:/MACRO -SIGC-

:::*****

```

(4) : *TEST 32 *TEST THAT EOI SETS WHEN EOP SETS, ALSO TEST CLEAR
(4) : *****
(3) 005216 000004 TST32: SCOPE
(3)
(1)
(1) 005220 005077 174122 CLR @IBS ;/CLEAR CSR.
(1) 005224 052777 000002 174114 BIS #BIT1,@IBS ;/SET EOP, SHOULD SET EOI.
(1) 005232 032777 100000 174110 BIT #BIT15,@IBD ;/DID EOI SET?
(1) 005240 001011 BNE 1$ ;/YES - LETS TRY CLEARING IT.
(1) 005242 052777 000002 174076 BIS #BIT1,@IBS ;/SET EOP, MEMORY
(1) ;/REFRESH COULD HAVE
(1) ;/INTERRUPTED US.
(1) 005250 032777 100000 174072 BIT #BIT15,@IBD ;/DID EOI SET THIS TIME?
(1) 005256 001002 BNE 1$
(2)
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(2)
(2) 005260 104002 ERROR 2 ;/MODULE FAULT DETECTED:
(1) ;/EOI FILED TO SET WHEN EOP SET.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

```

(3) 005262 000410 BR TST33 ;/
(1) 005264 042777 000002 174054 1$: @IC #BIT1,@IBS ;/CLEAR EOP, SHOULD CLEAR EOI.
(1) 005272 032777 100000 174050 BIT #BIT15,@IBD ;/DID EOI CLEAR?
(3) 005300 001401 BEQ TST33 ;/
(2)
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(2)
(2) 005302 104002 ERROR 2 ;/MODULE FAULT DETECTED:
(1) ;/EOI FAILED TO CLEAR.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

```

(1)
6344
6345 : *****
(3) : *TEST 33 *TEST THAT RFD SET WHEN CSR CLEAR,CLEAR WHEN ACC SET
(3) : *****
(2) 005304 000004 TST33: SCOPE
(2)
6346
6347 005306 005077 174034 CLR @IBS ;/CLEAR CSR.
6348 005312 032777 002000 174030 BIT #BIT10,@IBD ;/DID RFD SET?
6349 005320 001002 BNE 1$ ;/YES CONTINUE.
6350
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005322 104002 ERROR 2 ;/MODULE FAULT DETECTED:
6351 ;/RFD FAILED TO SET.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$\$\$

```

6353 005324 000410 BR TST34 ;;
6354
6355 005326 052777 000200 174012 1$: BIS #BIT7,@IBS ;NOW SET ACC,RFD SHOULD CLEAR.
6356 005334 032777 002000 174C06 BIT #BIT10,@IBD ;DID IT CLEAR?
6357 005342 001401 BEQ TST34 ;;
6358
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005344 104002 ERROR 2 ;/MODULE FAULT DETECTED:
6359 ;RFD FAILED TO CLEAR.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6361
6362
6363 *****
(3) *TEST 34 *TEST THAT WE CAN GENERATE AN ER2
(3) *****
(2) 005346 000004 TST34: SCOPE
(2)
  
```

```

6364
6365 005350 005077 173772 CLR @IBS ;CLEAR THE STATUS REG.
6366 005354 052777 000041 173764 BIS #BIT5!BIT0,@IBS ;SET TON; THIS SHOULD CAUSE AN
6367 ;ERROR SENCE NO LISTENERS ARE ON
6368 005362 105077 173762 CLRB @IBD ;AND WE SENT DATA TO THE BUS.
6369 ;BUS.
6370 005366 032777 040000 173752 BIT #BIT14,@IBS ;DID ER2 SET?
6371 005374 001001 BNE TST35 ;;
6372
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005376 104001 ERROR 1 ;/MODULE FAULT DETECTED:
6373 ;ER2 FAILED TO SET.
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6375
6376 *****
(3) *TEST 35 *TEST THAT BUS INIT CLEARS ACC,TON,LON,REM,EIP,TCS
(3) *****
(2) 005400 000004 TST35: SCOPE
(2)
  
```

```

(1) 005402 012737 000005 001160 MOV #5,$TIMES ;;DO 5 ITERATIONS
6377
6378 005410 012777 000367 173730 MOV #367,@IBS ;SET ACC,TON,LON,REM,EOP, AND TCS.
6379 005416 000005 RESET ;ISSUE SYS INIT.
6380 005420 105777 173722 TSTB @IBS ;DID THEY ALL CLEAR?
6381 005424 001401 BEQ TST36 ;;
6382
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005426 104001 ERROR 1 ;/MODULE FAULT DETECTED:
  
```


6383 ;BUS INIT FAILED TO CLEAR CSR.
 6384

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

```

6386
6387
(3)
(3)
(2) 005430 000004
(2)
(1) 005432 012737 000005 001160      MOV    #5,$TIMES      ;;DO 5 ITERATIONS
6388
6389 005440 012777 000266 173700      MOV    #266,@IBS     ;SET ACC,TONLON,REM, AND EOP.
6390 005446 052777 000010 173672      BIS    #BIT3,@IBS   ;SET IBC, THIS SHOULD CLEAR ABOVE BITS.
6391 005454 032777 000010 173664      BIT    #BIT3,@IBS   ;WAIT TILL IBS CLEARS
6392 005462 001374
6393 005464 032777 000266 173654      BNE    1$           ;
6394 005472 001401
6395
1$: BIT    #BIT3,@IBS   ;DID THEY CLEAR?
    BEQ    TST37      ;;
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005474 104001      ERROR 1              ;/MODULE FAULT DETECTED:
6396
6397
                                ;ACC,TON,LON,REM, AND/OR EOP
                                ;FAILED TO CLEAR ON IBC
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

```

6399
6400
(3)
(3)
(2) 005476 000004
(2)
(1) 005500 012737 000005 001160      MOV    #5,$TIMES      ;;DO 5 ITERATIONS
6401
6402 005506 012777 000260 173632      MOV    #BIT7!BIT5!BIT4,@IBS ;SET ACC,TON, AND LON.
6403 005514 012777 000377 173626      MOV    #377,@IBD     ;LOAD IBD
6404 005522 000005
6405 005524 105777 173620      RESET
6406 005530 001401      TSTB   @IBD          ;ISSUE SYS INIT.
6407
                                ;DID IT CLEAR?
                                ;;
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(1)
(1) 005532 104002      ERROR 2              ;/MODULE FAULT DETECTED:
6408
6409
                                ;FAILED TO CLEAR LOW BYTE OF IBD ON
                                ;SYSTEM INIT.
  
```

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

```

6411
6412      .SBTTL
6413      .SBTTL      INTERRUPT TESTS
6414      .SBTTL
6415
  
```

6416 (3) 005534 000004
(2) 005534 000004
(2)
6417
6418 005536 005077 173604 CLR @IBS ;CLEAR THE CSR.
6419 005542 012777 000200 173636 MOV #200,@PRC ;SET UP INTERRUPT VECTOR
6420 005550 012777 000606 173604 MOV #1\$,@VECTC ;SET TCS, SHOULD CAUSE
6421 005556 052777 000101 173562 BIS #BIT0!BIT6,@IBS ;/PR
6422 (1) 005564 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 005570 012746 005576 MOV #64\$,-(SP) ;/SHOW RETURN ADDRESS
(1) 005574 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
(1) 005576 ;/IN STATUS REG.)

*TEST 40 *TEST THAT CMD CAN GENERATE AN INTERRUPT B

TST40: SCOPE

6423 005576 000240 64\$: NOP ;CMD TO SET AND GIVE US AN
6424 005600 000240 NOP ;INTERRUPT.

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(1) 005602 104001 ERROR 1 ;/MODULE FAULT DETECTED:
6426 (1) 005602 104001 ;CMD FAILED TO GENERATE AN INTERRUPT.

:::\$\$\$\$\$\$\$\$\$\$\$^^^ ERROR ^^ \$\$\$\$\$\$\$\$\$\$\$\$

6428 005604 000402 BR 2\$
6429 005606 1\$: ADD #4,\$P ;/ADD #4 TO STACK POINTER.
(1) 005606 062706 000004 2\$: CLR @IBS ;CLEAR INTERRUPT
6430 005612 005077 173530 ;/-RESV-
6431 (1) 005616 013777 001406 173536 MOV PRC,@VECTC ;/RESTORE VECTOR FOR
(1) 005624 012777 004700 173554 MOV #4700,@PRC ;/ILLEGAL INTRO.

*TEST 41 *TEST THAT TKR AND LNR CAN GENERATE INTERRUPTS

TST41: SCOPE

6434 005634 012777 000200 173544 MOV #200,@PRC ;SET UP INTERRUPT VECTOR FOR TKR INTERRUPT
6435 005642 012777 005726 173512 MOV #1\$,@VECTC ;SET TON AND LON
6436 005650 012777 000060 173470 MOV #BIT4!BIT5,@IBS ;ALLOW INTERRUPT
6437 005656 052777 000100 173462 BIS #BIT6,@IBS ;/PR
6438 (1) 005664 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 005670 012746 005676 MOV #64\$,-(SP) ;/SHOW RETURN ADDRESS
(1) 005674 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
(1) 005676 ;/IN STATUS REG.)

64\$: NOP ;/IN STATUS REG.)

6439 005676 000240 NOP
6440 005700 000240 NOP
6441

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

(1) 005702 104001 ERROR 1 ;/MODULE FAULT DETECTED:

6442 :FAILED TO GENERATE A TKR INTERRUPT.

```

:::$$$$$$$$$$$^ ERROR ^$$$$$$$$$$$
6444 005704 005077 173436 CLR @IBS :CLR CSR
6445 (1) 005710 013777 001406 173444 MOV PRC,@VECTC ;/RESTORE VECTOR FOR
(1) 005716 012777 004700 173462 MOV #4700,@PRC ;/ILLEGAL INTRO.
6446 005724 000443 BR TST42 ::
6447
6448 005726 062706 000004 1$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
(1) 005726 062706 000004 :/-RESV-
6449 (1) 005732 013777 001406 173422 MOV PRC,@VECTC ;/RESTORE VECTOR FOR
(1) 005740 012777 004700 173440 MOV #4700,@PRC ;/ILLEGAL INTRO.
6450 005746 012777 000200 173434 MOV #200,@PRD
6451 005754 012777 006010 173402 MOV #2$,@VECTD ;SET UP FOR LNR INTERRUPT.
6452 (1) 005762 012746 000000 :/PR
(1) 005766 012746 005774 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 005772 000002 MOV #65$,-(SP) ;/SHOW RETURN ADDRESS
(1) 005774 105277 173350 65$: RTI ;/CAUSE A RETURN (PUTS NEW STATUS
;IN STATUS REG.)
6453 006000 000240 INCB @IBD ;SEND DATA - CLRS TKR SETS LNR
6454 006002 000240 NOP ;FOR INTERRUPT
6455 NOP
6456 NOP
6457

```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$

```

(1) 006004 104001 ERROR 1 ;/MODULE FAULT DETECTED:
6458 :FAILED TO GENERATE LNR INTERRUPT

```

:::\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$

```

6460 006006 000402 BR 3$
6461
6462 006010 062706 000004 2$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
(1) 006010 062706 000004 3$: CLR @IBS ;CLEAR THE STATUS REG.
6463 006014 005077 173326 :/-RESV-
6464 (1) 006020 013777 001410 173336 MOV PRD,@VECTD ;/RESTORE VECTOR FOR
(1) 006026 012777 004700 173354 MOV #4700,@PRD ;/ILLEGAL INTRO.
6465
6466

```

```

:*****
:*TEST 42 *TEST THAT ER2 CAN GENERATE AN INTERRUPT
:*****
TST42: SCOPE

```

```

(2) 006034 000004
(2)
6467 006036 005077 173304 CLR @IBS ;START WITH CSR CLEAR
6468 006042 012777 000200 173332 MOV #200,@PRA
6469 006050 012777 006126 173300 MOV #1$,@VECTA ;SET UP INTERRUPT VECTOR
6471 (1) 006056 012746 000200 :/PR
(1) 006062 012746 006070 MOV #200,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 006066 000002 MOV #64$,-(SP) ;/SHOW RETURN ADDRESS
;CAUSE A RETURN (PUTS NEW STATUS

```

```

(1) 006070 6472 006070 052777 000140 173250 64$: BIS #BIT5!BIT6,@IBS ;/IN STATUS REG.)
6473 006076 105077 173246 CLR @IBD ;SET TON - NO LISTNERS ON
6474 006102 000240 NOP ;BUS BUT DATA PUT ON
6475 006104 000240 NOP ;BUS - THEREFORE AN INTERRUPT
6476 ;SHOULD BE POSTED.
(1) 006106 012746 000000 MOV #0,-(SP) ;/PR
(1) 006112 012746 006120 MOV #65,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 006116 000002 RTI ;/SHOW RETURN ADDRESS
(1) 006120 65$: ;/CAUSE A RETURN (PUTS NEW STATUS
6477 006120 000240 NOP ;/IN STATUS REG.)
6478
6479
  
```

:::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

```

(1) 6480 006122 104001 ERROR 1 ;/MODULE FAULT DETECTED:
;FAILED TO INTERRUPT ON ERROR2
  
```

:::\$\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$\$

```

6482 006124 000402 BR 2$
6483 006126 1$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
6484 006132 005077 173210 2$: CLR @IBS ;CLEAR CSR
6485 ;/-RESV-
(1) 006136 013777 001402 173212 MOV PRA,@VECTA ;/RESTORE VECTOR FOR
(1) 006144 012777 004700 173230 MOV #4700,@PRA ;/ILLEGAL INTRO.
  
```

.SBTTL
 .SBTTL SECOND MODULE TESTS
 .SBTTL

 :*TEST 43 *TEST THAT MODULE PASSES 'BIAKI'

 TST43: SCOPE

;*WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
 ;*SECOND MODULE (IBV-11) WITH SWITCH 'ER1 INH' SET.
 ;*ADDRESS OF THE SECOND MODULE IS IN LOCATION 'IBS2' VECTOR
 ;*ADDRESS IS IN LOCATION 'VECTA2'. THE SECOND IBV-11 SHOULD BE ELECTRICALLY SEC
 ;*TO INHIBIT THE USE OF TESTING WITH A SECOND MODULE, MAKE
 ;*LOCATION '\$CDW1' ZERO.
 ;*

```

6504 (1)
6505 006154 005737 001254 TST $CDW1 ;TESTING WITH
6506 006160 001002 BNE 3$ ;SECOND IBV11?
6507 006162 000137 006764 JMP EOP ;NO-END PASS.
6508 006166 3$:
6509
6510 006166 005077 173154 CLR @IBS ;CLEAR CSR.
6511 006172 005077 173170 CLR @IBS2 ;CLEAR SECOND MODULE.
6512 006176 012777 000200 173212 MOV #200,@PRC2
  
```

```

6513 006204 012777 006242 173164 MOV #1$,@VECTC2 ;SET UP VECTOR ADDR.
6514 ;/PR
(1) 006212 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 006216 012746 006224 MOV #64$,-(SP) ;/SHOW RETURN ADDRESS
(1) 006222 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
(1) 006224 ;/IN STATUS REG.)
6515 006224 012777 000140 173134 64$: MOV #BIT6!BIT5,@IBS2 ;SET INTR ENABLE AND TON ON SECONUD
6516 006232 000240 NOP ;IBV - SHOULD CAUSE A TKR INTERRUPT.
6517 006234 000240 NOP
6518
  
```

:::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 006236 104001 ERROR 1 ;/MODULE FAULT DETECTED:
6519 ;ASSUMING SECONUD MODULE IS GOOD,
6520 ;MODULE (IBV-11) UNDER TEST FAILED
6521 ;TO PASS Q BUSS SIGNAL 'BTAKI"
  
```

:::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6523 006240 000402 BR 2$
6524 006242 1$: ADD #4,SP ;/ADD #4 TO STACK POINTER.
(1) 006242 062706 000004 17$: CLR @IBS2 ;CLEAR SECONUD MODULE
6525 006246 005077 173114 2$: ;/-RESV-
6526 (1) 006252 013777 001416 173116 MOV PRC2,@VECTC2 ;/RESTORE VECTOR FOR
(1) 006260 012777 004700 173130 MOV #4700,@PRC2 ;/ILLEGAL INTRO.
6527
6528
  
```

 *TEST 44 *TEST THAT SRQ CAN GENERATE AN INTERRUPT

 TST44: SCOPE

;*WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
 ;*SECONUD MODULE (IBV-11) WITH SWITCH 'ER1 INH' SET.
 ;*ADDRESS OF THE SECONUD MODULE IS IN LOCATION "IBS2" VECTOR
 ;*ADDRESS IS IN LOCATION "VECTA2". THE SECONUD IBV-11 SHOULD BE ELECTRICALLY SEC
 ;*TO INHIBIT THE USE OF TESTING WITH A SECONUD MODULE, MAKE
 ;*LOCATION "\$CDW1" ZERO.
 ;*

```

6530
6531 006270 005077 173052 CLR @IBS ;CLEAR CSRS.
6532 006274 005077 173066 CLR @IBS2
6533
6534 006300 012777 000200 173076 MOV #200,@PRB ;SET UP INTERRUPT VECTOR.
6535 006306 012777 006352 173044 MOV #1$,@VECTB
6536 ;/PR
(1) 006314 012746 000000 MOV #0,-(SP) ;/SET CPU PRIORITY ON RETURN
(1) 006320 012746 006326 MOV #64$,-(SP) ;/SHOW RETURN ADDRESS
(1) 006324 000002 RTI ;/CAUSE A RETURN (PUTS NEW STATUS
(1) 006326 ;/IN STATUS REG.)
6537 006326 012777 000100 173012 64$: MOV #100,@IBS ;ENABLE INTERRUPTS
6538 006334 052777 100000 173024 BIS #BIT15,@IBS2 ;SETTING SRQ IN THE "CDW" MODULE
6539 ;WILL PUT SRQ ON THE IB BUS
6540 ;IS ER1 INH SW IS SET.
6541 006342 000240 NOP
  
```

6542 006344 000240
6543

```

NOP
:::$$$$$$$$$>>> ERROR <<<$$$$$$$$$

```

(1)
(1) 006346 104001
6544
6545

```

ERROR 1 ;/MODULE FAULT DETECTED:
;SRQ FAILED TO GENERATE
;AN INTERRUPT

```

6547 006350 000402
6548
6549 006352
(1) 006352 062706 000004
6550

```

:::$$$$$$$$$^^^ ERROR ^^$$$$$$$$$$
BR 2$
1$: ADD #4,SP ;/ADD #4 TO STACK POINTER.

```

6551 006356
(1)
(1) 006356 013777 001404 172774
(1) 006364 012777 004700 173012
6552 006372 005077 172750
6553 006376 005077 172764
6554

```

2$:
;/-RESV-
MOV PRB,@VECTB ;/RESTORE VECTOR FOR
MOV #4700,@PRB ;/ILLEGAL INTRO.
CLR @IBS ;CLEAR CSRS
CLR @IBS2

```

6555
(3)
(3)
(2) 006402 000004
(2)

```

:::*****
;*TEST 45 *TEST THAT ERROR1 IS GENERATED IF ATN IS ON THE IB BUS
:::*****
TST45: SCOPE

```

6556
6557
(1)
(1)
(1)
(1)
(1)
(1)

```

;*WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
;*SECOUND MODULE (IBV-11) WITH SWITCH 'ER1 INH' SET.
;*ADDRESS OF THE SECOUND MODULE IS IN LOCATION 'IBS2' VECTOR
;*ADDRESS IS IN LOCATION 'VECTA2'. THE SECOUND IBV-11 SHOULD BE ELECTRICALLY SEC
;*TO INHIBIT THE USE OF TESTING WITH A SECOUND MODULE, MAKE
;*LOCATION '$CDW1' ZERO.
;*

```

6558
6559 006404 005077 172756
6560 006410 005277 172752
6561
6562
6563
6564 006414 032777 020000 172724
6565 006422 001001
6566

```

CLR @IBS2 ;CLR CSR OF 2ND MODULE.
INC @IBS2 ;ASSERT ATN ON IB BUS
;ASSERTED ATN ON IBV UNDER TEST-
;THIS SHOULD CAUSE AN ERROR 1
;SENCE THE 2ND IBV HAS ATN SET.
;DID ERROR 1 SET?
BIT #BIT13,@IBS
BNE TST46

```

```

:::$$$$$$$$$>>> ERROR <<<$$$$$$$$$

```

(1)
(1) 006424 104001
6567

```

ERROR 1 ;/MODULE FAULT DETECTED:
;FAILED TO GENERATE ERROR 1

```

```

:::$$$$$$$$$^^^ ERROR ^^$$$$$$$$$$

```

6569
6570
(3)

```

:::*****
;*TEST 46 *TEST THAT ERROR 1 IS GENERATED IF IFC IS PUT ON IB BUS BY SECOUND MODUL

```

```

(3)
(2) 006426 000004
(2)
(1) 006430 012737 000005 001160
6571 MOV #5,$TIMES ;;DO 5 ITERATIONS
(1) ;*WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
(1) ;*SECOUND MODULE (IBV-11) WITH SWITCH "ER1 INH" SET.
(1) ;*ADDRESS OF THE SECOUND MODULE IS IN LOCATION "IBS2" VECTOR
(1) ;*ADDRESS IS IN LOCATION "VECTA2". THE SECOUND IBV-11 SHOULD BE ELECTRICALLY SEC
(1) ;*TO INHIBIT THE USE OF TESTING WITH A SECOUND MODULE, MAKE
(1) ;*LOCATION "$CDW1" ZERO.
(1)
6572 006436 005077 172704 CLR @IBS ;CLEAR CSR
6573 006442 012777 000010 172716 MOV #BIT3,@IBS2 ;ASSERT IFC FROM TESTOR
6574 006450 032777 020000 172670 BIT #BIT13,@IBS ;DID ERROR 1 GET SET?
6575 ;IF SO - NEXT TEST
6576 006456 001010 BNE TST47
6577 006460 012777 000010 172700 MOV #BIT3,@IBS2 ;IF NOT WE'LL TRY AGAIN SENCE MEMORY
6578 006466 032777 020000 172652 BIT #BIT13,@IBS ;REFRESH COULD HAVE GO IN THE WAY.
6579 006474 001001 BNE TST47
6580

```

;;\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```

(1)
(1) 006476 104001
6581 ERROR 1 ;/MODULE FAULT DETECTED:
6582 ;ERROR 1 FAILED TO SET WHEN
6583 ;IFC WAS ON IB-BUS AND MODULE
;UNDER TEST DIDN'T PUT IT THERE.

```

;;\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```

6585
6586 ;*TEST 47 *TEST THAT ERROR 1 IS GENERATED IF REN IS ON IB BUS
(3)
(3)
(2) 006500 000004
(2)
6587
6588 ;*WARNING! THIS TEST IS DESIGNED TO BE EXERCISED WITH A
(1) ;*SECOUND MODULE (IBV-11) WITH SWITCH "ER1 INH" SET.
(1) ;*ADDRESS OF THE SECOUND MODULE IS IN LOCATION "IBS2" VECTOR
(1) ;*ADDRESS IS IN LOCATION "VECTA2". THE SECOUND IBV-11 SHOULD BE ELECTRICALLY SEC
(1) ;*TO INHIBIT THE USE OF TESTING WITH A SECOUND MODULE, MAKE
(1) ;*LOCATION "$CDW1" ZERO.
(1)
6589
6590 006502 005077 172640 CLR @IBS ;CLEAR CSRS.
6591 006506 005077 172654 CLR @IBS2
6592 006512 052777 000004 172646 BIS #BIT2,@IBS2 ;ASSERT REN ON IB BUS FROM 2ND
6593 ;MODULE. 1ST IBV-11 SHOULD
6594 006520 032777 020000 172620 BIT #BIT13,@IBS ;GENERATE AN ERROR 1;DID IT??
6595 006526 001001 BNE TST50
6596

```

;;\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

(1)

(3) ;*TEST 51 *TEST THAT DATA CAN BE XFERRED BETWEEN THE MODULE UNDER TEST AND THE KGM
(4) ;* NOTE: KGM =KNOWN GOOD MODULE
(4) ;*IN THIS TEST WE'LL MAKE THE KGM A LISTENER
(4) ;* AND THE MODULE UNDER TEST A TALKER.
(4) ;*WE'VE ALREADY XFERRED DATA TO AND FROM THE IB-BUS
(4) ;*VIA THE MODULE UNDER TEST. THE ONLY UNKNOWN
(4) ;*IS THE CABLE CONNECTING THE KGM TO THE MODULE UNDER TEST,
(4) ;*AS WELL AS THE KGM.
(4) ;*

(3) ;*****
(2) 006646 000004 TST51: SCOPE

```
6637  
6638 006650 012737 006670 001110 MOV #1,$LPERR ;SET ERROR LOOP.  
6639 006656 012737 000000 001124 MOV #0,$GDDAT ;START PATTERN.  
6640 006664 005037 001126 CLR $BDDAT  
6641  
6642 006670 005077 172452 1$: CLR @IBS ;CLEAR CSRS.  
6643 006674 005077 172466 CLR @IBS2  
6644 006700 052777 000041 172440 BIS #BIT5!BIT0,@IBS ;SET TON AND TCS.  
6645 006706 052777 000020 172452 BIS #BIT4,@IBS2 ;SET LON ON KGM.  
6646 006714 013777 001124 172426 MOV $GDDAT,@IBD ;SEND PATTERN.  
6647 006722 117737 172442 001126 MOVB @IBD2,$BDDAT ;READ ATA FROM KGM.  
6648 006730 123737 001124 001126 CMPB $GDDAT,$BDDAT ;DATA SENT = DATA RECEIVED?  
6649 006736 001402 BEQ 2$ ;YES, CONTINUE  
6650
```

::\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$

```
(1)  
(1) 006740 104004 ERROR 4 ;/MODULE FAULT DETECTED:  
6651 ;ERROR - BAD DATA PASSED BETWEEN  
6652 ;MODULE UNDER TEST AND KGM.
```

::\$\$\$\$\$\$\$\$\$^^^ ERROR ^^\$\$\$\$\$\$\$\$\$\$

```
6654 006742 000407 BR TST52 ;;  
6655  
6656 006744 105237 001124 2$: INCB $GDDAT ;CHANGE PATTERN.  
6657 006750 001347 BNE 1$ ;IF NOT DONE, CONTINUE.  
6658  
6659 006752 005077 172370 CLR @IBS ;CLEAR CSR'S  
6660 006756 005077 172404 CLR @IBS2  
6661  
6662
```

::*****
;*TEST 52 *TEMP END OF TESTS
;*****
TST52: SCOPE

```
(3)  
(3)  
(2) 006762 000004  
(2)  
6663  
6664 006764 EOP:  
6665  
6666 .SBTTL SYSMAC ROUTINES:  
6667  
6668 .SBTTL END OF PASS ROUTINE  
(1)
```

::*****

(2)

```
(1) ;*INCREMENT THE PASS NUMBER ($PASS)
(1) ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
(1) ;*IF THERES A MONITOR GO TO IT
(1) ;*IF THERE ISN'T JUMP TO RSTART
(1)
(1) 006764 $EOP:
(1) 006764 000004 SCOPE
(1) 006766 005037 001102 CLR $STNM ;;ZERO THE TEST NUMBER
(1) 006772 005037 001160 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
(1) 006776 005237 001202 INC $PASS ;;INCREMENT THE PASS NUMBER
(1) 007002 042737 100000 001202 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
(1) 007010 005327 DEC (PC)+ ;;LOOP?
(1) 007012 000001 $EOPCT: .WORD 1
(1) 007014 003022 BGT $DOAGN ;;YES
(1) 007016 012737 MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
(1) 007020 000001 $ENDCT: .WORD 1
(1) 007022 007012 $EOPCT
(1) 007024 104401 007071 TYPE ,SENDMG ;;TYPE 'END PASS #'
(2) 007030 013746 001202 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
(2) 007034 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1) 007036 104401 007066 TYPE ,SENUL ;;TYPE A NULL CHARACTER
(1) 007042 013700 000042 $GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
(1) 007046 001405 BEQ $DOAGN ;;BRANCH IF NO MONITOR
(1) 007050 000005 RESET ;;CLEAR THE WORLD
(1) 007052 004710 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
(1) 007054 000240 NOP ;;SAVE ROOM
(1) 007056 000240 NOP ;;FOR
(1) 007060 000240 NOP ;;ACT11
(1) 007062 $DOAGN:
(1) 007062 000137 JMP @ (PC)+ ;;RETURN
(1) 007064 002266 $RTNAD: .WORD RSTART
(1)
(1) 007066 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
(1) 007071 015 042412 042116 $ENDMG: .ASCIZ <15><12>/END PASS #/
(1) 007076 050040 051501 020123
(1) 007104 000043
6669 ;/DELMA
(1)
(1) ;/ROUTINE TO PROVIDE DELAYS IN INCREMENTS OF 25 US
(1) ;/
(1) ;/ CALL= JSR R5,DEL50
(1) ;/ .WORD X (# OF 25 US TO DELAY)
(1) ;/ ;/ RETURNS HERE
(1) ;/
(1)
(1) 007106 012500 DEL50: MOV (5)+,R0 ;/GET # OF 25 US DELAYS
(1) 007110 012701 000002 1$: MOV #2.,R1 ;/# FOR LOOP TO DO 50 US.
(1) 007114 005301 2$: DEC R1 ;/DEC IT
(1) 007116 001376 BNE 2$ ;/WAITED 25. TIMES?
(1) 007120 005300 DEC R0 ;/DONE # OF 50 US DELAY DESIRED?
(1) 007122 001372 BNE 1$ ;/NO - NEXT ONE.
(1) 007124 000205 RTS R5 ;/YES - EXIT.
(1)
6670 .SBTTL ERROR HANDLER ROUTINE
(1)
```



```

(1) 007314 104401 001171      TYPE      ,SCLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 007320 010046      MOV      RO,-(SP)   ;;SAVE RO
(1) 007322 005000      CLR      RO        ;;PICKUP THE ITEM INDEX
(1) 007324 153700 001114      BISB     @#$ITEMB,RO
(1) 007330 001004      BNE     1$        ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                ;;TYPE THE PC OF THE ERROR
(2) 007332 013746 001116      MOV      $ERRPC,-(SP)
(2)                                ;;SAVE $ERRPC FOR TIMEOUT
(2)                                ;;ERROR ADDRESS
(2) 007336 104402      TYP0C                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 007340 000426      BR      6$        ;;GET OUT
(1) 007342 005300      1$: DEC      RO        ;;ADJUST THE INDEX SO THAT IT WILL
(1) 007344 006300      ASL     RO        ;;WORK FOR THE ERROR TABLE
(1) 007346 006300      ASL     RO
(1) 007350 006300      ASL     RO
(1) 007352 062700 001256      ADD     #$ERRTB,RO  ;;FORM TABLE POINTER
(1) 007356 012037 007366      MOV     (RO)+,2$   ;;PICKUP 'ERROR MESSAGE' POINTER
(1) 007362 001404      BEQ     3$        ;;SKIP TIMEOUT IF NO POINTER
(1) 007364 104401      TYPE                    ;;TYPE THE 'ERROR MESSAGE'
(1) 007366 000000      2$: .WORD  0        ;;'ERROR MESSAGE' POINTER GOES HERE
(1) 007370 104401 001171      TYPE      ,SCLF     ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 007374 012037 007404      3$: MOV     (RO)+,4$   ;;PICKUP 'DATA HEADER' POINTER
(1) 007400 001404      BEQ     5$        ;;SKIP TIMEOUT IF 0
(1) 007402 104401      TYPE                    ;;TYPE THE 'DATA HEADER'
(1) 007404 000000      4$: .WORD  0        ;;'DATA HEADER' POINTER GOES HERE
(1) 007406 104401 001171      TYPE      ,SCLF     ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 007412 011000      5$: MOV     (RO),RO   ;;PICKUP 'DATA TABLE' POINTER
(1) 007414 001004      BNE     7$        ;;GO TYPE THE DATA
(1) 007416 012600      6$: MOV     (SP)+,RO  ;;RESTORE RO
(1)                                ;;
(1) 007420 104401 001171      TYPE      ,SCLF     ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 007424 000207      RTS     PC        ;;RETURN
(1)                                ;;
(2) 007426 013046      7$: MOV     @(RO)+,-(SP) ;;SAVE @(RO)+ FOR TIMEOUT
(2) 007430 104402      TYP0C                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 007432 005710      TST     (RO)       ;;IS THERE ANOTHER NUMBER?
(1) 007434 001770      BEQ     6$        ;;BR IF NO
(1) 007436 104401 007444      TYPE     8$        ;;TYPE TWO(2) SPACES
(1) 007442 000771      BR      7$        ;;LOOP
(1) 007444 020040 000      8$: .ASCIZ  / /      ;;TWO(2) SPACES
(1) 007450      .EVEN
6672 .SBTTL SCOPE HANDLER ROUTINE STARS
(1) ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1) ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1) ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1) ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1) ;*SW14=1 LOOP ON TEST
(1) ;*SW11=1 INHIBIT ITERATIONS
(1) ;*SW09=1 LOOP ON ERROR
(1) ;*SW08=1 LOOP ON TEST IN SWR<7:0>
(1) ;*CALL
(1) ;* SCOPE ;;SCOPE=IOT
(1) $SCOPE:
(1) 007450      CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
(1) 007452 104407      CKSWR
(2) 007454 104407      BIT    #BIT14,@SWR ;;LOOP ON PRESENT TEST?
(1) 007454 032777 040000 171456 1$:

```

```
(1) 007462 001114          BNE      $OVER          ;; YES IF SW14=1
(1)                      :##### START OF CODE FOR THE XOR TESTER#####
(1) 007464 000416          $XTSTR: BR      6$      ;; IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                      ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
(1) 007466 013746 000004          MOV      @#ERRVEC, -(SP) ;; SAVE THE CONTENTS OF THE ERROR VECTOR
(1) 007472 012737 007512 000004          MOV      #5$, @#ERRVEC  ;; SET FOR TIMEOUT
(1) 007500 005737 177060          TST      @#177060      ;; TIME OUT ON XOR?
(1) 007504 012637 000004          MOV      (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
(1) 007510 000463          BR      $SVLAD        ;; GO TO THE NEXT TEST
(1) 007512 022626          5$:      CMP      (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
(1) 007514 012637 000004          MOV      (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
(1) 007520 000423          BR      7$          ;; LOOP ON THE PRESENT TEST
(1) 007522          6$: :##### END OF CODE FOR THE XOR TESTER#####
(1) 007522 032777 000400 171410          BIT      #BIT08, @SWR   ;; LOOP ON SPEC. TEST?
(1) 007530 001404          BEQ      2$          ;; BR IF NO
(1) 007532 127737 171402 001102          CMPB    @SWR, $STNM    ;; ON THE RIGHT TEST?   SWR<7:0>
(1) 007540 001465          BEQ      $OVER       ;; BR IF YES
(1) 007542 105737 001103          2$:      TSTB    $ERFLG    ;; HAS AN ERROR OCCURRED?
(1) 007546 001421          BEQ      3$          ;; BR IF NO
(1) 007550 123737 001115 001103          CMPB    $ERMAX, $ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
(1) 007556 101015          BHI      3$          ;; BR IF NO
(1) 007560 032777 001000 171352          BIT      #BIT09, @SWR   ;; LOOP ON ERROR?
(1) 007566 001404          BEQ      4$          ;; BR IF NO
(1) 007570 013737 001110 001106          7$:      MOV      $LPERR, $LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
(1) 007576 000446          BR      $OVER
(1) 007600 105037 001103          4$:      CLRB    $ERFLG    ;; ZERO THE ERROR FLAG
(1) 007604 005037 001160          CLR     $TIMES      ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1) 007610 000415          BR      1$          ;; ESCAPE TO THE NEXT TEST
(1) 007612 032777 004000 171320          3$:      BIT      #BIT11, @SWR   ;; INHIBIT ITERATIONS?
(1) 007620 001011          BNE     1$          ;; BR IF YES
(1) 007622 005737 001202          TST     $PASS      ;; IF FIRST PASS OF PROGRAM
(1) 007626 001406          BEQ     1$          ;; INHIBIT ITERATIONS
(1) 007630 005237 001104          INC     $ICNT      ;; INCREMENT ITERATION COUNT
(1) 007634 023737 001160 001104          CMP     $TIMES, $ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
(1) 007642 002024          BGE     $OVER      ;; BR IF MORE ITERATION REQUIRED
(1) 007644 012737 000001 001104          1$:      MOV     #1, $ICNT   ;; REINITIALIZE THE ITERATION COUNTER
(1) 007652 013737 007730 001160          MOV     $MXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
(1) 007660 105237 001102          $SVLAD: INCB    $STNM   ;; COUNT TEST NUMBERS
(1) 007664 113737 001102 001200          MOVB   $STNM, $TESTN ;; SET TEST NUMBER IN APT MAILBOX
(1) 007672 011637 001106          MOV     (SP), $LPADR ;; SAVE SCOPE LOOP ADDRESS
(1) 007676 011637 001110          MOV     (SP), $LPERR ;; SAVE ERROR LOOP ADDRESS
(1) 007702 005037 001162          CLR     $ESCAPE    ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
(1) 007706 112737 000001 001115          MOVB   #1, $ERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1) 007714 013777 001102 171220          $OVER:  MOV     $STNM, @DISPLAY ;; DISPLAY TEST NUMBER
(1) 007722 013716 001106          MOV     $LPADR, (SP) ;; FUDGE RETURN ADDRESS
(1) 007726 000002          RTI
(1) 007730 003720          $MXCNT: 2000.      ;; FIXES PS
(1)                      ;; MAX. NUMBER OF ITERATIONS
```

6673

(1)
(2)
(1)
(1)
(1)
(1)
(2)
(1)

```
.SBTTL TTY INPUT ROUTINE
:*****
.ENABL LSB
:*****
*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
```

```

(1) ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1) ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1) ;*WHEN OPERATING IN TTY FLAG MODE.
(1) 007732 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;: IS THE SOFT-SWR SELECTED?
(1) 007740 001074 BNE 15$ ;: BRANCH IF NO
(1) 007742 105777 171176 TSTB @STKS ;: CHAR THERE?
(1) 007746 100071 BPL 15$ ;: IF NO, DON'T WAIT AROUND
(1) 007750 117746 171172 MOVB @STKB,-(SP) ;: SAVE THE CHAR
(1) 007754 042716 177600 BIC #^C177,(SP) ;: STRIP-OFF THE ASCII
(1) 007760 022726 000007 CMP #7,(SP)+ ;: IS IT A CONTROL G?
(1) 007764 001062 BNE 15$ ;: NO, RETURN TO USER
(1) 007766 123727 001134 000001 CMPB $AUTOB,#1 ;: ARE WE RUNNING IN AUTO-MODE?
(1) 007774 001456 BEQ 15$ ;: BRANCH IF YES
(1) 007776 104401 010457 TYPE ,SCNTLG ;: ECHO THE CONTROL-G (^G)
(1) 010002 104401 010464 $GTSWR: TYPE ,SMSWR ;: TYPE CURRENT CONTENTS
(2) 010006 013746 000176 MOV SWREG,-(SP) ;: SAVE SWREG FOR TYPEOUT
(2) 010012 104402 TYPOC ;: GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 010014 104401 010475 TYPE ,SMNEW ;: PROMPT FOR NEW SWR
(1) 010020 005046 19$: CLR -(SP) ;: CLEAR COUNTER
(1) 010022 005046 CLR -(SP)
(1) 010024 105777 171114 7$: TSTB NEW SWR @STKS ;: CHAR THERE?
(1) 010030 100375 BPL 7$ ;: IF NOT TRY AGAIN
(1) 010032 117746 171110 MOVB @STKB,-(SP) ;: PICK UP CHAR
(1) 010036 042716 177600 BIC #^C177,(SP) ;: MAKE IT 7-BIT ASCII
(1)
(1)
(1) 010042 021627 000025 9$: CMP (SP),#25 ;: IS IT A CONTROL-U?
(1) 010046 001005 BNE 10$ ;: BRANCH IF NOT
(1) 010050 104401 010452 TYPE ,SCNTLU ;: YES, ECHO CONTROL-U (^U)
(1) 010054 062706 000006 20$: ADD #6,SP ;: IGNORE PREVIOUS INPUT
(1) 010060 000757 BR 19$ ;: LET'S TRY IT AGAIN
(1)
(1)
(1) 010062 021627 000015 10$: CMP (SP),#15 ;: IS IT A <CR>?
(1) 010066 001022 BNE 16$ ;: BRANCH IF NO
(1) 010070 005766 000004 TST 4(SP) ;: YES, IS IT THE FIRST CHAR?
(1) 010074 001403 BEQ 11$ ;: BRANCH IF YES
(1) 010076 016677 000002 171034 MOV 2(SP),@SWR ;: SAVE NEW SWR
(1) 010104 062706 000006 11$: ADD #6,SP ;: CLEAR UP STACK
(1) 010110 104401 001171 14$: TYPE ,SCRLF ;: ECHO <CR> AND <LF>
(1) 010114 123727 001135 000001 CMPB $INTAG,#1 ;: RE-ENABLE TTY KBD INTERRUPTS?
(1) 010122 001003 BNE 15$ ;: BRANCH IF NOT
(1) 010124 012777 000100 171012 MOV #100,@STKS ;: RE-ENABLE TTY KBD INTERRUPTS
(1) 010132 000002 15$: RTI ;: RETURN
(1) 010134 004737 011372 16$: JSR PC,$TYPEC ;: ECHO CHAR
(1) 010140 021627 000060 CMP (SP),#60 ;: CHAR < 0?
(1) 010144 002420 BLT 18$ ;: BRANCH IF YES
(1) 010146 021627 000067 CMP (SP),#67 ;: CHAR > 7?
(1) 010152 003015 BGT 18$ ;: BRANCH IF YES
(1) 010154 042726 000060 BIC #60,(SP)+ ;: STRIP-OFF ASCII
(1) 010160 005766 000002 TST 2(SP) ;: IS THIS THE FIRST CHAR
(1) 010164 001403 BEQ 17$ ;: BRANCH IF YES

```



```

(1) 010366 000763          BR      1$          ::CLEAR THE BUFFER AND LOOP
(1) 010370 111337 010440 3$:  MOVB   (R3),9$      ::ECHO THE CHARACTER
(1) 010374 104401 010440      TYPE   ,9$
(1) 010400 122723 000015      CMPB   #15,(R3)+    ::CHECK FOR RETURN
(1) 010404 001356          BNE    2$          ::LOOP IF NOT RETURN
(1) 010406 105063 177777      CLRB  -1(R3)       ::CLEAR RETURN (THE 15)
(1) 010412 104401 001172      TYPE   ,SLF        ::TYPE A LINE FEED
(1) 010416 012603          MOV    (SP)+,R3     ::RESTORE R3
(1) 010420 011646          MOV    (SP),-(SP)   ::ADJUST THE STACK AND PUT ADDRESS OF THE
(1) 010422 016666 000004 000002 MOV    4(SP),2(SP)  ::FIRST ASCII CHARACTER ON IT
(1) 010430 012766 010442 000004 MOV    #STTYIN,4(SP)
(1) 010436 000002          RTI          ::RETURN
(1) 010440 000          9$:  .BYTE  0          ::STORAGE FOR ASCII CHAR. TO TYPE
(1) 010441 000          .BYTE  0          ::TERMINATOR
(1) 010442 000010          $TTYIN: .BLKB 8.    ::RESERVE 8 BYTES FOR TTY INPUT
(1) 010452 052536 005015 000  $CNTLU: .ASCIZ /^U/<15><12> ::CONTROL 'U'
(1) 010457 136 006507 000012 $CNTLG: .ASCIZ /^G/<15><12> ::CONTROL 'G'
(1) 010464 005015 053523 020122 $MSWR:  .ASCIZ <15><12>/SWR = /
(1) 010472 020075 000
(1) 010475 040 047040 053505 $MNEW:  .ASCIZ / NEW = /
(1) 010502 036440 000040
  
```


| | | | | | | | |
|-----|--------|--------|---------------|----------|-------|-------------|-----------------------------------|
| (1) | 010650 | 005204 | | 4\$: | INC | R4 | ::DON'T SUPPRESS ANYMORE 0'S |
| (1) | 010652 | 052703 | 000060 | | BIS | #'0,R3 | ::MAKE THIS DIGIT ASCII |
| (1) | 010656 | 052703 | 000040 | 5\$: | BIS | #',R3 | ::MAKE ASCII IF NOT ALREADY |
| (1) | 010662 | 110337 | 010726 | | MOVB | R3,8\$ | ::SAVE FOR TYPING |
| (1) | 010666 | 104401 | 010726 | | TYPE | ,8\$ | ::GO TYPE THIS DIGIT |
| (1) | 010672 | 105337 | 010730 | 7\$: | DECB | \$OCNT | ::COUNT BY 1 |
| (1) | 010676 | 003347 | | | BGT | 2\$ | ::BR IF MORE TO DO |
| (1) | 010700 | 002402 | | | BLT | 6\$ | ::BR IF DONE |
| (1) | 010702 | 005204 | | | INC | R4 | ::INSURE LAST DIGIT ISN'T A BLANK |
| (1) | 010704 | 000744 | | | BR | 2\$ | ::GO DO THE LAST DIGIT |
| (1) | 010706 | 012605 | | 6\$: | MOV | (SP)+,R5 | ::RESTORE R5 |
| (1) | 010710 | 012604 | | | MOV | (SP)+,R4 | ::RESTORE R4 |
| (1) | 010712 | 012603 | | | MOV | (SP)+,R3 | ::RESTORE R3 |
| (1) | 010714 | 016666 | 000002 000004 | | MOV | 2(SP),4(SP) | ::SET THE STACK FOR RETURNING |
| (1) | 010722 | 012616 | | | MOV | (SP)+,(SP) | |
| (1) | 010724 | 000002 | | | RTI | | ::RETURN |
| (1) | 010726 | 000 | | 8\$: | .BYTE | 0 | ::STORAGE FOR ASCII DIGIT |
| (1) | 010727 | 000 | | | .BYTE | 0 | ::TERMINATOR FOR TYPE ROUTINE |
| (1) | 010730 | 000 | | \$OCNT: | .BYTE | 0 | ::OCTAL DIGIT COUNTER |
| (1) | 010731 | 000 | | \$OFILL: | .BYTE | 0 | ::ZERO FILL SWITCH |
| (1) | 010732 | 000000 | | \$OMODE: | .WORD | 0 | ::NUMBER OF DIGITS TO TYPE |


```

(1) 011300 001171          $CRLF
(1) 011302 105037 011436  CLR  $CHARCNT      ;;CLEAR CHARACTER COUNT
(1) 011306 000755          BR    2$            ;;GET NEXT CHARACTER
(1) 011310 004737 011372  5$:  JSR  PC,$TYPEC   ;;GO TYPE THIS CHARACTER
(1) 011314 123726 001156  6$:  CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
(1) 011320 001350          BNE   2$            ;;IF NO GO GET NEXT CHAR.
(1) 011322 013746 001154  MOV   $NULL,-(SP)   ;;GET # OF FILLER CHARS. NEEDED
(1)                                ;;AND THE NULL CHAR.
(1) 011326 105366 000001  7$:  DECB 1(SP)      ;;DOES A NULL NEED TO BE TYPED?
(1) 011332 002770          BLT   6$            ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1) 011334 004737 011372  JSR  PC,$TYPEC   ;;GO TYPE A NULL
(1) 011340 105337 011436  DECB  $CHARCNT    ;;DO NOT COUNT AS A COUNT
(1) 011344 000770          BR    7$            ;;LOOP
  
```

;;HORIZONTAL TAB PROCESSOR

```

(1) 011346 112716 000040  8$:  MOV  #'(SP)      ;;REPLACE TAB WITH SPACE
(1) 011352 004737 011372  9$:  JSR  PC,$TYPEC   ;;TYPE A SPACE
(1) 011356 132737 000007 011436  BITB #7,$CHARCNT   ;;BRANCH IF NOT AT
(1) 011364 001372          BNE   9$            ;;TAB STOP
(1) 011366 005726          TST  (SP)+         ;;POP SPACE OFF STACK
(1) 011370 000724          BR    2$            ;;GET NEXT CHARACTER
(1) 011372 105777 167552  $TYPEC: TSTB @ $TPS  ;;WAIT UNTIL PRINTER IS READY
(1) 011376 100375          BPL  $TYPEC
(1) 011400 116677 000002 167544  MOV  2(SP),@$TPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 011406 122766 000015 000002  CMPB #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
(1) 011414 001003          BNE   1$            ;;BRANCH IF NO
(1) 011416 105037 011436  CLR  $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
(1) 011422 000406          BR    $TYPEX       ;;EXIT
(1) 011424 122766 000012 000002  1$:  CMPB #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
(1) 011432 001402          BEQ  $TYPEX       ;;BRANCH IF YES
(1) 011434 105227          INCB (PC)+        ;;COUNT THE CHARACTER
(1) 011436 000000          $CHARCNT: WORD 0  ;;CHARACTER COUNT STORAGE
(1) 011440 000207          $TYPEX: RTS      PC
  
```

;;SBTTL APT COMMUNICATIONS ROUTINE

6680

```

(1)
(1)
(2)
(1) 011442 112737 000001 011706  $ATY1: MOV  #1,$FFLG  ;;TO REPORT FATAL ERROR
(1) 011450 112737 000001 011704  $ATY3: MOV  #1,$MFLG  ;;TO TYPE A MESSAGE
(1) 011456 000403          BR    $ATYC
(1) 011460 112737 000001 011706  $ATY4: MOV  #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
(1) 011466          $ATYC:
(3) 011466 010046          MOV  R0,-(SP)      ;;PUSH R0 ON STACK
(3) 011470 010146          MOV  R1,-(SP)      ;;PUSH R1 ON STACK
(1) 011472 105737 011704          TSTB $MFLG        ;;SHOULD TYPE A MESSAGE?
(1) 011476 001450          BNE  5$            ;;IF NOT: BR
(1) 011500 122757 000001 001274  CMPB #APTENV,$ENV  ;;OPERATING UNDER APT?
(1) 011506 001031          BNE  3$            ;;IF NOT: BR
(1) 011510 132737 000100 001215  BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 011516 001425          BEQ  3$            ;;IF NOT: BR
(1) 011520 017600 000004          MOV  @4(SP),R0    ;;GET MESSAGE ADDR.
(1) 011524 002766 000002 000004  ADD  #2,4(SP)      ;;BUMP RETURN ADDR.
  
```

```

(1) 011532 005737 001174 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
(1) 011536 001375 BNE 1$ ;;IF NOT: WAIT
(1) 011540 010037 001210 MOV R0,$MSGAD
(1) ;;PUT ADDR IN MAILBOX
(1) 011544 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 011546 001376 BNE 2$
(1) 011550 163700 001210 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 011554 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
(1) 011556 010037 001212 MOV R0,$MSGLGT ;;PUT LENGTH IN MAILBOX
(1) 011562 012737 000004 001174 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
(1) 011570 000413 BR 5$
(1) 011572 017637 000004 011616 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
(1) 011600 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
(3) 011606 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
(1) 011612 004737 011160 JSR PC,$TYPE ;;CALL TYPE MACRO
(1) 011616 000000 4$: .WORD 0
(1) 011620 5$:
(1) 011620 105737 011706 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
(1) 011624 001416 BEQ 12$ ;;IF NOT: BR
(1) 011626 005737 001214 TST $ENV ;;RUNNING UNDER APT?
(1) 011632 001413 BEQ 12$ ;;IF NOT: BR
(1) 011634 005737 001174 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
(1) 011640 001375 BNE 11$ ;;IF NOT: WAIT
(1) 011642 017637 000004 001176 MOV @4(SP),$FATAL ;;GET ERROR #
(1) 011650 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 011656 005237 001174 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
(1) 011662 105037 011706 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
(1) 011666 105037 011705 CLRB $LFLG ;;CLEAR LOG FLAG
(1) 011672 105037 011704 CLRB $MFLG ;;CLEAR MESSAGE FLAG
(3) 011676 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
(3) 011700 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
(1) 011702 000207 RTS PC ;;RETURN
(1) 011704 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
(1) 011705 000 $LFLG: .BYTE 0
(1) ;;LOG FLAG
(1) 011706 000 $FFLG: .BYTE 0 ;;FATAL FLAG
(1) 011710 .EVEN
(1) 000200 APTSIZE=200
(1) 000001 APTENV=001
(1) 000100 APTSPOOL=100
(1) 000040 APTCSUP=040

```

6681 .SBTTL POWER DOWN AND UP ROUTINES

```

(1) *****
(2) :POWER DOWN ROUTINE
(1) 011710 012737 012050 000024 $PWRDN: MOV #SILLUP,@#PWRVEC ;;SET FOR FAST UP
(1) 011716 012737 000300 000026 MOV #PR6,@#PWRVEC+2 ;;PRIO:6
(3) 011724 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 011726 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(3) 011730 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
(3) 011732 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
(3) 011734 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
(3) 011736 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
(3) 011740 017746 167174 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
(1) 011744 010637 012054 MOV SP,$SAVR6 ;;SAVE SP
(1) 011750 012737 011762 000024 MOV #SPPWRUP,@#PWRVEC ;;SET UP VECTOR

```

```

(1) 011756 000000          HALT
(1) 011760 000776          BR      .-2          ;;HANG UP
(1)
(2)
(1)
(1) 011762 012737 012050 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1) 011770 013706 012054          MOV    $$SAVR6,$P      ;;GET SP
(1) 011774 005037 012054          CLR    $$SAVR6        ;;WAIT LOOP FOR THE TTY
(1) 012000 005237 012054          1$:   INC    $$SAVR6        ;;WAIT FOR THE INC
(1) 012004 001375          BNE    1$             ;;OF WORD
(3) 012006 012677 167126          MOV    (SP)+,@SWR     ;;POP STACK INTO @SWR
(3) 012012 012605          MOV    (SP)+,R5      ;;POP STACK INTO R5
(3) 012014 012604          MOV    (SP)+,R4      ;;POP STACK INTO R4
(3) 012016 012603          MOV    (SP)+,R3      ;;POP STACK INTO R3
(3) 012020 012602          MOV    (SP)+,R2      ;;POP STACK INTO R2
(3) 012022 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
(3) 012024 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
(1) 012026 012737 011710 000024 MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 012034 012737 000300 000026 MOV    #PR6,@#PWRVEC+2 ;;PRIO:6
(1) 012042 104401          TYPE
(1) 012044 012056          $PWRMG: .WORD $POWER ;;REPORT THE POWER FAILURE
(1) 012046 000002          RTI                ;;POWER FAIL MESSAGE POINTER
(1) 012050 000000          $ILLUP: HALT
(1) 012052 000776          BR      .-2          ;;THE POWER UP SEQUENCE WAS STARTED
(1) 012054 000000          $$SAVR6: 0          ;;BEFORE THE POWER DOWN WAS COMPLETE
(1) 012056 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER' ;;PUT THE SP HERE
(1) 012064 000122          .EVEN
(1)
6682
6683
6684
6685
6686
6687
6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703 012066 011637 012132          IOTRD: MOV    (6),TRTO      ;;GET WHERE WE CAME TO.
6704 012072 162737 000004 012132 SUB    #4,TRTO        ;;FORM REAL ADDR.
6705
6706 012100 023727 012132 001000 CMP    TRTO,#1000     ;;DID TRAP COME FROM LESS THAN ADDR. 1000?
6707 012106 003402          BLE    2$
6708
6709 012110 000000          1$:   HALT          ;;NO! MUST BE A BUSS ILLEGAL ADDR. TIME OUT.

```

```
6710 ; ADDRESS CONTAINED IN TRTO.  
6711  
6712 012112 000776 BR 1$ ;DON'T ALLOW A CONTINUE.  
6713 012114 2$:  
6714  
6715 012114 016637 000004 012134 MOV 4(6),TRFRO ;GET TRAPPED FROM ADDR.  
6716  
6717 012122 062706 000004 ADD #4,SP ;/ADD #4 TO STACK POINTER.  
6718  
6719
```

::\$\$\$\$\$\$\$\$\$\$\$>>> ERROR <<<\$\$\$\$\$\$\$\$\$\$\$\$

```
(1)  
(1) 012126 104007 ERROR 7 ;/MODULE FAULT DETECTED:  
6720 ;ERROR! ILLEGAL INTERRUPT  
6721 ;OR INTERRUPT TO WRONG  
6722 ;VECTOR - IF TEST NUMBER  
6723 ;IS LESS THAN 10, ITS LIKELY  
6724 ;(BUT NOT EXCLUSIVELY) TO BE A  
6725 ;DEVICE OTHER THAN THE IBV-11  
6726 ;TO BLAME.  
6727 ;IF THE INTERRUPT OCCURRED  
6728 ;DURING AN INTERRUPT TEST, I'D  
6729 ;SUSPECT A PROBLEM WITH THE  
6730 ;IBV-11.  
6731 ;IF THE ADDRESS THE INTERRUPT  
6732 ;VECTOR TO IS WITHIN THE RANGE  
6733 ;OF VECTORS ASSIGNED TO THE IBV-11,  
6734 ;THEN I'D SUSPECT THE IBV-11  
6735 ;INTERRUPTED ILLEGALLY.  
6736 ;IF THE ADDRESS THE INTERRUPT  
6737 ;VECTORED TO IS OUTSIDE OF THE  
6738 ;RANGE ASSIGNED TO THE IBV-11,  
6739 ;I'D SUSPECT THAT THE  
6740 ;IBV-11 PUT THE WRONG VECTOR ON  
6741 ;THE BUSS DURING THE INTERRUPT  
6742 ;PROCESS.  
6743 ;FOR THIS ERROR - DON'T  
6744 ;USE 'LOOP ON ERROR' OPTION.  
6745 ;ALSO EXPECT THE INTERRUPT TEST TO  
6746 ;REPORT THAT THE IBV-11 DIDN'T  
6747 ;INTERRUPT.  
6748 ;FOLLOW RECOMMENDED PROCEDURE  
6749 ;IN THE DOCUMENT (ON THIS DIAGNOSTIC)  
6750 ;FOR LOOPING ON ERROR
```

::\$\$\$\$\$\$\$\$\$\$\$\$\$\$\$^ ERROR ^\$\$\$\$\$\$\$\$\$\$\$\$\$\$
RTI

```
6752 012130 000002  
6753  
6754 012132 000000 TRTO: .WORD 0 ;ADDR THAT WE INTERRUPTED TO  
6755 012134 000000 TRFRO: .WORD 0 ;ADDR THAT WE INTERRUPTED FROM.  
6756  
6757 .SBTTL TRAP DECODER  
(1)  
(2)  
:*****
```



```

(1)      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)      ;*GO TO THAT ROUTINE.
(1)
(1) 012136 010046          $TRAP:  MOV    RO,-(SP)          ;;SAVE RO
(1) 012140 016600 000002  MOV    2(SP),RO        ;;GET TRAP ADDRESS
(1) 012144 005740          TST    -(RO)           ;;BACKUP BY 2
(1) 012146 111000          MOVB   (RO),RO         ;;GET RIGHT BYTE OF TRAP
(1) 012150 006300          ASL    RO              ;;POSITION FOR INDEXING
(1) 012152 016000 012172  MOV    $TRPAD(RO),RO   ;;INDEX TO TABLE
(1) 012156 000200          RTS     RO             ;;GO TO ROUTINE
(1)
(1)
(1)      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1) 012160 011646          $TRAP2: MOV   (SP),-(SP) ;;MOVE THE PC DOWN
(1) 012162 016666 000004 000002  MOV   4(SP),2(SP)      ;;MOVE THE PSW DOWN
(1) 012170 000002          RTI                    ;;RESTORE THE PSW
(1)
(3)      .SBTTL TRAP TABLE
(3)
(3)      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)      ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)      :      ROUTINE
(3)      :      -----
(3) 012172 012160          $TRPAD: .WORD   $TRAP2          TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3) 012174 011160          $TYPE   ;;CALL=TYPE          TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3) 012176 010532          $TYPOC  ;;CALL=TYPOC         TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3) 012200 010506          $TYPOS  ;;CALL=TYPOS         TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3) 012202 010546          $TYPON  ;;CALL=TYPON         TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
(3) 012204 010734          $TYPDS  ;;CALL=TYPDS
(1)
(3) 012206 010002          $GTSWR  ;;CALL=GTSWR         TRAP+6(104406)  GET SOFT-SWR SETTING
(1)
(3) 012210 007732          $CKSWR  ;;CALL=CKSWR         TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
(3) 012212 010214          $RDCHR  ;;CALL=RDCHR         TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
(3) 012214 010334          $RDLIN  ;;CALL=RDLIN         TRAP+11(104411) TTY TYPEIN STRING ROUTINE
6758
6759
6760
6761 012216 005007 044415 051502  EM1:  .ASCIZ<7><12><15>#IBS FUNCTION ERROR#
      012224 043040 047125 052103
      012232 047511 020116 051105
      012240 047522 000122
6762
6763 012244 005007 044415 042102  EM2:  .ASCIZ<7><12><15>#IBD FUNCTION ERROR#
      012252 043040 047125 052103
      012260 047511 020116 051105
      012266 047522 000122
6764
6765 012272 005007 044415 051502  EM3:  .ASCIZ<7><12><15>#IBS DATA ERROR#
      012300 042040 052101 020101
      012306 051105 047522 000122
6766
  
```

| | | | | | | | | | | |
|------|--------|--------|--------|--------|-------|---|-------|----------|-------------|------|
| 6767 | 012314 | 005007 | 044415 | 042102 | FM4: | .ASCIZ<7><12><15>#IBD DATA ERROR# | | | | |
| | 012322 | 042040 | 052101 | 020101 | | | | | | |
| | 012330 | 051105 | 047522 | 000122 | | | | | | |
| 6768 | | | | | | | | | | |
| 6769 | 012336 | 100007 | 041111 | 027523 | EM5: | .ASCIZ<7><200>#IBS/IBD ADDRESS ERROR# | | | | |
| | 012344 | 041111 | 020104 | 042101 | | | | | | |
| | 012352 | 051104 | 051505 | 020123 | | | | | | |
| | 012360 | 051105 | 047522 | 000122 | | | | | | |
| 6770 | 012366 | 100007 | 041111 | 041527 | EM6: | .ASCIZ <7><200>#IBWC/IBCA DATA ERROR# | | | | |
| | 012374 | 044457 | 041502 | 020101 | | | | | | |
| | 012402 | 040504 | 040524 | 042440 | | | | | | |
| | 012410 | 051122 | 051117 | 000 | | | | | | |
| 6771 | | | | | | | | | | |
| 6772 | 012415 | 007 | 044600 | 052116 | EM7: | .ASCIZ <7><200>#INTERRUPT ERROR# | | | | |
| | 012422 | 051105 | 052522 | 052120 | | | | | | |
| | 012430 | 042440 | 051122 | 051117 | | | | | | |
| | 012436 | 000 | | | | | | | | |
| 6773 | | | | | | | | | | |
| 6774 | 012437 | 200 | 042524 | 052123 | DH1: | .ASCIZ<CRLF>#TEST | ERRPC | IB ADR | IBS | IBD# |
| | 012444 | 020040 | 020040 | 051105 | | | | | | |
| | 012452 | 050122 | 020103 | 020040 | | | | | | |
| | 012460 | 041111 | 040440 | 051104 | | | | | | |
| | 012466 | 020040 | 044440 | 051502 | | | | | | |
| | 012474 | 020040 | 020040 | 044440 | | | | | | |
| | 012502 | 042102 | 000 | | | | | | | |
| 6775 | | | | | | | | | | |
| 6776 | 012505 | 200 | 042524 | 052123 | DH3: | .ASCIZ<CRLF>#TEST | ERRPC | GOOD | BAD# | |
| | 012512 | 020040 | 020040 | 051105 | | | | | | |
| | 012520 | 050122 | 020103 | 020040 | | | | | | |
| | 012526 | 047507 | 042117 | 020040 | | | | | | |
| | 012534 | 020040 | 040502 | 000104 | | | | | | |
| 6777 | | | | | | | | | | |
| 6778 | 012542 | 052200 | 051505 | 020124 | DH5: | .ASCIZ <CRLF>#TEST | ERRPC | IB ADDR# | | |
| | 012550 | 020040 | 042440 | 051122 | | | | | | |
| | 012556 | 041520 | 020040 | 044440 | | | | | | |
| | 012564 | 020102 | 042101 | 051104 | | | | | | |
| | 012572 | 000 | | | | | | | | |
| 6779 | | | | | | | | | | |
| 6780 | 012573 | 200 | 042524 | 052123 | DH7: | .ASCIZ <CRLF>#TEST | ERRPC | TO | FROM ADDR.# | |
| | 012600 | 020040 | 020040 | 051105 | | | | | | |
| | 012606 | 050122 | 020103 | 020040 | | | | | | |
| | 012614 | 047524 | 020040 | 020040 | | | | | | |
| | 012622 | 020040 | 051106 | 046517 | | | | | | |
| | 012630 | 040440 | 042104 | 027122 | | | | | | |
| | 012636 | 000 | | | | | | | | |
| 6781 | | | | | | | | | | |
| 6782 | | 012640 | | | | .EVEN | | | | |
| 6783 | | | | | | | | | | |
| 6784 | 012640 | 001200 | 001116 | 001346 | DT1: | .WORD \$TESTN,\$ERRPC,IBS | | | | |
| 6785 | | | | | | | | | | |
| 6786 | 012646 | 000000 | | | IBSA: | .WORD 0 | | | | |
| 6787 | | | | | | | | | | |
| 6788 | 012650 | 000000 | 000000 | | IBDA: | .WORD 0,0 | | | | |
| 6789 | | | | | | | | | | |
| 6790 | 012654 | 001200 | 001116 | 001124 | DT3: | .WORD \$TESTN,\$ERRPC,\$GDDAT,\$BDDAT,0 | | | | |
| | 012662 | 001126 | 000000 | | | | | | | |

```

6791
6792 012666 001200 001116 001346 DT5: .WORD $TESTN,$ERRPC,IBS,0
      012674 000000
6793
6794 012676 001200 001116 012132 DT7: .WORD $TESTN,$ERRPC,TRTO,TRFRO,0
      012704 012134 000000
6795
6796 012710 000000 000000 DF0: .WORD 0,0
6797
6798
6799
6800
6801
      012714
      000100 000100
      000102 012714
      000140 000300
      000140 000140
      000142 170000
      000142 000300
      012714
      012714 104401 012722
      012720 000000
      012722 005015 045514 042526
      012730 020103 047111 042524
      012736 051122 050125 020124
      012744 020055 044504 041523
      012752 047117 042516 052103
      012760 046040 041524 000040
6802 000001
      ;THE FOLLOWING CALL TO CNMAC2.SML LIBRARY WAS ADDED TO INIT 11/21
      ;SPECIFIC VECTORS.
      POINT=. ;SAVE POINTER
      . =100
      $CLKVEC ;LKVEC HANDLER
      300 ;INTERRUPT HANDLER PRI
      . =140 ;BRKVEC
      170000 ;ODT START ADDRESS
      300 ;PRIORITY
      . =POINT ;RESTORE POINTER
      $CLKVEC: TYPE,CLKMES
      HALT
      CLKMES: .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
      .END
  
```


| | | | | | | | |
|--------|---|--------|-------|-------|-------|-------|-------|
| SW0 | = | 000001 | 5719# | | | | |
| SW00 | = | 000001 | 5719# | | | | |
| SW01 | = | 000002 | 5719# | | | | |
| SW02 | = | 000004 | 5719# | | | | |
| SW03 | = | 000010 | 5719# | | | | |
| SW04 | = | 000020 | 5719# | | | | |
| SW05 | = | 000040 | 5719# | | | | |
| SW06 | = | 000100 | 5719# | | | | |
| SW07 | = | 000200 | 5719# | | | | |
| SW08 | = | 000400 | 5719# | | | | |
| SW09 | = | 001000 | 5719# | | | | |
| SW1 | = | 000002 | 5719# | | | | |
| SW10 | = | 002000 | 5719# | | | | |
| SW11 | = | 004000 | 5719# | | | | |
| SW12 | = | 010000 | 5719# | | | | |
| SW13 | = | 020000 | 5719# | | | | |
| SW14 | = | 040000 | 5719# | | | | |
| SW15 | = | 100000 | 5719# | | | | |
| SW2 | = | 000004 | 5719# | | | | |
| SW3 | = | 000010 | 5719# | | | | |
| SW4 | = | 000020 | 5719# | | | | |
| SW5 | = | 000040 | 5719# | | | | |
| SW6 | = | 000100 | 5719# | | | | |
| SW7 | = | 000200 | 5719# | | | | |
| SW8 | = | 000400 | 5719# | | | | |
| SW9 | = | 001000 | 5719# | | | | |
| TBITVE | = | 000014 | 5719# | | | | |
| TKVEC | = | 000060 | 5719# | | | | |
| TPVEC | = | 000064 | 5719# | | | | |
| TRAPVE | = | 000034 | 5719# | 5886* | | | |
| TRFRO | | 012134 | 6715* | 6755# | 6794 | | |
| TRTO | | 012132 | 6703* | 6704* | 6706 | 6754# | 6794 |
| TRTVEC | = | 000014 | 5719# | | | | |
| TST1 | | 002444 | 5940# | | | | |
| TST10 | | 003156 | 6086 | 6091 | 6096# | | |
| TST11 | | 003246 | 6107 | 6112 | 6117# | | |
| TST12 | | 003372 | 6132 | 6140 | 6144# | | |
| TST13 | | 003462 | 6156 | 6161 | 6166# | | |
| TST14 | | 003564 | 6180 | 6185 | 6190# | | |
| TST15 | | 003654 | 6201 | 6206 | 6238# | | |
| TST16 | | 003746 | 6238 | 6239# | | | |
| TST17 | | 004040 | 6239 | 6240# | | | |
| TST2 | | 002570 | 5978# | | | | |
| TST20 | | 004132 | 6240 | 6241# | | | |
| TST21 | | 004224 | 6241 | 6242# | | | |
| TST22 | | 004316 | 6242 | 6243# | | | |
| TST23 | | 004410 | 6243 | 6244# | | | |
| TST24 | | 004502 | 6244 | 6245# | | | |
| TST25 | | 004574 | 6245 | 6247# | | | |
| TST26 | | 004626 | 6253 | 6261# | | | |
| TST27 | | 004752 | 6270 | 6280 | 6289 | 6291 | 6337# |
| TST3 | | 002654 | 6007# | | | | |
| TST30 | | 005040 | 6337 | 6339# | | | |
| TST31 | | 005130 | 6339 | 6341# | | | |
| TST32 | | 005216 | 6341 | 6343# | | | |
| TST33 | | 005304 | 6343 | 6345# | | | |

MAINDEC-11-CNIBA-A MACY11 30(1046) 15-DEC-82 16:09 PAGE 61-2^E 6
CNIBAA.P11 10-DEC-82 15:35 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0069

. ABS. 012766 000

ERRORS DETECTED: 0

CNIBAA,CNIBAA/CR/NL:TOC=CNMAC2.SML,CNIBAA.P11
RUN-TIME: 16 14 1 SECONDS
RUN-TIME RATIO: 71/32=2.1
CORE USED: 33K (66 PAGES)