

DRV11-J

DIAG TST PRT1  
CNDRCAO

AH-T448A-MC  
FICHE 1 OF 1

MAY 1983  
COPYRIGHT © 82-83  
MADE IN USA



Table with multiple columns and rows of data, likely diagnostic test results. The text is very faint and difficult to read, but appears to be organized in a grid format. The table contains various alphanumeric characters and symbols, possibly representing test parameters and their corresponding values or status.



IDENTIFICATION

PRODUCT NAME: CNDRCAO DRV11J DIAG TST PRT1  
PRODUCT CODE: AC-T447A-MC  
PRODUCT DATE: DECEMBER, 1982  
MAINTAINER: DIAGNOSTICS SERVICES/ISS  
AUTHOR: W.HEAVEY

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.

```
5685          ::GPA  .HEADER ^/CNDRCA  DRV11J DIAG TST PRT1/,1982,^/BILL HEAVEY/
5686          .TITLE CNDRCA  DRV11J DIAG TST PRT1
(1)          .*COPYRIGHT (C) 1982
(1)          .*DIGITAL EQUIPMENT CORP.
(1)          .*MAYNARD, MASS. 01754
(1)          .*
(1)          .*PROGRAM BY BILL HEAVEY
(1)          .*
(1)          .*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
(1)          .*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
(1)          .*
(1)          $TN=1
(1)          000001 $SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP
(1)          160000 .* EDITED TO PERMIT DRV'S TO RUN ON FALCON (KXT11).      ;;GPA
5687          .*                                     G. PASQUANTONIO, JULY '81      ;;GPA
5688          .*
5689          .*
5690          165400 $SWR=165400
5691          000001 $TN=1
5692          .SBTTL OPERATIONAL SWITCH SETTINGS
(1)          .*
(1)          .SWITCH              USE
(1)          ----              -----
(1)          15              HALT ON ERROR
(1)          14              LOOP ON TEST
(1)          13              INHIBIT ERROR TYPEOUTS
(1)          11              INHIBIT ITERATIONS
(1)          9              LOOP ON ERROR
(1)          8              LOOP ON TEST IN SWR<7:0>
5693          .SBTTL BASIC DEFINITIONS
(1)          .*
(1)          .;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)          001100 STACK= 1100
(1)          .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
(1)          .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
(1)          .*
(1)          .;*MISCELLANEOUS DEFINITIONS
(1)          000011 HT= 11          ;;CODE FOR HORIZONTAL TAB
(1)          000012 LF= 12          ;;CODE FOR LINE FEED
(1)          000015 CR= 15          ;;CODE FOR CARRIAGE RETURN
(1)          000200 CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
(1)          177776 PS= 177776     ;;PROCESSOR STATUS WORD
(1)          .EQUIV PS,PSW
(1)          177774 STKLMT= 177774  ;;STACK LIMIT REGISTER
(1)          177772 PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)          177570 DSWR= 177570   ;;HARDWARE SWITCH REGISTER
(1)          177570 DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
(1)          .***** THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
(1)          170000 ODTST= 170000
(1)          .;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)          000000 R0= %0          ;;GENERAL REGISTER
(1)          000001 R1= %1          ;;GENERAL REGISTER
(1)          000002 R2= %2          ;;GENERAL REGISTER
(1)          000003 R3= %3          ;;GENERAL REGISTER
(1)          000004 R4= %4          ;;GENERAL REGISTER
(1)          000005 R5= %5          ;;GENERAL REGISTER
(1)          000006 R6= %6          ;;GENERAL REGISTER
```

```
(1) 000007 R7= X7 ;;GENERAL REGISTER
(1) 000006 SP= X6 ;;STACK POINTER
(1) 000007 PC= X7 ;;PROGRAM COUNTER
(1)
(1) ;*PRIORITY LEVEL DEFINITIONS
(1) 000000 PR0= 0 ;;PRIORITY LEVEL 0
(1) 000040 PR1= 40 ;;PRIORITY LEVEL 1
(1) 000100 PR2= 100 ;;PRIORITY LEVEL 2
(1) 000140 PR3= 140 ;;PRIORITY LEVEL 3
(1) 000200 PR4= 200 ;;PRIORITY LEVEL 4
(1) 000240 PR5= 240 ;;PRIORITY LEVEL 5
(1) 000300 PR6= 300 ;;PRIORITY LEVEL 6
(1) 000340 PR7= 340 ;;PRIORITY LEVEL 7
(1)
(1) ;*'SWITCH REGISTER' SWITCH DEFINITIONS
(1) 100000 SW15= 100000
(1) 040000 SW14= 40000
(1) 020000 SW13= 20000
(1) 010000 SW12= 10000
(1) 004000 SW11= 4000
(1) 002000 SW10= 2000
(1) 001000 SW09= 1000
(1) 000400 SW08= 400
(1) 000200 SW07= 200
(1) 000100 SW06= 100
(1) 000040 SW05= 40
(1) 000020 SW04= 20
(1) 000010 SW03= 10
(1) 000004 SW02= 4
(1) 000002 SW01= 2
(1) 000001 SW00= 1
(1) .EQUIV SW09,SW9
(1) .EQUIV SW08,SW8
(1) .EQUIV SW07,SW7
(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)
(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)
(1)
(1)
(1) 000004 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1) 000010 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(1) 000014 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
5694 174160 ABASE= 174160 ;:BASE ADDRESS
5695 000001 ADEVM= 1 ;:DEFAULT TO ONE DRV11J
5696 100000 RDY= BIT15
5697 000400 DIR= BIT8
5698 001000 IE= BIT9
5699
5700 ;CHIP COMMAND SUMMARY
5701 000020 CIRM= 20 ;:CLEAR IRR AND IMR
5702 000030 CSIRM= 30 ;:CLEAR SINGLE IRR AND IMR BIT
5703
5704 000040 CIMR= 40 ;:CLEAR IMR
5705 000050 CSIMR= 50 ;:CLEAR SINGLE IMR BIT
5706 000060 SIMR= 60 ;:SET ALL IMR BITS
5707 000070 SSIMR= 70 ;:SET SINGLE IMR BITS
5708
5709 000100 CIRR= 100 ;:CLEAR IRR
5710 000110 CSIRR= 110 ;:CLEAR SINGLE IRR BITS
5711 000120 SIRR= 120 ;:SET ALL IRR BITS
5712 000130 SSIRR= 130 ;:SET SINGLE IRR BITS
5713
5714 000140 CHPISR= 140 ;:CLEAR HIGHEST PRIORITY ISR BIT
5715 000160 CISR= 160 ;:CLEAR ISR
5716 000170 CSISR= 170 ;:CLEAR SINGLE ISR BIT
5717
5718 000200 LMD04= 200 ;:LOAD MODE BITS M0-M4
5719 000240 LMD57= 240 ;:LOAD MODE BITS M5-M7
```

```
5720
5721      :CHIP MODE BIT PRESELECTION
5722      000240      MISR= 240
5723      000244      MIMR= 244
5724      000250      MIRR= 250
5725      000254      MACR= 254
5726
5727      :CHIP WRITE PRESELECTION
5728      000300      PACR= 300      ;PRESELECT AUTO CLEAR REG. FOR WRITING
5729      000260      PIMR= 260      ;PRESELECT IMR REG. FOR WRITING
5730      000340      PVMA= 340      ;PRESELECT VECTOR MEMORY ADDRESS
5731
5732      .SBTTL TRAP CATCHER
(1)
(1)      000000      .=0
(1)      :*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
(1)      :*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)      :*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)      000174      .=174
(1) 000174 000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
(1) 000176 000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
(1)
(1) 000200 000137 001402      .SBTTL STARTING ADDRESS(ES)
5733      000100      JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
5734 000100 000104 000200 000002      .=100
5735      .WORD 104,200,2      ;IF 'B EVENT' ON Q BUS IS CONNECTED
;IGNORE IT'S INTERRUPT - JUST DO A RTI
```

5737

(1)

(2)

(1)

(1)

(1)

(1)

(1)

(1)

(1)

5738

5739

5740

5741

5742

(1)

(2)

(1)

(2)

(1)

(1)

(1)

(1)

(1)

(1)

(2)

(1)

(1)

(1)

(1)

(1)

(1)

(1)

(1)

(1)

000046

000052

001000

001000

001002

001004

001006

001010

001012

000106

000046

013022

000052

000000

000106

001000

001000

000024

000200

000044

001000

001000

000000

001170

000006

000024

000024

000031

.SBTTL ACT11 HOOKS

\*\*\*\*\*  
:HOOKS REQUIRED BY ACT11

.\$VPC=.

.=46

.\$ENDAD

.=52

.WORD 0

.=.\$VPC

.=1000

;SAVE PC

::1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .\$EOP

::2)SET LOC.52 TO ZERO

:: RESTORE PC

;LONGEST TEST TIME

;1ST PASS RUN TIME

;ADDITIONAL RUN TIME

.SBTTL APT PARAMETER BLOCK

\*\*\*\*\*  
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

.\$X=.

.=24

200

.=44

.\$APTHDR

.=.\$X

::SAVE CURRENT LOCATION  
::SET POWER FAIL TO POINT TO START OF PROGRAM  
::FOR APT START UP  
::POINT TO APT INDIRECT ADDRESS PNTR.  
::POINT TO APT HEADER BLOCK  
::RESET LOCATION COUNTER

\*\*\*\*\*  
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
:INTERFACE SPEC.

.\$APTHD:

.\$HIBTS: .WORD

.\$MBADR: .WORD

.\$STSM: .WORD

.\$PASTM: .WORD

.\$UNITM: .WORD

.WORD

0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
.\$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)  
6. ;;RUN TIME OF LONGEST TEST  
20. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
20. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
.\$ETEND-.\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)





```
(2) 001206 000000 $MSGLG: .WORD AMSGLG ::MESSAGE LENGTH
(2) 001210 $ETABLE: ::APT ENVIRONMENT TABLE
(2) 001210 000 $ENV: .BYTE AENV ::ENVIRONMENT BYTE
(2) 001211 000 $ENVM: .BYTE AENVM
(2) ::ENVIRONMENT MODE BITS
(2) 001212 000000 $SWREG: .WORD ASWREG ::APT SWITCH REGISTER
(2) 001214 000000 $USWR: .WORD AUSWR ::USER SWITCHES
(2) 001216 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,PDQ=07,Q=10
(2) * BIT 10=REAL TIME CLOCK
(2) * BIT 9=FLOATING POINT PROCESSOR
(2) * BIT 8=MEMORY MANAGEMENT
(2) 001220 000 $MAMS1: .BYTE AMAMS1 ::HIGH ADDRESS,M.S. BYTE
(2) 001221 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) 001222 000000 $MADR1: .WORD AMADR1 ::HIGH ADDRESS,BLK#1
(2) * MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2) 001224 000 $MAMS2: .BYTE AMAMS2 ::HIGH ADDRESS,M.S. BYTE
(2) 001225 000 $MTYP2: .BYTE AMTYP2 ::MEM. TYPE,BLK#2
(2) 001226 000000 $MADR2: .WORD AMADR2 ::MEM.LAST ADDRESS,BLK#2
(2) 001230 000 $MAMS3: .BYTE AMAMS3 ::HIGH ADDRESS,M.S.BYTE
(2) 001231 000 $MTYP3: .BYTE AMTYP3 ::MEM. TYPE,BLK#3
(2) 001232 000000 $MADR3: .WORD AMADR3 ::MEM.LAST ADDRESS,BLK#3
(2) 001234 000 $MAMS4: .BYTE AMAMS4 ::HIGH ADDRESS,M.S.BYTE
(2) 001235 000 $MTYP4: .BYTE AMTYP4 ::MEM. TYPE,BLK#4
(2) 001236 000000 $MADR4: .WORD AMADR4 ::MEM.LAST ADDRESS,BLK#4
(2) 001240 000000 $VECT1: .WORD AVECT1 ::INTERRUPT VECTOR#1,BUS PRIORITY#1
(2) 001242 000000 $VECT2: .WORD AVECT2 ::INTERRUPT VECTOR#2BUS PRIORITY#2
(2) 001244 174160 $BASE: .WORD ABASE
(2) *::BASE ADDRESS OF EQUIPMENT UNDER TEST
(2) 001246 000001 $DEVN: .WORD ADEVN ::DEVICE MAP
(2) 001250 000000 $CDW1: .WORD ACDW1 ::CONTROLLER DESCRIPTION WORD#1
(2) 001252 $ETEND:
(2) .MEXIT
```

(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)

001252

5744

5745

5746

5747

5748

5749

5750

5751

5752

5753

5754

5755

5756

5757

5758

5759

5760

5761

5762

5763

5764

5765

5766

5767

5768

5769

5770

5771

5772

5773

5774

5775

5776

5777

5778

5779

5780

5781

5782

5783

5784

001252 016633

001254 016765

001256 017110

001260 000000

001262 016652

001264 016765

001266 017110

001270 000000

001272 016674

001274 016765

001276 017110

001300 000000

001302 016707

001304 016765

001306 017110

001310 000000

001312 016722

001314 016765

001316 017110

001320 000000

001322 016735

001324 016765

001326 017110

001330 000000

001332 016750

001334 016765

001336 017110

.SBTTL ERROR POINTER TABLE

;\*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.  
;\*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN  
;\*LOCATION \$I,EMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
;\*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).  
;\*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

:\* EM ::POINTS TO THE ERROR MESSAGE  
:\* DH ::POINTS TO THE DATA HEADER  
:\* DT ::POINTS TO THE DATA  
:\* DF ::POINTS TO THE DATA FORMAT

\$ERRTB:

:ERROR 1  
EM1 :REG TIMEOUT ER  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 2  
EM2 :REG READ/WRITE ER  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 3  
EM3 :IRR REG ER  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 4  
EM4 :ACR REG ER  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 5  
EM5 :IMR REG ERROR  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 6  
EM6 :ISR REG ERROR  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)  
0

:ERROR 7  
EM7 :CHIP STAT ER  
DH1 :ERRPC TSTNUM BUSADR EXPCT RCVD  
DT1 :\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT(R5)

5785 001340 000000

0

5786

5787

5788

5789

5790

5791 001342 174160

5792 001344 174162

5793 001346 174164

5794 001350 174166

5795 001352 174170

5796 001354 174172

5797 001356 174174

5798 001360 174176

5799

5800

5801

5802

5803 001362 000000

5804 001364 000001

5805 001366 000000

5806 001370 000000

5807 001372 000000

5808 001374 000000

5809 001376 000000

5810 001400 000000

; BUS REGISTER ADDRESS POINTERS

DRCSA: ABASE  
DRDBA: ABASE+2  
DRCSB: ABASE+4  
DRDBB: ABASE+6  
DR CSC: ABASE+10  
DRDBC: ABASE+12  
DR CSD: ABASE+14  
DRDBD: ABASE+16

; COMMON PROGRAM LOCATION(S)

TSTNUM: 0 ;CONTAINS TEST NUMBER ON ERROR  
DMAP: 1  
INTFLG: .WORD 0  
XXDP: .WORD 0  
IMRLOC: .WORD 0  
ISRLOC: .WORD 0  
IRRLOC: .WORD 0  
ACRLOC: .WORD 0

```

5813          .SBTTL PROGRAM START
5814 001402 START:
(1)          .SBTTL INITIALIZE THE COMMON TAGS
(1)          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1) 001402 012706 0011C0 MOV    #CMTAG,R6    ;;FIRST LOCATION TO BE CLEARED
(1) 001406 005026 CLR    (R6)+        ;;CLEAR MEMORY LOCATION
(1) 001410 022706 001140 CMP    #SWR,R6    ;;DONE?
(1) 001414 001374 BNE    -6          ;;LOOP BACK IF NO
(1) 001416 012706 001100 MOV    #1100,SP    ;;SETUP THE STACK POINTER
(1)          ;;INITIALIZE A FEW VECTORS
(1) 001422 012737 015216 000020 MOV    $$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1) 001430 012737 000300 000022 MOV    #PR6,@#IOTVEC+2 ;;LEVEL 6
(1) 001436 012737 014670 000030 MOV    $ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1) 001444 012737 000300 000032 MOV    #PR6,@#EMTVEC+2 ;;LEVEL 6
(1)          ;;BIT02
(1) 001452 012737 016456 000034 MOV    $STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1) 001460 012737 000300 000036 MOV    #PR6,@#TRAPVEC-2 ;;LEVEL 6
(1) 001466 012737 016252 000024 MOV    $PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
(1) 001474 012737 000300 000026 MOV    #PR6,@#PWRVEC+2 ;;LEVEL 6
(1) 001502 005037 001160 CLR    $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
(1) 001506 005037 001162 CLR    $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1) 001512 112737 000001 001115 MOVB  #1,$ERMAX    ;;ALLOW ONE ERROR PER TEST
(1) 001520 012737 001520 001106 MOV    #.,$LPADR   ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1) 001526 012737 001526 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) 001534 013746 000004 MOV    @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
(2) 001540 012737 001574 000004 MOV    #64,$@#ERRVEC ;;SET UP ERROR VECTOR
(2) 001546 012737 177570 001140 MOV    #DSWR,SWR    ;;SETUP FOR A HARDWARE SWITCH REGISTER
(2) 001554 012737 177570 001142 MOV    #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2) 001562 022777 177777 177350 CMP    #-1,@SWR    ;;TRY TO REFERENCE HARDWARE SWR
(2) 001570 001012 BNE    66$        ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)          ;;AND THE HARDWARE SWR IS NOT = -1
(2) 001572 000403 BR    65$        ;;BRANCH IF NO TIMEOUT
(2) 001574 012716 001602 64$: MOV    #65$,(SP)    ;;SET UP FOR TRAP RETURN
(2) 001600 000002 RTI
(2) 001602 012737 000176 001140 65$: MOV    #SWREG,SWR    ;;POINT TO SOFTWARE SWR
(2) 001610 012737 000174 001142 MOV    #DISPREG,DISPLAY
(2) 001616 012637 000004 66$: MOV    (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
(1)
(2) 001622 005037 001176 CLR    $PASS      ;;CLEAR PASS COUNT
(2) 001626 132737 000200 001211 BITB  #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2) 001634 001403 BEQ    67$        ;;YES,USE NON-APT SWITCH
(2) 001636 012737 001212 001140 MOV    #$$SWREG,SWR ;;NO,USE APT SWITCH REGISTER
(2) 001644
5815 001644 005037 001172 CLR    $FATAL     ;;CLEAR ERROR NUMBER
5816 001650 005037 001170 CLR    $MSGTYP    ;;CLEAR MESSAGE TYPE
5817 001654 005037 001174 CLR    $TESTN     ;;CLEAR TEST NUMBER
5818          ;;THE FOLLOWING 9 LINES WERE DELETED
5819          ;;IN ORDER TO MAKE THIS DIAGNOSTIC SPECIFIC TO 11/21 PROCESSOR.
5820          :CALL  FALCON      ;;CHECK FOR FALCON (KXT11)          ;;GPA
5821          :BEQ   1000$     ;;BR IF NOT KXT11 SYSTEM          ;;GPA
5822          :BIC  #40,@#22   ;;YES, STRAP IOT...          ;;GPA
5823          :BIC  #40,@#32   ;;...EMT...          ;;GPA
5824          :BIC  #40,@#36   ;;...AND TRAP TO LEVEL 6.  ;;GPA
5825          :CMP   $BASE,#ABASE ;;IS $BASE VIRGIN ??          ;;GPA

```

```

5826                                     ;BNE 1000$           ; BR IF NOT.           ;;GPA
5827                                     ;MOV #174160,$BASE ; YES, USE ENGINEERING DEFAULT ;;GPA
5828                                     ;:1000$:           ;;GPA
5829                                     ;CHECK OPERATING ENVIRONMENT
5830 001660 005737 000042                TST @#42           ;ARE WE IN ACT/XXDP AUTO MODE?
5831 001664 001410                       BEQ 1$            ;BRANCH IF NO
5832 001666 023737 000042 000046       CMP @#42,@#46     ;IS IT ACT AUTO MODE?
5833 001674 001410                       BEQ 2$            ;BRANCH IF YES
5834 001676 012737 177777 001370       MOV #-1,XXDP     ;SET XXDP CHAIN MODE INDICATOR
5835 001704 000404                       BR 2$
5836 001706 123727 001210 000001 1$:   CMPB $ENV,#1     ;ARE WE IN APT AUTO MODE?
5837 001714 001003                       BNE 3$            ;BRANCH IF NO
5838 001716 112737 000001 001134 2$:   MOVB #1,$AUTOB   ;SET AUTO MODE INDICATOR
5839
5840                                     ;PRINT TITLE IF NOT IN ACT OR APT AUTO MODE
5841 001724 005227 177777                3$:   INC #-1         ;FIRST TIME?
5842 001730 001012                       BNE 5$            ;SKIP TITLE IF NO
5843 001732 005737 001134                TST $AUTOB       ;ARE WE IN AUTO MODE?
5844 001736 001403                       BEQ 4$            ;BRANCH TO TITLE TYPEOUT IF NOT
5845 001740 005737 001370                TST XXDP         ;IS THE AUTO MODE UNDER XXDP?
5846 001744 001404                       BEQ 5$            ;SKIP TITLE IF NOT
5847 001746 104401 016536                4$:   TYPE ,TITLED ;PRINT OUT THE TITLE
5848 001752 104401 016604                TYPE ,TLCABL     ;PRINT 'DRV11J CABLE REQ'D'
5849
5850                                     ;GET THE VALUE IN THE SOFTWARE SWITCH REGISTER
5851 001756 005737 001134                5$:   TST $AUTOB   ;ARE WE IN AUTOMATIC MODE?
5852 001762 001001                       BNE START1       ;BRANCH IF YES
5853 001764 104406                       GTSWR            ;ASK FOR SWR INPUT FROM CONSOLE
5854 001766 005037 001202                START1: CLR $UNIT  ;CLEAR UNIT NUMBER
5855 001772 013737 001246 001364       MOV $DEV#,$DMAP ;POSITION OF DRV11-J'S
5856 002000 042737 177760 001364       BIC #177760,$DMAP ;UP TO 4 DRV11-J'S ONLY
5857 002006 013701 001244                MOV $BASE,R1    ;GET BASE ADDRESS
5858 002012 010137 001342                MOV R1,$DRCSA  ;MAKE BUS REG. POINTER = $BASE
5859 002016 032737 000001 001364       BIT #1,$DMAP    ;IS FIRST DRV11-J SELECTED?
5860 002024 001002                       BNE YES         ;YES
5861 002026 000137 012700                JMP NXDEV1      ;ADVANCE BASE DRV11-J ADDRESS
5862 002032 012700 001342                NEXPAS: MOV #DRCSA,R0 ;SET UP REGISTER ADDRESS POINTERS
5863 002036 010120                NEXPA1: MOV R1,(R0)+ ;LOAD EM,R1 = DRV11-J CSRA ADDRESS
5864 002040 062701 000002                ADD #2,R1       ;DO POINTERS FOR ALL REGS, A THRU D
5865 002044 022700 001362                CMP #DRDBD+2,R0 ;ALL DONE?
5866 002050 001372                       BNE NEXPA1      ;BR IF NOT
5867 002052 012706 001100                MOV #STACK,SP  ;ALWAYS RESET STACK
5868 002056 012705 001126                MOV #SBDDAT,R5 ;INIT R5 WITH SBDDAT
5869 002062 013737 001202 001200       MOV $UNIT,$DEVCT ;LOAD APT COUNTER WITH UNIT NO.
5870 002070 106427 000300                MTPS #PR6      ;SET PRIORITY TO 6
5871
5872
5873
5874                                     ;*****
(3)                                     ;*TEST 1 TEST THAT ALL REGISTERS ARE ADDRESSABLE
(3)                                     ;*****
(2) 002074 000004                TST1: SCOPE
(2)
5875 002076 005037 001124                CLR $GDDAT      ;NO DATA COMPARE
5876 002102 005015                       CLR (R5)        ;NO DATA COMPARE
5877 002104 012737 002140 000004       MOV #2$,@#ERRVEC ;SET UP TIMEOUT RETURN ADDR

```

```

5878 002112 013700 001342      MOV      DRCSA,R0      ;SET UP 1ST DRV11 BUS ADRS
5879 002116 012701 000010      MOV      #8.,R1      ;SET UP REG COUNT
5880 002122 010037 001122      1$:     MOV      R0,$BDADR ;SET UP CURRENT DRV BUS ADRS
5881 002126 005010                CLR      (R0)        ;SEE IF THERE
5882 002130 005720                TST      (R0)+       ;BUMP TO NEXT
5883 002132 005301                DEC      R1          ;COUNT 8 OF THEM
5884 002134 001403                BEQ      3$         ;BR IF ALL DONE
5885 002136 000771                BR       1$         ;TRY NEXT
5886 002140 022626      2$:     CMP      (SP)+,(SP)+ ;FIX STACK SINCE NO RTI
5887 002142 104001                ERROR   1          ;BUS ADRS INDICATED DID NOT RESPOND
5888 002144 012737 000006 000004 3$:     MOV      #ERRVEC+2,@#ERRVEC ;RESTORE LOC 4
5889
5890

```

```

:*****
:*TEST 2      TEST CSRA W/R DIR BIT,INT. CHIP RESET STATUS
:*****
TST2:  SCOPE

```

```

5891 002154 013700 001342      MOV      DRCSA,R0      ;GET CSR ADDRESS
5892 002160 013701 001352      MOV      DRCSA,R1      ;STORE CSRC ADDRESS
5893 002164 005010                CLR      (R0)        ;INIT CSRA
5894 002166 005011                CLR      (R1)        ;INIT CSRC
5895 002170 010037 001122      MOV      R0,$BDADR      ;STORE CSR ADDRESS
5896                                ;SET UP EXPECTED DATA
5897 002174 012737 100700 001124  MOV      #RDY!DIR!BIT7!BIT6,$GDDAT
5898 002202 112760 000001 000001  MOVVB   #BIT0,1(R0)    ;SET DIRECTION BIT
5899 002210 011015                MOV      (R0),(R5)    ;GET CSR DATA
5900 002212 042715 000007                BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
5901 002216 023715 001124                CMP      $GDDAT,(R5)
5902 002222 001401                BEQ      100$        ;CSRA ERROR
5903 002224 104002                ERROR   2          ;STORE CSRC ADDRESS
5904 002226 010137 001122      100$:   MOV      R1,$BDADR      ;STORE EXPECTED
5905 002232 012737 000200 001124  MOV      #BIT7,$GDDAT ;STORE CSRC
5906 002240 011115                MOV      (R1),(R5)    ;CLEAR UNDEFINED BITS
5907 002242 042715 000007                BIC      #7,(R5)
5908 002246 023715 001124                CMP      $GDDAT,(R5)
5909 002252 001401                BEQ      1$         ;CSRC ERROR
5910 002254 104002                ERROR   2          ;CSRA ADDRESS
5911 002256 010037 001122      1$:     MOV      R0,$BDADR      ;CLEAR CSRA DIR BIT
5912 002262 012737 100300 001124  MOV      #RDY!BIT7!BIT6,$GDDAT
5913 002270 105060 000001                CLR      1(R0)
5914 002274 011015                MOV      (R0),(R5)    ;READ CSR
5915 002276 042715 000007                BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
5916 002302 023715 001124                CMP      $GDDAT,(R5)
5917 002306 001401                BEQ      2$         ;:BR IF EQUAL
5918 002310 104002                ERROR   2
5919 002312 012737 000300 001124 2$:     MOV      #BIT7!BIT6,$GDDAT
5920 002320 112761 000001 000001  MOVVB   #BIT0,1(R1)    ;CSRC TO OUTPUT MODE
5921 002326 011015                MOV      (R0),(R5)    ;READ CSRA
5922 002330 042715 000007                BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
5923 002334 023715 001124                CMP      $GDDAT,(R5) ;CHECK IF RDY BIT CLEARED
5924 002340 001401                BEQ      TST3        ;:BR IF EQUAL
5925 002342 104002                ERROR   2          ;CSRA REG ERROR
5926
5927

```

```

:*****
:*TEST 3      TEST CSRA INT. ENABLE BIT
:*****

```

```

(2) 002344 000004 TST3: SCOPE
(2)
5928 002346 004737 013056 JSR PC,CLRCR :CLEAR CSR REGISTER
5929 002352 013700 001342 MOV DRCSA,R0 :GET CSRA ADDRESS
5930 002356 013701 001352 MOV DRCSA,R1 :GET CSRC ADDRESS
5931 002362 112761 000001 000001 MOVB #BIT0,1(R1) :SET DIRECTION BIT CSRC
5932 002370 010037 001122 MOV R0,$BDADR :STORE CSRA ADDRESS
5933 002374 012737 001300 001124 MOV #BIT9!BIT7!BIT6,$GDDAT
5934 002402 012710 001000 MOV #IE,(R0) :SET INTERRUPT ENABLE BIT
5935 002406 011015 MOV (R0),(R5)
5936 002410 042715 000007 BIC #7,(R5) :CLEAR UNDEFINED BITS
5937 002414 023715 001124 CMP $GDDAT,(R5)
5938 002420 001401 BEQ 1$
5939 002422 104002 ERROR 2 :INT. ENABLE ERROR,CSRA
5940 002424 012737 000300 001124 1$: MOV #BIT7!BIT6,$GDDAT
5941 002432 105060 000001 CLRB 1(R0) :CLEAR I/E BIT
5942 002436 011015 MOV (R0),(R5)
5943 002440 042715 000007 BIC #7,(R5) :CLEAR UNDEFINED BITS
5944 002444 023715 001124 CMP $GDDAT,(R5)
5945 002450 001401 BEQ TST4 :;BR IF EQUAL
5946 002452 104002 ERROR 2 :CSRA ERROR
5947
5948

```

```

:*****
:*TEST 4 TEST CSRA I/E,DIR BIT
:*****

```

```

(2) 002454 000004 TST4: SCOPE
(2)
5949 002456 004737 013056 JSR PC,CLRCR :CLEAR CSR REGISTERS
5950 002462 013700 001342 MOV DRCSA,R0 :GET CSRA ADDRESS
5951 002466 010037 001122 MOV R0,$BDADR :SAVE CSRA ADDRESS
5952 002472 012737 101700 001124 MOV #101700,$GDDAT :EXPECTED I/E,DIR
5953 002500 112760 000003 000001 MOVB #BIT1!BIT0,1(R0) :SET I/E AND DIR BIT
5954
5955 002506 011015 MOV (R0),(R5)
5956 002510 042715 000007 BIC #7,(R5) :CLEAR UNDEFINED BITS
5957 002514 023715 001124 CMP $GDDAT,(R5)
5958 002520 001401 BEQ 1$
5959 002522 104002 ERROR 2 :CSRA ERROR
5960 002524 012737 100300 001124 1$: MOV #RDY!BIT7!BIT6,$GDDAT
5961 002532 005010 CLR (R0)
5962 002534 011015 MOV (R0),(R5)
5963 002536 042715 000007 BIC #7,(R5) :CLEAR UNDEFINED BITS
5964 002542 023715 001124 CMP $GDDAT,(R5)
5965 002546 001401 BEQ TST5 :;BR IF EQUAL
5966 002550 104002 ERROR 2 :CSRA ERROR
5967
5968

```

```

:*****
:*TEST 5 TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS
:*****

```

```

(2) 002552 000004 TST5: SCOPE
(2)
5970 002554 004737 013056 JSR PC,CLRCR :CLEAR CSR REGISTERS
5971 002560 013700 001346 MOV DRCSB,R0 :GET CSR ADDRESS
5972 002564 013701 001356 MOV DRCSB,R1 :GET CSR ADDRESS
5973 002570 010037 001122 MOV R0,$BDADR :STORE CSR ADDRESS

```

CNDRCA DRV11J DIAG TST PRT1 MACY11 30(1046) 15-DEC-82 15:25 PAGE 57-13  
CNDRCA.P11 10-DEC-82 14:37 T5 TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS

SEQ 0015

```

5974 002574 012737 100400 001124      MOV      #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
5975 002602 112760 000001 000001      MOVVB   #BIT0,1(R0)    ;SET DIRECTION BIT
5976 002610 011015                MOV      (R0),(R5)      ;GET CSRB DATA
5977 002612 023715 001124      CMP      $GDDAT,(R5)
5978 002616 001401                BEQ      100$
5979 002620 104002                ERROR    2              ;CSRB ERROR
5980 002622 010137 001122      100$:  MOV      R1,$BDADR     ;STORE CSRD ADDRESS
5981 002626 005037 001124      CLR      $GDDAT        ;STORE EXPECTED
5982 002632 011115                MOV      (R1),(R5)     ;READ CSRD
5983 002634 023715 001124      CMP      $GDDAT,(R5)
5984 002640 001401                BEQ      1$
5985 002642 104002                ERROR    2              ;CSRD ERROR
5986 002644 010037 001122      1$:    MOV      R0,$BDADR     ;STORE CSRB ADDRESS
5987 002650 012737 100000 001124      MOV      #RDY,$GDDAT   ;STORE EXPECTED
5988 002656 105060 000001                CLRB    1(R0)          ;CLEAR CSR DIR BIT
5989 002662 011015                MOV      (R0),(R5)     ;READ CSR
5990 002664 023715 001124      CMP      $GDDAT,(R5)
5991 002670 001401                BEQ      2$            ;;BR IF EQUAL
5992 002672 104002                ERROR    2
5993 002674 005037 001124      2$:    CLR      $GDDAT        ;STORE EXPECTED
5994 002700 112761 000001 000001      MOVVB   #BIT0,1(R1)    ;CSRD TO OUTPUT MODE
5995 002706 011015                MOV      (R0),(R5)     ;READ CSRB
5996 002710 023715 001124      CMP      $GDDAT,(R5)   ;RDY BIT CLEARED
5997 002714 001401                BEQ      TST6          ;;BR IF EQUAL
5998 002716 104002                ERROR    2              ;CSRB REG ERROR
5999

```

```

6000
(3)
(3)
(2) 002720 000004
(2)
::*****
:*TEST 6      TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS
::*****
TST6:  SCOPE

```

```

6001 002722 004737 013056                JSR      PC,CLRCR      ;CLEAR CSR REGISTERS
6002 002726 013700 001352                MOV      DRCSC,R0     ;GET CSR ADDRESS
6003 002732 013701 001342                MOV      DRCSA,R1     ;GET CSRA ADDRESS
6004 002736 010037 001122                MOV      R0,$BDADR     ;STORE CSR ADDRESS
6005 002742 012737 100600 001124      MOV      #RDY!DIR!BIT7,$GDDAT
6006 002750 112760 000001 000001      MOVVB   #BIT0,1(R0)    ;SET DIRECTION BIT
6007 002756 011015                MOV      (R0),(R5)     ;GET CSR DATA
6008 002760 042715 000007                BIC     #7,(R5)        ;CLEAR UNDEFINED BITS
6009 002764 023715 001124      CMP      $GDDAT,(R5)
6010 002770 001401                BEQ      100$
6011 002772 104002                ERROR    2
6012 002774 010137 001122      100$:  MOV      R1,$BDADR     ;STORE CSR ADDRESS
6013 003000 012737 000300 001124      MOV      #BIT7!BIT6,$GDDAT
6014 003006 011115                MOV      (R1),(R5)     ;READ CSRA
6015 003010 042715 000007                BIC     #7,(R5)        ;CLEAR UNDEFINED BITS
6016 003014 023715 001124      CMP      $GDDAT,(R5)
6017 003020 001401                BEQ      1$           ;CHECK CSRA
6018 003022 104002                ERROR    2              ;CSRA ERROR
6019 003024 010037 001122      1$:    MOV      R0,$BDADR     ;STORE CSR ADDRESS
6020 003030 012737 100200 001124      MOV      #RDY!BIT7,$GDDAT ;STORE EXPECTED DATA
6021 003036 105060 000001                CLRB    1(R0)          ;CLEAR CSR DIR BIT
6022 003042 011015                MOV      (R0),(R5)     ;READ CSR
6023 003044 042715 000007                BIC     #7,(R5)        ;CLEAR UNDEFINED BITS
6024 003050 023715 001124      CMP      $GDDAT,(R5)
6025 003054 001401                BEQ      2$            ;;BR IF EQUAL

```



CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046)  
T6

15-DEC-82 15:25 PAGE 57-14  
TEST CSRC W/R DIR BIT,INT CHIP RESET STATUS

SEQ 0016

```

6026 003056 104002
6027 003060 012737 000200 001124 2$: ERROR 2
6028 003066 112761 000001 000001 MOV #BIT7,$GDDAT ;EXPECTED DATA
6029 003074 011015 MOVB #BIT0,1(R1) ;CSRA TO OUTPUT MODE
6030 003076 042715 000007 MOV (R0),(R5) ;READ CSRC
6031 003102 023715 001124 BIC #7,(R5) ;CLEAR UNDEFINED BITS
6032 003106 001401 CMP $GDDAT,(R5) ;RDY BIT CLEARED
6033 003110 104002 BEQ TST7 ;:BR IF EQUAL
6034 ERROR 2 ;CSRC REG ERROR

```

```

6035
(3)
(3)
(2) 003112 000004
(2)
TST7: SCOPE

```

```

6036 003114 004737 013056 JSR PC,CLRCR ;CLEAR CSR REGISTERS
6037 003120 013700 001356 MOV DRCSD,R0 ;GET CSR ADDRESS
6038 003124 013701 001346 MOV DRCR,R1 ;CSR ADDRESS
6039 003130 010037 001122 MOV R0,$BDADR ;STORE CSR ADDRESS
6040 003134 012737 100400 001124 MOV #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
6041 003142 112760 000001 000001 MOVB #BIT0,1(R0) ;SET DIRECTION BIT
6042 003150 011015 MOV (R0),(R5) ;GET CSR DATA
6043 003152 023715 001124 CMP $GDDAT,(R5)
6044 003156 001401 BEQ 100$
6045 003160 104002 ERROR 2
6046 003162 010137 001122 100$: MOV R1,$BDADR ;STORE CSR ADDRESS
6047 003166 005037 001124 CLR $GDDAT ;STORE EXPECTED
6048 003172 011115 MOV (R1),(R5)
6049 003174 023715 001124 CMP $GDDAT,(R5)
6050 003200 001401 BEQ 1$
6051 003202 104002 ERROR 2 ;CSR ERROR
6052 003204 010037 001122 1$: MOV R0,$BDADR ;STORE CSR ADDRESS
6053 003210 012737 100000 001124 MOV #RDY,$GDDAT ;STORE EXPECTED
6054 003216 105060 000001 CLR 1(R0) ;CLEAR CSR DIR BIT
6055 003222 011015 MOV (R0),(R5) ;READ CSR
6056 003224 023715 001124 CMP $GDDAT,(R5)
6057 003230 001401 BEQ 2$ ;:BR IF EQUAL
6058 003232 104002 ERROR 2
6059 003234 005037 001124 2$: CLR $GDDAT ;EXPECTED
6060 003240 112761 000001 000001 MOVB #BIT0,1(R1) ;CSR TO OUTPUT MODE
6061 003246 011015 MOV (R0),(R5) ;READ CSR
6062 003250 023715 001124 CMP $GDDAT,(R5) ;RDY BIT CLEARED
6063 003254 001401 BEQ TST10 ;:BR IF EQUAL
6064 003256 104002 ERROR 2 ;CSR REG ERROR

```

```

6065
6066
(3)
(3)
(2) 003260 000004
(2)
TST10: SCOPE

```

```

6067 003262 004737 013056 JSR PC,CLRCR ;CLEAR ALL CSRS
6068 003266 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6069 003272 012737 003322 001110 MOV #1$,$LPERR ;SET UP SCOPE ADDRESS
6070 003300 013700 001342 MOV DRCR,R0 ;GET CSRA
6071 003304 013701 001344 MOV DRDBA,R1 ;GET DBRA ADDRESS
6072 003310 010137 001122 MOV R1,$BDADR ;STORE DBRA ADDRESS
6073 003314 112760 000001 000001 MOVB #BIT0,1(R0) ;SET CSRA IN OUTPUT MODE

```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 PAGE 57-15  
T10 TEST DBRA W/R IN OUTPUT MODE

SEQ 0017

```

6074 003322 011337 001124 1$: MOV (R3),%GDDAT ;SAVE EXPECTED DATA
6075 003326 005011 CLR (R1) ;CLEAR DBRA
6076 003330 051311 BIS (R3),(R1) ;WRITE INTO DBRA
6077 003332 011115 MOV (R1),(R5) ;READ DBRA
6078 003334 023715 001124 CMP %GDDAT,(R5) ;CHECK W/R DBRA
6079 003340 001401 BEQ 2$ ;NEXT PAT. IF EQUAL
6080 003342 104002 ERROR 2 ;DBRA W/R ERROR
6081 003344 005723 013464 2$: TST (R3)+ ;INC FOR NEXT PATTERN
6082 003346 020327 CMP R3,#ENDDAT ;CHECK FOR END
6083 003352 001363 BNE 1$ ;DO NEXT PATTERN
6084
6085

```

```

:*****
:*TEST 11 TEST DBRB W/R IN OUTPUT MODE
:*****
TST11: SCOPE

```

```

6086 003356 004737 013056 JSR PC,CLRCR ;CLEAR ALL CSRS
6087 003362 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6088 003366 012737 003416 001110 MOV #1$,%LPERR ;SET UP SCOPE ADDRESS
6089 003374 013700 001346 MOV DRCSB,R0 ;GET CSRB
6090 003400 013701 001350 MOV DRDBB,R1 ;GET DBRB ADDRESS
6091 003404 010137 001122 MOV R1,%BDADR ;STORE DBRB ADDRESS
6092 003410 112760 000001 000001 MOVB #BIT0,1(R0) ;SET CSRB IN OUTPUT MODE
6093 003416 011337 001124 1$: MOV (R3),%GDDAT ;SAVE EXPECTED DATA
6094 003422 005011 CLR (R1) ;CLEAR DBRB
6095 003424 051311 BIS (R3),(R1) ;WRITE INTO DBRB
6096 003426 011115 MOV (R1),(R5) ;READ DBRB
6097 003430 023715 001124 CMP %GDDAT,(R5) ;CHECK W/R DBRB
6098 003434 001401 BEQ 2$ ;NEXT PAT. IF EQUAL
6099 003436 104002 ERROR 2 ;DBRB W/R ERROR
6100 003440 005723 013464 2$: TST (R3)+ ;INC FOR NEXT PATTERN
6101 003442 020327 CMP R3,#ENDDAT ;CHECK FOR END
6102 003446 001363 BNE 1$ ;DO NEXT PATTERN
6103
6104
6105

```

```

:*****
:*TEST 12 TEST DBRC W/R IN OUTPUT MODE
:*****
TST12: SCOPE

```

```

6106 003452 004737 013056 JSR PC,CLRCR ;CLEAR ALL CSRS
6107 003456 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6108 003462 012737 003512 001110 MOV #1$,%LPERR ;SET UP SCOPE ADDRESS
6109 003470 013700 001352 MOV DRCSB,R0 ;GET CSRB
6110 003474 013701 001354 MOV DRDBC,R1 ;GET DBRC ADDRESS
6111 003500 010137 001122 MOV R1,%BDADR ;STORE DBRC ADDRESS
6112 003504 112760 000001 000001 MOVB #BIT0,1(R0) ;SET CSRB IN OUTPUT MODE
6113 003512 011337 001124 1$: MOV (R3),%GDDAT ;SAVE EXPECTED DATA
6114 003516 005011 CLR (R1) ;CLEAR DBRC
6115 003520 051311 BIS (R3),(R1) ;WRITE INTO DBRC
6116 003522 011115 MOV (R1),(R5) ;READ DBRC
6117 003524 023715 001124 CMP %GDDAT,(R5) ;CHECK W/R DBRC
6118 003530 001401 BEQ 2$ ;NEXT PAT. IF EQUAL
6119 003532 104002 ERROR 2 ;DBRC W/R ERROR
6120 003534 005723 013464 2$: TST (R3)+ ;INC FOR NEXT PATTERN
6121 003536 020327 CMP R3,#ENDDAT ;CHECK FOR END

```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 F 2  
T12 TEST DBRC W/R IN OUTPUT MODE PAGE 57-16

SEQ 0018

6122 003542 001363

BNE 1\$ :DO NEXT PATTERN

6123  
6124  
6125

\*\*\*\*\*  
: \*TEST 13 TEST DBRD W/R IN OUTPUT MODE  
: \*\*\*\*\*  
TST13: SCOPE

(3)  
(3)  
(2) 003544 000004  
(2)

6126 003546 004737 013056

JSR PC,CLRCR :CLEAR ALL CSRS

6127 003552 012703 013164

MOV #BEGPAT,R3 :GET DATA PATTERN TABLE

6128 003556 012737 003606 001110

MOV #1\$,\$LPERR :SET UP SCOPE ADDRESS

6129 003564 013700 001356

MOV DRCSD,R0 :GET CSR

6130 003570 013701 001360

MOV DRDBD,R1 :GET DBRD ADDRESS

6131 003574 010137 001122

MOV R1,\$BDADR :STORE DBRD ADDRESS

6132 003600 112760 000001 000001

MOVB #BIT0,1(R0) :SET CSR IN OUTPUT MODE

6133 003606 011337 001124

1\$: MOV (R3),\$GDDAT :SAVE EXPECTED DATA

6134 003612 005011

CLR (R1) :CLEAR DBRD

6135 003614 051311

BIS (R3),(R1) :WRITE INTO DBRD

6136 003616 011115

MOV (R1),(R5) :READ DBRD

6137 003620 023715 001124

CMP \$GDDAT,(R5) :CHECK W/R DBRD

6138 003624 001401

BEQ 2\$ :NEXT PAT. IF EQUAL

6139 003626 104002

ERROR 2 :DBRD W/R ERROR

6140 003630 005723

2\$: TST (R3)+ :INC FOR NEXT PATTERN

6141 003632 020327 013464

CMP R3,#ENDDAT :CHECK FOR END

6142 003636 001363

BNE 1\$ :DO NEXT PATTERN

```
6144
6145
6146      ::*****
(3)      :*TEST 14      TEST CSR UNIQUENESS,CSRS (A-B),(C-D)
(3)      :*****
(2)      003640  000004      TST14: SCOPE
(2)
6147      003642  004737  013056      JSR      PC,CLRCSR      ;CLEAR ALL CSRS
6148                                          ;CSRA = R0
6149                                          ;CSRB = R1
6150                                          ;CSRC = R2
6151                                          ;CSRD = R3
6152      003646  112760  000003  000001      MOVB     #3,1(R0)      ;CSRA OUTPUT MODE,I/E
6153      003654  112763  000001  000001      MOVB     #BIT0,1(R3)   ;CSRD OUTPUT MODE
6154      003662  010037  001122      MOV      R0,$BDADR
6155      003666  012737  101700  001124      MOV      #101700,$GDDAT ;EXPECTED DATA
6156      003674  011015      MOV      (R0),(R5)
6157      003676  042715  000007      BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
6158      003702  023715  001124      CMP      $GDDAT,(R5)
6159      003706  001401      BEQ      1$
6160      003710  104002      ERROR   2
6161      003712  010137  001122      1$:     MOV      R1,$BDADR      ;CSRA ERROR
6162      003716  005037  001124      CLR      $GDDAT      ;CHECK CSRB
6163      003722  011115      MOV      (R1),(R5)
6164      003724  023715  001124      CMP      $GDDAT,(R5)
6165      003730  001401      BEQ      2$
6166      003732  104002      ERROR   2
6167      003734  010337  001122      2$:     MOV      R3,$BDADR      ;CSRB ERROR
6168      003740  012737  100400  001124      MOV      #100400,$GDDAT
6169      003746  011315      MOV      (R3),(R5)
6170      003750  023715  001124      CMP      $GDDAT,(R5)
6171      003754  001401      BEQ      3$
6172      003756  104002      ERROR   2
6173      003760  010237  001122      3$:     MOV      R2,$BDADR      ;CSRD ERROR
6174      003764  012737  006200  001124      MOV      #BIT7,$GDDAT ;CHECK CSRC
6175      003772  011215      MOV      (R2),(R5)   ;EXPECTED
6176      003774  042715  000007      BIC      #7,(R5)
6177      004000  023715  001124      CMP      $GDDAT,(R5)
6178      004004  001401      BEQ      TST15      ;:BR IF EQUAL
6179      004006  104002      ERROR   2
6180                                          ;CSRC ERROR
6181      ::*****
(3)      :*TEST 15      TEST CSR UNIQUENESS,CSRS (A-D),(C-B)
(3)      :*****
(2)      004010  000004      TST15: SCOPE
(2)
6182      004012  004737  013056      JSR      PC,CLRCSR      ;CLR ALL CSRS
6183                                          ;CSRA = R0
6184                                          ;CSRB = R1
6185                                          ;CSRC = R2
6186                                          ;CSRD = R3
6187      004016  112760  000003  000001      MOVB     #3,1(R0)      ;CSRA OUTPUT,I/E
6188      004024  112761  000001  000001      MOVB     #BIT0,1(R1)   ;CSRB OUTPUT MODE
6189      004032  010037  001122      MOV      R0,$BDADR      ;CSRA ADDRESS
6190      004036  012737  101700  001124      MOV      #101700,$GDDAT
6191      004044  011015      MOV      (R0),(R5)
```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 PAGE 58-1  
T15 TEST CSR UNIQUENESS,CSRS (A-D),(C-B)

SEQ 0020

```

6192 004046 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS
6193 004052 023715 001124 CMP $GDDAT,(R5)
6194 004056 001400 BEQ 1$
6195 004060 010137 001122 1$: MOV R1,$BDADR ;CHECK CSRB
6196 004064 012737 100400 001124 MOV #100400,$GDDAT
6197 004072 011115 MOV (R1),(R5)
6198 004074 023715 001124 CMP $GDDAT,(R5)
6199 004100 001401 BEQ 2$
6200 004102 104002 ERROR 2 ;CSRB ERROR
6201 004104 010237 001122 2$: MOV R2,$BDADR ;CHECK CSRC
6202 004110 012737 000200 001124 MOV #BIT7,$GDDAT
6203 004116 011215 MOV (R2),(R5)
6204 004120 042715 000007 BIC #7,(R5)
6205 004124 023715 001124 CMP $GDDAT,(R5)
6206 004130 001401 BEQ 3$
6207 004132 104002 ERROR 2 ;CSRC ERROR
6208 004134 010337 001122 3$: MOV R3,$BDADR
6209 004140 005037 001124 CLR $GDDAT
6210 004144 011315 MOV (R3),(R5)
6211 004146 023715 001124 CMP $GDDAT,(R5)
6212 004152 001401 BEQ TST16 ;:BR IF EQUAL
6213 004154 104002 ERROR 2 ;:CSRD ERROR
6214
6215
6216
(3)
(3)
(2) 004156 000004
(2)
6217 004160 004737 013056 JSR PC,CLRCSR ;CLEAR ALL CSRS
6218 004164 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6219 004170 012737 004220 001110 MOV #1$,$LPERR ;SET UP SCOPE ADDRESS
6220 004176 013700 001342 MOV DRCSA,R0 ;GET PORT A, CSRA ADDRESS
6221 004202 013701 001352 MOV DRCSB,R1 ;GET PORT C, CSRC ADDRESS
6222 004206 013702 001346 MOV DRCSB,R2 ;CSRB ADDRESS
6223 004212 112762 000001 000001 MOV #BIT0,1(R2) ;CSRB IN OUTPUT MODE
6224 004220 010037 001122 1$: MOV R0,$BDADR ;STORE CSRA ADDRESS
6225 004224 005062 000002 CLR 2(R2) ;CLEAR DBRB
6226 004230 105061 000001 CLR 1(R1) ;PORT C,CSRC,INPUT MODE
6227 004234 112760 000001 000001 MOV #BIT0,1(R0) ;SET CSRA IN OUTPUT MODE
6228 004242 011360 000002 MOV (R3),2(R0) ;WRITE INTO DBRA
6229 004246 016104 000002 MOV 2(R1),R4 ;READ DBRC
6230 004252 012737 100700 001124 MOV #RDY!DIR!BIT7!BIT6,$GDDAT ;CSRA DIR SHOULD STAY SET
6231 MOV (R0),(R5) ;READ CSRA
6232 004260 011015 BIC #7,(R5) ;CLEAR UNDEFINED BITS
6233 004262 042715 000007 CMP $GDDAT,(R5)
6234 004266 023715 001124 BEQ 100$
6235 004272 001401 ERROR 2 ;CSRA ERROR
6236 004274 104002
6237 004276 012737 000200 001124 100$: MOV #BIT7,$GDDAT ;CSRC ADDRESS
6238 004304 010137 001122 MOV R1,$BDADR ;READ CSRC
6239 004310 011115 MOV (R1),(R5) ;CLEAR UNDEFINED BITS
6240 004312 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS
6241 004316 023715 001124 CMP $GDDAT,(R5)
6242 004322 001401 BEQ 101$
6243 004324 104002 ERROR 2 ;CSRC ERROR

```

```

:*****
:*TEST 16 TEST PORT A TO PORT C INTERACTION
:*****
TST16: SCOPE

```

```

6244 004326 013737 001354 001122 101$: MOV DRDBC,$BDADR ;STORE DBRC ADDRESS
6245 004334 011337 001124 MOV (R3),$GDDAT ;SAVE EXPECTED
6246 004340 010415 MOV R4,(R5) ;DBRC CONTENTS
6247 004342 023715 001124 CMP $GDDAT,(R5) ;CHECK PORT C,DBRC
6248 004346 001401 BEQ 2$ ;BEQ TO NEXT SUBTEST
6249 004350 104002 ERROR 2 ;DBRC REG ERROR
6250 004352 005137 001124 2$: COM $GDDAT ;SET UP TO WRITE COM DATA FROM
6251 ;PORT C TO PORT A
6252 004356 013737 001344 001122 MOV DRDBA,$BDADR ;STORE DBRA ADDRESS
6253 004364 013761 001124 000002 3$: MOV $GDDAT,2(R1) ;WRITE PORT C,INPUT MODE
6254 004372 105060 000001 CLRB 1(R0) ;MAKE PORT A INPUT MODE
6255 004376 112761 000001 000001 MOVB #RITO,1(R1) ;MAKE PORT C OUTPUT MODE
6256 004404 016015 000002 MOV 2(R0),(R5) ;READ PORT A,DBRA
6257 004410 023715 001124 CMP $GDDAT,(R5) ;PORT A =PORT C?
6258 004414 001401 BEQ 4$ ;CHECK DBRB FOR NO DATA CHANGE
6259 004416 104002 ERROR 2 ;PORT A,DBRA REG ERROR
6260 004420 013737 001350 001122 4$: MOV DRDBB,$BDADR ;STORE DBRB ADDRESS
6261 004426 005037 001124 CLR $GDDAT ;CONTENTS SHOULD STAY ZERO
6262 004432 016215 000002 MOV 2(R2),(R5) ;READ DBRB FOR CLEAR
6263 004436 023715 001124 CMP $GDDAT,(R5) ;CHECK FOR DBRB CLEAR
6264 004442 001401 BEQ 5$ ;YES,CONTINUE
6265 004444 104002 ERROR 2 ;DBRB INTERACTION ERROR
6266 004446 005723 5$: TST (R3)+ ;INC FOR NEXT PATTERN
6267 004450 020327 013464 CMP R3,#ENDDAT ;CHECK FOR END
6268 004454 001261 BNE 1$ ;DO NEXT PATTERN

```

```

6269
6270
6271 ;*****
(3) ;*TEST 17 TEST PORT B TO PORT D INTERACTION
(3) ;*****
(2) 004456 000004 TST17: SCOPE
(2)

```

```

6272 004460 004737 013056 JSR PC,CLRCR ;CLEAR ALL CSRS
6273 004464 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6274 004470 012737 004520 001110 MOV #1$,$LPERR ;SET UP SCOPE ADDRESS
6275 004476 013700 001346 MOV DRCSB,R0 ;GET PORT B,CSRB ADDRESS
6276 004502 013701 001356 MOV DRCSA,R1 ;GET PORT D,CSRD ADDRESS
6277 004506 013702 001342 MOV DRCSA,R2 ;STORE CSRA ADDRESS
6278 004512 112762 000001 000001 MOVB #BITO,1(R2) ;CSRA IN OUTPUT MODE
6279 004520 010037 001122 1$: MOV R0,$BDADR ;STORE CSRB ADDRESS
6280 004524 005062 000002 CLR 2(R2) ;CLEAR DBRA
6281 004530 105061 000001 CLRB 1(R1) ;PORT D,CSRD,INPUT MODE
6282 004534 112760 CJO001 000001 MOVB #BITO,1(R0) ;SET CSRB IN OUTPUT MODE
6283 004542 011360 000002 MOV (R3),2(R0) ;WRITE INTO DBRB
6284 004546 016104 000002 MOV 2(R1),R4 ;DBRD CONTENTS
6285 004552 012737 100400 001124 MOV #RDY!DIR,$GDDAT ;SAVE EXPECTED
6286 004560 011015 MOV (R0),(R5) ;READ CSRB
6287 004562 023715 001124 CMP $GDDAT,(R5)
6288 004566 001401 BEQ 100$
6289 004570 104002 ERROR 2 ;CSRB ERROR
6290 004572 005037 001124 100$: CLR $GDDAT ;SAVE EXPECTED
6291 004576 010137 001122 MOV R1,$BDADR ;SAVE CSRD ADDRESS
6292 004602 011115 MOV (R1),(R5)
6293 004604 023715 001124 CMP $GDDAT,(R5)
6294 004610 001401 BEQ 101$
6295 004612 104002 ERROR 2 ;CSRD ERROR

```

```

6296 004614 013737 001360 001122 101$: MOV DRDBD,$BDADR ;SAVE DBRD ADDRESS
6297 004622 011337 001124 MOV (R3),$GDDAT ;SAVE EXPECTED
6298 004626 010415 MOV R4,(R5) ;DBRD CONTENTS
6299 004630 023715 001124 CMP $GDDAT,(R5) ;CHECK PORT D,DBRD
6300 004634 001401 BEQ 2$ ;BEQ TO NEXT SUBTEST
6301 004636 104002 ERROR 2 ;DBRD REG ERROR
6302 004640 005137 001124 2$: COM $GDDAT ;SET UP TO WRITE COM DATA FROM
6303 ;PORT D TO PORT B
6304 004644 013737 001350 001122 MOV DRDBB,$BDADR ;STORE DBRB ADDRESS
6305 004652 013761 001124 000002 3$: MOV $GDDAT,2(R1) ;WRITE PORT D,INPUT MODE
6306 004660 105060 000001 CLRB 1(R0) ;MAKE PORT B INPUT MODE
6307 004664 112761 000001 000001 MOVB #BIT0,1(R1) ;MAKE PORT D OUTPUT MODE
6308 004672 016015 000002 MOV 2(R0),(R5) ;READ PORT B,DBRB
6309 004676 023715 001124 CMP $GDDAT,(R5) ;PORT B =PORT D?
6310 004702 001401 BEQ 4$ ;CHECK DBRA FOR NO DATA CHANGE
6311 004704 104002 ERROR 2 ;PORT B,DBRB REG ERROR
6312 004706 013737 001344 001122 4$: MOV DRDBA,$BDADR ;STORE DBRA ADDRESS
6313 004714 005037 001124 CLR $GDDAT ;CONTENTS = 0
6314 004720 016215 000002 MOV 2(R2),(R5) ;READ DBRA FOR CLEAR
6315 004724 023715 001124 CMP $GDDAT,(R5) ;DBRA CLEAR?
6316 004730 001401 BEQ 5$ ;YES,CONTINUE
6317 004732 104002 ERROR 2 ;DBRA INTERACTION ERROR
6318 004734 005723 5$: TST (R3)+ ;INC FOR NEXT PATTERN
6319 004736 020327 013464 CMP R3,#ENDDAT ;CHECK FOR END
6320 004742 001266 BNE 1$ ;DO NEXT PATTERN

```

```

6321
6322
6323 ;*****
6324 (3) ;*TEST 20 TEST PORT C TO PORT A INTERACTION
6325 (3) ;*****
6326 (2) 004744 000004 TST20: SCOPE
6327 (2)

```

```

6324 004746 004737 013056 JSR PC,CLRCR ;CLEAR ALL CSRS
6325 004752 012703 013164 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
6326 004756 012737 005006 001110 MOV #1$,$LPERR ;SET UP SCOPE ADDRESS
6327 004764 013700 001352 MOV DRCSA,R0 ;GET PORT C, CSRC ADDRESS
6328 004770 013701 001342 MOV DRCSA,R1 ;GET PORT A, CSRA ADDRESS
6329 004774 013702 001356 MOV DRCSA,R2 ;STORE CSRD ADDRESS
6330 005000 112762 000001 000001 MOVB #BIT0,1(R2) ;CSRD IN OUTPUT MODE
6331 005006 010037 001122 1$: MOV R0,$BDADR ;STORE CSRC ADDRESS
6332 005012 005062 000002 CLR 2(R2) ;CLEAR DBRD
6333 005016 105061 000001 CLRB 1(R1) ;PORT A,CSRA,INPUT MODE
6334 005022 112760 000001 000001 MOVB #BIT0,1(R0) ;SET CSRC IN OUTPUT MODE
6335 005030 011360 000002 MOV (R3),2(R0) ;WRITE INTO DBRC
6336 005034 016104 000002 MOV 2(R1),R4 ;READ DBRA
6337 005040 012737 100600 001124 MOV #RDY!DIR!BIT7,$GDDAT ;CSRC DIR SHOULD BE SET
6338 ;CSRC DIR SHOULD BE SET
6339 005046 011015 MOV (R0),(R5) ;READ CSRC
6340 005050 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS
6341 005054 023715 001124 CMP $GDDAT,(R5)
6342 005060 001401 BEQ 100$
6343 005062 104002 ERROR 2 ;CSRC ERROR
6344 005064 012737 000300 001124 100$: MOV #BIT7!BIT6,$GDDAT
6345 005072 010137 001122 MOV R1,$BDADR ;CSRA ADDRESS
6346 005076 011115 MOV (R1),(R5) ;READ CSRA
6347 005100 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS

```

6348	005104	023715	001124			CMP	\$GDDAT,(R5)	
6349	005110	001401				BEQ	101\$	
6350	005112	104002				ERROR	2	:CSRA ERROR
6351	005114	013737	001344	001122	101\$:	MOV	DRDBA,\$BDADR	:STORE DBRA ADDRESS
6352	005122	011337	001124			MOV	(R3),\$GDDAT	:SAVE EXPECTED
6353	005126	010415				MOV	R4,(R5)	:DBRA CONTENTS
6354	005130	023715	001124			CMP	\$GDDAT,(R5)	:CHECK PORT A,DBRA
6355	005134	001401				BEQ	2\$	:BEQ TO NEXT SUBTEST
6356	005136	104002				ERROR	2	:DBRA REG ERROR
6357	005140	005137	001124		2\$:	COM	\$GDDAT	:SET UP TO WRITE COM DATA FROM
6358								:PORT A TO PORT C
6359	005144	013737	001354	001122		MOV	DRDBC,\$BDADR	:STORE DBRC ADDRESS
6360	005152	013761	001124	000002	3\$:	MOV	\$GDDAT,2(R1)	:WRITE PORT A,INPUT MODE
6361	005160	105060	000001			CLRB	1(R0)	:MAKE PORT C INPUT MODE
6362	005164	112761	000001	000001		MOVB	#BIT0,1(R1)	:MAKE PORT A OUTPUT MODE
6363	005172	016015	000002			MOV	2(R0),(R5)	:READ PORT C,DBRC
6364	005176	023715	001124			CMP	\$GDDAT,(R5)	:PORT C =PORT A?
6365	005202	001401				BEQ	4\$	:CHECK DBRD FOR NO DATA CHANGE
6366	005204	104002				ERROR	2	:PORT C,DBRC REG ERROR
6367	005206	013737	001360	001122	4\$:	MOV	DRDBD,\$BDADR	:STORE DBRD ADDRESS
6368	005214	005037	001124			CLR	\$GDDAT	:DBRD = 0 EXPECTED
6369	005220	016215	000002			MOV	2(R2),(R5)	:READ DBRD FOR CLEAR
6370	005224	023715	001124			CMP	\$GDDAT,(R5)	:IS DBRD CLEAR?
6371	005230	001401				BEQ	5\$	:YES,CONTINUE
6372	005232	104002				ERROR	2	:DBRD INTERACTION ERROR
6373	005234	005723			5\$:	TST	(R3)+	:INC FOR NEXT PATTERN
6374	005236	020327	013464			CMP	R3,#ENDDAT	:CHECK FOR END
6375	005242	001261				BNE	1\$	:DO NEXT PATTERN
6376								
6377								
(3)								::*****
(3)								:*TEST 21 EST PORT D TO PORT B INTERACTION
(2)	005244	000004						::*****
(2)								TST21: SCOPE
6378	005246	004737	013056			JSR	PC,CLRCR	:CLEAR ALL CSRS
6379	005252	012703	013164			MOV	#BEGPAT,R3	:GET DATA PATTERN TABLE
6380	005256	012737	005306	001110		MOV	#1\$,SLPERR	:SET UP SCOPE ADDRESS
6381	005264	013700	001356			MOV	DRDSD,R0	:GET PORT D, CSR D ADDRESS
6382	005270	013701	001346			MOV	DRCSB,R1	:GET PORT B, CSRB ADDRESS
6383	005274	013702	001352			MOV	DRDSC,R2	:STORE CSRC ADDRESS
6384	005300	112762	000001	000001		MOVB	#BIT0,1(R2)	:CSRC IN OUTPUT MODE
6385	005306	010037	001122		1\$:	MOV	R0,\$BDADR	:STORE CSR D ADDRESS
6386	005312	005062	000002			CLR	2(R2)	:CLEAR DBRC
6387	005316	105061	000001			CLRB	1(R1)	:PORT B,CSRB,INPUT MODE
6388	005322	112760	000001	000001		MOVB	#BIT0,1(R0)	:SET CSR D IN OUTPUT MODE
6389	005330	011360	000002			MOV	(R3),2(R0)	:WRITE INTO DBRD
6390	005334	016104	000002			MOV	2(R1),R4	:READ DBRB
6391	005340	012737	100400	001124		MOV	#RDY!DIR,\$GDDAT	:SAVE EXPECTED
6392	005346	011015				MOV	(R0),(R5)	:READ CSR D
6393	005350	023715	001124			CMP	\$GDDAT,(R5)	
6394	005354	001401				BEQ	100\$	
6395	005356	104002				ERROR	2	:CSR D ERROR
6396	005360	005037	001124		100\$:	CLR	\$GDDAT	:SAVE EXPECTED
6397	005364	010137	001122			MOV	R1,\$BDADR	:SAVE CSRB ADDRESS
6398	005370	011115				MOV	(R1),(R5)	
6399	005372	023715	001124			CMP	\$GDDAT,(R5)	



6400	005376	001401				BEQ	101\$	
6401	005400	104002				ERROR	2	:CSRB ERROR
6402	005402	013737	001350	001122	101\$:	MOV	DRDBB,\$BDADR	:SAVE DBRB ADDRESS
6403	005410	011337	001124			MOV	(R3),\$GDDAT	:SAVE EXPECTED
6404	005414	010415				MOV	R4,(R5)	:DBRB CONTENTS
6405	005416	023715	001124			CMP	\$GDDAT,(R5)	:CHECK PORT B,DBRB
6406	005422	001401				BEQ	2\$	:BEQ TO NEXT SUBTEST
6407	005424	104002				ERROR	2	:DBRB REG ERROR
6408	005426	005137	001124		2\$:	COM	\$GDDAT	:SET UP TO WRITE COM DATA FROM
6409								:PORT B TO PORT D
6410	005432	013737	001360	001122		MOV	DRDBD,\$BDADR	:STORE DBRD ADDRESS
6411	005440	013761	001124	000002	3\$:	MOV	\$GDDAT,2(R1)	:WRITE PORT B,INPUT MODE
6412	005446	105060	000001			CLRB	1(R0)	:MAKE PORT D INPUT MODE
6413	005452	112761	000001	000001		MOVB	#BIT0,1(R1)	:MAKE PORT B OUTPUT MODE
6414	005460	016015	000002			MOV	2(R0),(R5)	:READ PORT D,DBRD
6415	005464	023715	001124			CMP	\$GDDAT,(R5)	:PORT B =PORT D?
6416	005470	001401				BEQ	4\$	:CHECK DBRC FOR NO DATA CHANGE
6417	005472	104002				ERROR	2	:PORT D,DBRD REG ERROR
6418	005474	013737	001354	001122	4\$:	MOV	DRDBC,\$BDADR	:STORE DBRC ADDRESS
6419	005502	005037	001124			CLR	\$GDDAT	:EXPECT DBRC = 0
6420	005506	016215	000002			MOV	2(R2),(R5)	:READ DBRC
6421	005512	023715	001124			CMP	\$GDDAT,(R5)	:IS DBRC = 0
6422	005516	001401				BEQ	5\$	:YES,CONTINUE
6423	005520	104002				ERROR	2	:DBRC INTERACTION ERROR
6424	005522	005723			5\$:	TST	(R3)+	:INC FOR NEXT PATTERN
6425	005524	020327	013464			CMP	R3,#ENDDAT	:CHECK FOR END
6426	005530	001266				BNE	1\$	:DO NEXT PATTERN

\*\*\*\*\*  
 :\*TEST 22 TEST GROUP 1,2 IMR,IRR,ACR WITH CHIP RESET  
 \*\*\*\*\*  
 TST22: SCOPE

6429	005534	004737	013056			JSR	PC,CLRCSR	:CLEAR ALL CSks
6430	005540	012702	000002			MOV	#2,R2	:TWO GROUPS TO TEST
6431	005544	013700	001342			MOV	DRCSA,R0	:STORE CSRA
6432	005550	013701	001346			MOV	DRCSB,R1	:STORE CSRB
6433	005554	004737	013126			JSR	PC,CLRIRR	:CLEAR IRR REGISTERS
6434	005560	010137	001122			MOV	R1,\$BDADR	:STORE ADDRESS
6435	005564	012737	000377	001124	111\$:	MOV	#377,\$GDDAT	:STORE EXPECTED DATA
6436	005572	112710	000244			MOVB	#MIMR,(R0)	:LOAD MODE BITS FOR IMR
6437	005576	111115				MOVB	(R1),(R5)	:READ IMR REGISTER
6438	005600	023715	001124			CMP	\$GDDAT,(R5)	
6439	005604	001401				BEQ	1\$	
6440	005606	104005				ERROR	5	:IMR REG ERROR
6441	005610	005037	001124		1\$:	CLR	\$GDDAT	:STORE EXPECTED
6442	005614	112710	000250			MOVB	#MIRR,(R0)	:LOAD MODE BITS FOR IRR
6443	005620	111115				MOVB	(R1),(R5)	:READ IRR REGISTER
6444	005622	023715	001124			CMP	\$GDDAT,(R5)	
6445	005626	001401				BEQ	2\$	
6446	005630	104003				ERROR	3	:IRR REG ERROR
6447	005632	112710	000254		2\$:	MOVB	#MACR,(R0)	:LOAD MODE BITS FOR ACR
6448	005636	111115				MOVB	(R1),(R5)	:READ ACR REGISTER
6449	005640	023715	001124			CMP	\$GDDAT,(R5)	
6450	005644	001401				BEQ	3\$	
6451	005646	104004				ERROR	4	:ACR REG ERROR

```

6452 005650 005302          3$:   DEC      R2          :FINISHED BOTH GROUPS?
6453 005652 001405          BEQ      TST23        ;;BR IF EQUAL
6454 005654 013700 001352   MOV      DRCSA,R0
6455 005660 013701 001356   MOV      DRCSB,R1
6456 005664 000735          BR       111$
6457
6458          ::*****
(3)          :*TEST 23      TEST GROUPS 1 AND 2 ACR UNIQUENESS
(3)          ::*****
(2) 005666 000004          TST23: SCOPE
(2)
6459 005670 004737 013056   JSR      PC,CLRCR    :CLEAR ALL CSRS
6460 005674 013700 001342   MOV      DRCSA,R0    :CSRA ADDRESS
6461 005700 013701 001346   MOV      DRCSB,R1    :CSRB ADDRESS
6462 005704 012703 000002   MOV      #2,R3       :COUNTER FOR TESTING TWO GROUPS
6463 005710 004737 013126   JSR      PC,CLRIRR   :CLEAR IRR REGS WITH CHIP RESET
6464 005714 010137 001122   MOV      R1,$BCADR   :STORE ADDRESS
6465 005720 012737 000252 001124 111$: MOV      #252,$GDDAT :STORE EXPECTED
6466 005726 112710 000254   MOV      #MACR,(R0)  :LOAD MODE BITS FOR ACR
6467 005732 112710 000300   MOV      #PACR,(R0)  :PRESELECT ACR FOR WRITING
6468 005736 113711 001124   MOV      $GDDAT,(R1) :WRITE INTO DATA PORT
6469 005742 111115          MOV      (R1),(R5)   :STORE DATA FOR COMPARE
6470 005744 023715 001124   CMP      $GDDAT,(R5) :CHECK ACR RESULTS
6471 005750 001401          BEQ      1$
6472 005752 104004          ERROR    4           :ACR ERROR
6473 005754 112710 000244 001124 1$: MOV      #MIMR,(R0) :CHANGE TO IMR REGISTER
6474 005760 012737 000377   MOV      #377,$GDDAT :STORE EXPECTED
6475 005766 111115          MOV      (R1),(R5)   :READ IMR
6476 005770 023715 001124   CMP      $GDDAT,(R5) :SHOULD STILL BE ALL 1'S
6477 005774 001401          BEQ      2$
6478 005776 104005          ERROR    5           :IMR REG ERROR
6479 006000 112710 000240 2$: MOV      #MISR,(R0) :LOAD MODE BITS FOR ISR
6480 006004 005037 001124   CLR      $GDDAT     :STORE EXPECTED
6481 006010 111115          MOV      (R1),(R5)   :READ ISR
6482 006012 023715 001124   CMP      $GDDAT,(R5) :ISR SHOULD BE CLEARED
6483 006016 001401          BEQ      3$
6484 006020 104006          ERROR    6           :ISR REG ERROR
6485 006022 112710 000250 3$: MOV      #MIRR,(R0) :LOAD MODE BITS FOR IRR
6486 006026 111115          MOV      (R1),(R5)   :READ IRR
6487 006030 023715 001124   CMP      $GDDAT,(R5) :IRR SHOULD BE CLEARED
6488 006034 001401          BEQ      4$
6489 006036 104003          ERROR    3           :IRR REG ERROR
6490 006040 105010          CLRB     (R0)       :CHIP RESET GROUP 1
6491 006042 112710 000254   MOV      #MACR,(R0)  :LOAD MODE BITS FOR ACR
6492 006046 111115          MOV      (R1),(R5)   :READ ACR
6493 006050 023715 001124   CMP      $GDDAT,(R5) :ACR SHOULD BE CLEARED
6494 006054 001401          BEQ      5$
6495 006056 104004          ERROR    4           :ACR REG ERROR
6496 006060 005303          DEC      R3
6497 006062 001405          BEQ      TST24
6498 006064 013700 001352   MOV      DRCSA,R0
6499 006070 013701 001356   MOV      DRCSB,R1
6500 006074 000707          BR       111$
6501
6502          ::*****
(3)          :*TEST 24      TEST GROUPS 1 AND 2 IMR UNIQUENESS
  
```

3

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 N 2  
T24 TEST GROUPS 1 AND 2 IMR UNIQUENESS PAGE 58-7

SEG 0026

```

(3)
(2) 006076 000004
(2)
6503 ;;GPA JSR PC,CLRCR :CLEAR ALL CSRS
6504 006100 004737 013126 JSR PC,CLRIRR : ** VDRCA1 CHANGES THIS ** ;;GPA
6505 006104 013700 001342 MOV DRCSA,R0 :CSRA ADDRESS
6506 006110 013701 001346 MOV DRCSB,R1 :CSRB ADDRESS
6507 006114 012703 000002 MOV #2,R3 :COUNTER FOR TWO GROUP TESTING
6508 006120 010137 001122 111$: MOV R1,$BDADR :STORE ADDRESS
6509 006124 012737 000252 001124 MOV #252,$GDDAT :STORE EXPECTED
6510 006137 112710 000244 MOVB #MIMR,(R0) :LOAD MODE BITS FOR IMR
6511 006136 112710 000260 MOVB #PIMR,(R0) :PRESELECT IMR FOR WRITING
6512 006142 113711 001124 MOVB $GDDAT,(R1) :WRITE INTO DATA PORT
6513 006146 111115 MOVB (R1),(R5) :STORE DATA FOR COMPARE
6514 006150 023715 001124 CMP $GDDAT,(R5) :CHECK IMR RESULTS
6515 006154 001401 BEQ 1$
6516 006156 104005 ERROR 5 :IMR ERROR
6517 006160 112710 000254 1$: MOVB #MACR,(R0) :CHANGE TO ACR REGISTER
6518 006164 005037 001124 CLR $GDDAT :STORE EXPECTED
6519 006170 111115 MOVB (R1),(R5) :READ ACR
6520 006172 023715 001124 CMP $GDDAT,(R5) :SHOULD STILL BE CLEARED
6521 006176 001401 BEQ 2$
6522 006200 104004 ERROR 4 :ACR REG ERROR
6523 006202 112710 000240 2$: MOVB #MISR,(R0) :LOAD MODE BITS FOR ISR
6524 006206 111115 MOVB (R1),(R5) :READ ISR
6525 006210 023715 001124 CMP $GDDAT,(R5) :ISR SHOULD BE CLEARED
6526 006214 001401 BEQ 3$
6527 006216 104006 ERROR 6 :ISR REG ERROR
6528 006220 112710 000250 3$: MOVB #MIRR,(R0) :LOAD MODE BITS FOR IRR
6529 006224 111115 MOVB (R1),(R5) :READ IRR
6530 006226 023715 001124 CMP $GDDAT,(R5) :IRR SHOULD BE CLEARED
6531 006232 001401 BEQ 4$
6532 006234 104003 ERROR 3 :IRR REG ERROR
6533 006236 105010 4$: CLRB (R0) :CHIP RESET GROUP 1
6534 006240 112710 000244 MOVB #MIMR,(R0) :LOAD MODE BITS FOR IMR
6535 006244 012737 000377 001124 MOV #377,$GDDAT :STORE EXPECTED
6536 006252 111115 MOVB (R1),(R5) :READ IMR
6537 006254 023715 001124 CMP $GDDAT,(R5) :IMR SHOULD BE ALL ONES
6538 006260 001401 BEQ 5$ :DO NEXT GROUP
6539 006262 104005 ERROR 5 :IMR REG ERROR
6540 006264 005303 5$: DEC R3 :DO GROUPS 1 AND GROUP 2
6541 006266 001405 BEQ TST25 :BR IF BOTH GROUPS TESTED
6542 006270 013700 001352 MOV DRCSA,R0 :SET UP TO TEST GROUP 2
6543 006274 013701 001356 MOV DRCSB,R1 :GROUP 2 DATA PORT
6544 006300 000707 BR 111$ :DO GROUP 2 IMR UNIQUENESS
6545
6546

```

```

(3)
(3)
(2) 006302 000004
(2)
6547 006304 004737 013056 JSR PC,CLRCR :CLEAR ALL CSRS
6548 006310 013700 001342 MOV DRCSA,R0 :CSRA ADDRESS
6549 006314 013701 001346 MOV DRCSB,R1 :CSRB ADDRESS
6550 006320 012703 000002 MOV #2,R3 :COUNTER FOR TESTING TWO GROUPS
6551 006324 004737 013126 JSR PC,CLRIRR :CLEAR IRR REGS WITH CHIP RESET

```

```

6552 006330 010137 001122      111$: MOV R1,$BDADR ;STORE ADDRESS
6553 006334 012737 000200 001124 MOV #200,$GDDAT ;STORE EXPECTED
6554 006342 112710 000250 MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR
6555 006346 112710 000137 MOVB #137,(R0) ;SET SINGLE IRR BIT7
6556 006352 111115 MOVB (R1),(R5) ;STORE DATA FOR COMPARE
6557 006354 023715 001124 CMP $GDDAT,(R5) ;CHECK IRR RESULTS
6558 006360 001401 BEQ 1$ ;
6559 006362 104003 ERROR 3 ;IRR ERROR
6560 006364 112710 000244 1$: MOVB #MIMR,(R0) ;CHANGE TO IMR REGISTER
6561 006370 012737 000377 001124 MOV #377,$GDDAT ;STORE EXPECTED
6562 006376 111115 MOVB (R1),(R5) ;READ IMR
6563 006400 023715 001124 CMP $GDDAT,(R5) ;SHOULD STILL BE ALL 1'S
6564 006404 001401 BEQ 2$ ;
6565 006406 104005 ERROR 5 ;IMR REG ERROR
6566 006410 112710 000240 2$: MOVB #MISR,(R0) ;LOAD MODE BITS FOR ISR
6567 006414 005037 001124 CLR $GDDAT ;STORE EXPECTED
6568 006420 111115 MOVB (R1),(R5) ;READ ISR
6569 006422 023715 001124 CMP $GDDAT,(R5) ;ISR SHOULD BE CLEARED
6570 006426 001401 BEQ 3$ ;
6571 006430 104006 ERROR 6 ;ISR REG ERROR
6572 006432 112710 000254 3$: MOVB #MACR,(R0) ;LOAD MODE BITS FOR ACR
6573 006436 111115 MOVB (R1),(R5) ;READ ACR
6574 006440 023715 001124 CMP $GDDAT,(R5) ;ACR SHOULD BE CLEARED
6575 006444 001401 BEQ 4$ ;
6576 006446 104004 ERROR 4 ;ACR REG ERROR
6577 006450 105010 4$: CLRB (R0) ;CHIP RESET GROUP 1
6578 006452 112710 000250 MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR
6579 006456 111115 MOVB (R1),(R5) ;READ IRR
6580 006460 023715 001124 CMP $GDDAT,(R5) ;IRR SHOULD BE CLEARED
6581 006464 001401 BEQ 5$ ;
6582 006466 104003 ERROR 3 ;IRR REG ERROR
6583 006470 005303 5$: DEC R3 ;TEST GROUPS 1 AND 2
6584 006472 001405 BEQ TST26 ;BR IF BOTH GROUPS TESTED
6585 006474 013700 001352 MOV DRCSA,R0 ;GROUP 2 CONTROL PORT
6586 006500 013701 001356 MOV DRCSB,R1 ;GROUP 2 DATA PORT
6587 006504 000711 BR 111$ ;TEST GROUP 2 ACR UNIQUENESS

```

```

6588
6589
6590 ::*****
(3) ;*TEST 26 TEST GROUPS 1,2 ACR WITH PATTERNS
(3) ;*****
(2) 006506 000004 TST26: SCOPE
(2)

```

```

6591 006510 004737 013056      JSR PC,CLRCR ;CLEAR ALL CSRS
6592 006514 012702 000002 001110 MOV #2,R2 ;TEST TWO GROUPS
6593 006520 012737 006546 MOV #1,$LPERR ;SET UP LOOP RETURN
6594 006526 013700 001342 MOV DRCSA,R0 ;STORE CSRA ADDRESS
6595 006532 013701 001346 MOV DRCSB,R1 ;STORE CSRB ADDRESS
6596 006536 012703 013164 111$: MOV #BEGPAT,R3 ;SET UP PATTERN TABLE
6597 006542 010137 001122 MOV R1,$BDADR ;STORE ADDRESS
6598 006546 112710 000254 1$: MOVB #MACR,(R0) ;LOAD MODE BITS FOR ACR
6599 006552 112710 000300 MOVB #PACR,(R0) ;PRESELECT ACR FOR WRITING
6600 006556 111337 001124 MOVB (R3),$GDDAT ;STORE EXPECTED
6601 006562 111311 MOVB (R3),(R1) ;WRITE INTO ACR
6602 006564 111115 MOVB (R1),(R5) ;READ OUT OF ACR
6603 006566 023715 001124 CMP $GDDAT,(R5)

```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 C 3  
T26 TEST GROUPS 1,2 ACR WITH PATTERNS PAGE 58-9

SEQ 0028

6604 006572 001401  
6605 006574 104004  
6606 006576 005723  
6607 006600 020327 013464  
6608 006604 001360  
6609 006606 005302  
6610 006610 001405  
6611 006612 013700 001352  
6612 006616 013701 001356  
6613 006622 000745

2\$: BEQ 2\$  
ERROR 4 :ACR REGISTER ERROR  
TST (R3)+ :INC FOR NEXT PATTERN  
CMP R3,#ENDDAT :CHECK FOR TABLE END  
BNE 1\$ :W/R NEXT PATTERN  
DEC R2 :FINISHED BOTH GROUPS?  
BEQ TST27 :;BR IF EQUAL  
MOV DRCSC,R0  
MOV DRCSD,R1  
BR 111\$ :DO NEXT GROUP

6614  
6615  
(3)  
(3)  
(2) 0^4624 000004  
(2)

::\*\*\*\*\*  
:\*TEST 27 TEST GROUPS 1,2 IMR WITH PATTERNS  
:\*\*\*\*\*  
TST27: SCOPE

6616 006626 004737 013056  
6617 006632 012702 000002  
6618 006636 012737 006664 001110  
6619 006644 013700 001342  
6620 006650 013701 001346  
6621 006654 012703 013164  
6622 006660 010137 001122  
6623 006664 112710 000244  
6624 006670 112710 000260  
6625 006674 111337 001124  
6626 006700 111311  
6627 006702 111115  
6628 006704 023715 001124  
6629 006710 001401  
6630 006712 104005  
6631 006714 005723  
6632 006716 020327 013464  
6633 006722 001360  
6634 006724 005302  
6635 006726 001405  
6636 006730 013700 001352  
6637 006734 013701 001356  
6638 006740 000745

JSR PC,CLRCR :CLEAR ALL CSRS  
MOV #2,R2 :TEST TWO GROUPS  
MOV #1\$,\$LPERR :SET UP LOOP RETURN  
MOV DRCSA,R0 :STORE CSRA ADDRESS  
MOV DRCRB,R1 :STORE CSRB ADDRESS  
111\$: MOV #BEGPAT,R3 :SET UP PATTERN TABLE  
MOV R1,\$BDADR :STORE ADDRESS  
1\$: MOVB #MIMR,(R0) :LOAD MODE BITS FOR IMR  
MOVB #PIMR,(R0) :PRESELECT IMR FOR WRITING  
MOVB (R3),\$GDDAT :STORE EXPECTED  
MOVB (R3),(R1) :WRITE INTO IMR  
MOVB (R1),(R5) :READ OUT OF IMR  
CMP \$GDDAT,(R5)  
BEQ 2\$  
ERROR 5 :IMR REGISTER ERROR  
2\$: TST (R3)+ :INC FOR NEXT PATTERN  
CMP R3,#ENDDAT :CHECK FOR TABLE END  
BNE 1\$ :W/R NEXT PATTERN  
DEC R2 :FINISHED BOTH GROUPS?  
BEQ TST30 :;BR IF EQUAL  
MOV DRCSC,R0  
MOV DRCSD,R1  
BR 111\$ :DO NEXT GROUP

6639  
6640  
(3)  
(3)  
(2) 006742 000004  
(2)

::\*\*\*\*\*  
:\*TEST 30 TEST GROUP 1,2 CLEAR IMR INSTR.  
:\*\*\*\*\*  
TST30: SCOPE

6641 006744 004737 013056  
6642 006750 012704 000002  
6643 006754 013700 001342  
6644 006760 013701 001346  
6645 006764 010137 001122  
6646 006770 005037 001124  
6647 006774 112710 000244  
6648 007000 112710 000040  
6649 007004 111115  
6650 007006 023715 001124  
6651 007012 001401

JSR PC,CLRCR :CLEAR ALL CSRS  
MOV #2,R4 :COUNTER FOR TWO GROUPS  
MOV DRCSA,R0 :CSRA ADDRESS  
MOV DRCRB,R1 :CSRB ADDRESS  
111\$: MOV R1,\$BDADR :STORE ADDRESS  
CLR \$GDDAT :EXPECTED DATA  
MOVB #MIMR,(R0) :LOAD MODE BITS TO READ IMR  
MOVB #CIMR,(R0) :CLEAR IMR COMMAND  
MOVB (R1),(R5) :READ DATA PORT  
CMP \$GDDAT,(R5)  
BEQ 1\$

```

6652 007014 104005
6653 007016 005304
6654 007020 001405
6655 007022 013700 001352
6656 007026 013701 001356
6657 007032 000754
6658
6659
(3)
(3)
(2) 007034 000004
(2)
6660 007036 004737 013056
6661 007042 012704 000002
6662 007046 013700 001342
6663 007052 013701 001346
6664 007056 010137 001122
6665 007062 012737 000377 001124
6666 007070 112710 000244
6667 007074 112710 000040
6668 007100 112710 000060
6669 007104 111115
6670 007106 023715 001124
6671 007112 001401
6672 007114 104005
6673 007116 005304
6674 007120 001405
6675 007122 013700 001352
6676 007126 013701 001356
6677 007132 000751
6678
6679
(3)
(3)
(2) 007134 000004
(2)
6680 007136 004737 013056
6681 007142 012704 000002
6682 007146 012737 007204 001110
6683 007154 013700 001342
6684 007160 013701 001346
6685 007164 010137 001122
6686 007170 012703 013330
6687 007174 112710 000244
6688 007200 012702 000050
6689 007204 111337 001124
6690 007210 112710 000060
6691 007214 110210
6692 007216 111115
6693 007220 023715 001124
6694 007224 001401
6695 007226 104005
6696 007230 005723
6697 007232 020327 013350
6698 007236 001402
6699 007240 005202

      ERROR 5 ;ERROR ,IMR SHOULD BE CLEARED
1$: DEC R4 ;FINISHED BOTH GROUPS?
   BEQ TST31 ;:BR IF BOTH GROUPS TESTED
   MOV DRCSC,R0 ;SETUP FOR GROUP 2
   MOV DRCSD,R1
   BR 111$ ;DO IMR TEST WITH GROUP 2

;*****
;*TEST 31 TEST GROUP 1,2 SET IMR INSTR.
;*****
TST31: SCOPE

      JSR PC,CLRCR ;CLEAR ALL CSRS
      MOV #2,R4 ;COUNTER FOR TWO GROUPS
      MOV DRCSA,R0 ;CSRA ADDRESS
      MOV DRCRB,R1 ;CSRB ADDRESS
111$: MOV R1,$BDADR ;STORE ADDRESS
      MOV #377,$GDDAT ;EXPECTED DATA
      MOVB #MIMR,(R0) ;LOAD MODE BITS TO READ IMR
      MOVB #CIMR,(R0) ;CLEAR IMR
      MOVB #SIMR,(R0) ;SET IMR COMMAND
      MOVB (R1),(R5) ;READ DATA PORT
      CMP $GDDAT,(R5)
      BEQ 1$
      ERROR 5 ;ERROR ,IMR SHOULD BE SET
1$: DEC R4 ;FINISHED BOTH GROUPS?
   BEQ TST32 ;:BR IF BOTH GROUPS TESTED
   MOV DRCSC,R0 ;SETUP FOR GROUP 2
   MOV DRCSD,R1
   BR 111$ ;DO IMR TEST WITH GROUP 2

;*****
;*TEST 32 TEST GROUP 1,2 CLEAR SINGLE IMR BIT INSTR.
;*****
TST32: SCOPE

      JSR PC,CLRCR ;CLEAR ALL CSRS
      MOV #2,R4 ;GROUP COUNTER
      MOV #1,$LPERR ;SCOPE RETURN ADDRESS
      MOV DRCSA,R0 ;CSRA ADDRESS
      MOV DRCRB,R1 ;CSRB ADDRESS
111$: MOV R1,$BDADR ;STORE ADDRESS
      MOV #BGCHP4,R3 ;GOOD DATA PATTERN TABLE
      MOVB #MIMR,(R0) ;LOAD MODE BITS FOR IMR
      MOV #CSIMR,R2 ;CLEAR SINGLE IMR BIT VALUE
1$: MOVB (R3),$GDDAT ;STORE EXPECTED
      MOVB #SIMR,(R0) ;SET ALL IMR BITS
      MOVB R2,(R0) ;CLEAR SINGLE IMR BIT
      MOVB (R1),(R5) ;READ DATA PORT
      CMP $GDDAT,(R5)
      BEQ 2$
      ERROR 5 ;IMR REG ERROR
2$: TST (R3)+ ;INC EXPECTED DATA TABLE
      CMP R3,#EDCHP4 ;CHECK FOR END
      BEQ 3$
      INC R2 ;SET UP TO CLEAR NEXT IMR BIT
  
```

```

6700 007242 000760          BR      1$          ;CLEAR NEXT IMR BIT
6701 007244 005304      3$: DEC      R4          ;DO GROUPS 1 AND 2
6702 007246 001405          BEQ     TST33        ;:BR IF BOTH GROUPS TESTED
6703 007250 013700 001352    MOV     DRCSC,R0      ;CSRC ADDRESS
6704 007254 013701 001356    MOV     DRCSD,R1     ;CSRD ADDRESS
6705 007260 000741          BR      111$        ;DO GROUP 2
6706
6707
6708

```

```

(3)
(3)
(2) 007262 000004
(2)

```

```

6709 007264 004737 013056    JSR     PC,CLRCSR    ;CLEAR ALL CSRS
6710 007270 012704 000002    MOV     #2,R4        ;GROUP COUNTER
6711 007274 012737 007332 001110  MOV     #1$,$LPERR   ;SCOPE RETURN ADDRESS
6712 007302 013700 001342    MOV     DRCSA,R0     ;CSRA ADDRESS
6713 007306 013701 001346    MOV     DRCSE,R1     ;CSRSE ADDRESS
6714 007312 010137 001122 111$: MOV     R1,$BDADR    ;STORE ADDRESS
6715 007316 012703 013266    MOV     #BGCHP3,R3   ;GOOD DATA PATTERN TABLE
6716 007322 112710 000244    MOV     #MIMR,(R0)   ;LOAD MODE BITS FOR IMR
6717 007326 012702 000070    MOV     #SSIMR,R2    ;SET SINGLE IMR BIT VALUE
6718 007332 111337 001124 1$:  MOV     (R3),$GDDAT ;STORE EXPECTED
6719 007336 112710 000040    MOV     #CIMR,(R0)   ;CLEAR ALL IMR BITS
6720 007342 110210          MOV     R2,(R0)      ;SET SINGLE IMR BIT
6721 007344 111115          MOV     (R1),(R5)    ;READ DATA PORT
6722 007346 023715 001124    CMP     $GDDAT,(R5)
6723 007352 001401          BEQ     2$
6724 007354 104005          ERROR  5
6725 007356 005723          TST     (R3)+
6726 007360 020327 013306 2$:  CMP     R3,#EDCHP3   ;IMR REG ERROR
6727 007364 001402          BEQ     3$          ;INC EXPECTED DATA TABLE
6728 007366 005202          INC     R2          ;CHECK FOR END
6729 007370 000760          BR      1$          ;SET UP TO SET NEXT IMR BIT
6730 007372 005304          DEC     R4          ;SET NEXT IMR BIT
6731 007374 001405          BEQ     TST34       ;DO GROUPS 1 AND 2
6732 007376 013700 001352    MOV     DRCSC,R0     ;:BR IF BOTH GROUPS TESTED
6733 007402 013701 001356    MOV     DRCSD,R1     ;CSRC ADDRESS
6734 007405 000741          BR      111$        ;CSRD ADDRESS
6735
6736

```

```

(3)
(3)
(2) 007410 000004
(2)

```

```

6737 007412 004737 013056    JSR     PC,CLRCSR    ;CLEAR ALL CSRS
6738 007416 012704 000002    MOV     #2,R4        ;COUNTER FOR TWO GROUPS
6739 007422 013700 001342    MOV     DRCSE,R0     ;CSRE ADDRESS
6740 007426 013701 001346    MOV     DRCSE,R1     ;CSRE ADDRESS
6741 007432 004737 013126    JSR     PC,CLRIRR    ;CLEAR IRR REGS WITH CHIP RESET
6742 007436 010137 001122 111$: MOV     R1,$BDADR    ;STORE ADDRESS
6743 007442 012737 000377 001124  MOV     #377,$GDDAT ;EXPECTED DATA
6744 007450 112710 000250    MOV     #MIRR,(R0)   ;LOAD MODE BITS TO READ IRR
6745 007454 112710 000120    MOV     #SIRR,(R0)   ;SET IRR COMMAND
6746 007460 111115          MOV     (R1),(R5)    ;READ DATA PORT
6747 007462 023715 001124    CMP     $GDDAT,(R5)

```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25  
T34 TEST GROUP 1,2 SET IRR INSTR.

SEQ 0031

6748 007466 001401  
6749 007470 104003  
6750 007472 005304  
6751 007474 001405  
6752 007476 013700 001352  
6753 007502 013701 001356  
6754 007506 000753

1\$: BEQ 1\$  
ERROR 3 ;ERROR , IRR SHOULD BE SET  
DEC R4 ;FINISHED BOTH GROUPS?  
BEQ TST35 ;:BR IF BOTH GROUPS TESTED  
MOV DRCSC,R0 ;SETUP FOR GROUP 2  
MOV DRCSD,R1  
BR 111\$ ;DO IRR TEST WITH GROUP 2

6755  
6756  
6757

\*\*\*\*\*  
: \*TEST 35 TEST GROUP 1,2 CLEAR IRR INSTR.  
\*\*\*\*\*

(3)  
(3)  
(2) 007510 000004  
(2)

TST35: SCOPE

6758 007512 004737 013056  
6759 007516 012704 000002  
6760 007522 013700 001342  
6761 007526 013701 001346  
6762 007532 004737 013126  
6763 007536 010137 001122  
6764 007542 005037 001124  
6765 007546 112710 000250  
6766 007552 112710 000120  
6767 007556 112710 000100  
6768 007562 111115  
6769 007564 023715 001124

111\$: JSR PC,CLRCR ;CLEAR ALL CSRS  
MOV #2,R4 ;COUNTER FOR TWO GROUPS  
MOV DRCSA,R0 ;CSRA ADDRESS  
MOV DRCRB,R1 ;CSRB ADDRESS  
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
MOV R1,\$BDADR ;STORE ADDRESS  
CLR \$GDDAT ;EXPECTED DATA  
MOVB #MIRR,(R0) ;LOAD MODE BITS TO READ IRR  
MOVB #SIRR,(R0) ;SET IRR BITS  
MOVB #CIRR,(R0) ;CLEAR IRR COMMAND  
MOVB (R1),(R5) ;READ DATA PORT  
CMP \$GDDAT,(R5)  
BEQ 1\$

6770 007570 001401  
6771 007572 104003  
6772 007574 005304  
6773 007576 001405  
6774 007600 013700 001352  
6775 007604 013701 001356  
6776 007610 000752

1\$: BEQ 1\$  
ERROR 3 ;ERROR , IRR SHOULD BE CLEARED  
DEC R4 ;FINISHED BOTH GROUPS?  
BEQ TST36 ;:BR IF BOTH GROUPS TESTED  
MOV DRCSC,R0 ;SETUP FOR GROUP 2  
MOV DRCSD,R1  
BR 111\$ ;DO IRR TEST WITH GROUP 2

6777  
6778  
(3)  
(3)  
(2) 007612 000004  
(2)

\*\*\*\*\*  
: \*TEST 36 TEST GROUP 1,2 CLEAR SINGLE IRR BIT INSTR.  
\*\*\*\*\*

TST36: SCOPE

6779 007614 004737 013056  
6780 007620 012704 000002  
6781 007624 012737 007666 001110  
6782 007632 013700 001342  
6783 007636 013701 001346  
6784 007642 004737 013126  
6785 007646 010137 001122  
6786 007652 012703 013330  
6787 007656 112710 000250  
6788 007662 012702 000110  
6789 007666 111337 001124  
6790 007672 112710 000120  
6791 007676 110210  
6792 007700 111115  
6793 007702 023715 001124  
6794 007706 001401  
6795 007710 104003

111\$: JSR PC,CLRCR ;CLEAR ALL CSRS  
MOV #2,R4 ;GROUP COUNTER  
MOV #1,\$LPERR ;SCOPE RETURN ADDRESS  
MOV DRCSA,R0 ;CSRA ADDRESS  
MOV DRCRB,R1 ;CSRB ADDRESS  
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
MOV R1,\$BDADR ;STORE ADDRESS  
MOV #BGCHP4,R3 ;GOOD DATA PATTERN TABLE  
MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR  
MOV #CSIRR,R2 ;CLEAR SINGLE IRR BIT VALUE  
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED  
MOVB #SIRR,(R0) ;SET ALL IRR BITS  
MOVB R2,(R0) ;CLEAR SINGLE IRR BIT  
MOVB (R1),(R5) ;READ DATA PORT  
CMP \$GDDAT,(R5)  
BEQ 2\$  
ERROR 3 ;IRR REG ERROR



CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 PAGE 58-13  
T36 TEST GROUP 1,2 CLEAR SINGLE IRR BIT INSTR.

SEQ 0032

6796 007712 005723  
6797 007714 020327 013350  
6798 007720 001402  
6799 007722 005202  
6800 007724 000760  
6801 007726 005304  
6802 007730 001405  
6803 007732 013700 001352  
6804 007736 013701 001356  
6805 007742 000741  
6806  
6807

2\$: TST (R3)+ ;INC EXPECTED DATA TABLE  
CMP R3,#EDCHP4 ;CHECK FOR END  
BEQ 3\$  
INC R2 ;SET UP TO CLEAR NEXT IRR BIT  
BR 1\$ ;CLEAR NEXT IRR BIT  
3\$: DEC R4 ;DO GROUPS 1 AND 2  
BEQ TST37 ;:BR IF BOTH GROUPS TESTED  
MOV DRCSC,R0 ;CSRC ADDRESS  
MOV DRCSD,R1 ;CSRD ADDRESS  
BR 111\$ ;DO GROUP 2

::\*\*\*\*\*  
:\*TEST 37 TEST GROUP 1,2 SET SINGLE IRR BIT INSTR.  
:\*\*\*\*\*  
TST37: SCOPE

6808 007746 004737 013056  
6809 007752 012704 000002  
6810 007756 012737 010020 001110  
6811 007764 013700 001342  
6812 007770 013701 001346  
6813 007774 004737 013126  
6814 010000 010137 001122  
6815 010004 012703 013266  
6816 010010 112710 000250  
6817 010014 012702 000130  
6818 010020 111337 001124  
6819 010024 112710 000100  
6820 010030 110210  
6821 010032 111115  
6822 010034 023715 001124  
6823 010040 001401  
6824 010042 104003  
6825 010044 005723  
6826 010046 020327 013306  
6827 010052 001402  
6828 010054 005202  
6829 010056 000760  
6830 010060 005304  
6831 010062 001405  
6832 010064 013700 001352  
6833 010070 013701 001356  
6834 010074 000741  
6835  
6836

JSR PC,CLRCSR ;CLEAR ALL CSRS  
MOV #2,R4 ;GROUP COUNTER  
MOV #1\$,\$LPERR ;SCOPE RETURN ADDRESS  
MOV DRCSA,R0 ;CSRA ADDRESS  
MOV DRCRB,R1 ;CSR B ADDRESS  
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
111\$: MOV R1,\$BDADR ;STORE ADDRESS  
MOV #BGCHP3,R3 ;GOOD DATA PATTERN TABLE  
MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR  
MOV #SSIRR,R2 ;SET SINGLE IRR BIT VALUE  
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED  
MOVB #CIRR,(R0) ;CLEAR ALL IRR BITS  
MOVB R2,(R0) ;SET SINGLE IRR BIT  
MOVB (R1),(R5) ;READ DATA PORT  
CMP \$GDDAT,(R5)  
BEQ 2\$  
ERROR 3 ;IRR REG ERROR  
2\$: TST (R3)+ ;INC EXPECTED DATA TABLE  
CMP R3,#EDCHP5 ;CHECK FOR END  
BEQ 3\$  
INC R2 ;SET UP TO SET NEXT IRR BIT  
BR 1\$ ;SET NEXT IRR BIT  
3\$: DEC R4 ;DO GROUPS 1 AND 2  
BEQ TST40 ;:BR IF BOTH GROUPS TESTED  
MOV DRCSC,R0 ;CSRC ADDRESS  
MOV DRCSD,R1 ;CSRD ADDRESS  
BR 111\$ ;DO GROUP 2

::\*\*\*\*\*  
:\*TEST 40 TEST GROUP 1,2 CLEAR IRR+IMR INSTR.  
:\*\*\*\*\*  
TST40: SCOPE

6837 010100 004737 013056  
6838 010104 012704 000002  
6839 010110 013700 001342  
6840 010114 013701 001346  
6841 010120 004737 013126  
6842 010124 010137 001122  
6843 010130 005037 001124

JSR PC,CLRCSR ;CLEAR ALL CSRS  
MOV #2,R4 ;COUNTER FOR TWO GROUPS  
MOV DRCSA,R0 ;CSRA ADDRESS  
MOV DRCRB,R1 ;CSR B ADDRESS  
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
111\$: MOV R1,\$BDADR ;STORE ADDRESS  
CLR \$GDDAT ;EXPECTED DATA

```

6844 010134 112710 000250      MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
6845 010140 112710 000120      MOVB    #SIRR,(R0)      ;SET IRR BITS
6846 010144 112710 000060      MOVB    #SIMR,(R0)      ;SET IMR BITS
6847 010150 112710 000020      MOVB    #CIRMR,(R0)     ;CLEAR IRR+IMR COMMAND
6848 010154 111115              MOVB    (R1),(R5)       ;READ DATA PORT FOR IRR
6849 010156 023715 001124      CMP     $GDDAT,(R5)
6850 010162 001401              BEQ     1$
6851 010164 104003              ERROR   3               ;ERROR ,IRR SHOULD BE CLEARED
6852 010166 112710 000244      1$:    MOVB    #MIMR,(R0)     ;LOAD MODE BITS TO READ IMR
6853 010172 111115              MOVB    (R1),(R5)       ;READ IMR REG
6854 010174 023715 001124      CMP     $GDDAT,(R5)
6855 010200 001401              BEQ     2$
6856 010202 104005              ERROR   5               ;IRR+IMR COMMAND DID NOT CLEAR IMR
6857 010204 005304      2$:    DEC     R4           ;FINISHED BOTH GROUPS?
6858 010206 001405              BEQ     TST41           ;:BR IF BOTH GROUPS TESTED
6859 010210 013700 001352      MOV     DRCSC,R0        ;SETUP FOR GROUP 2
6860 010214 013701 001356      MOV     DRCSD,R1
6861 010220 000741              BR      111$           ;DO IRR+IMR TEST WITH GROUP 2
6862
6863      ;:*****
6864      ;*TEST 41      TEST GROUP 1,2 CLEAR SINGLE IRR+IMR BIT INSTR.
6865      ;:*****
6866      (3)
6867      (3)
6868      (2) 010222 000004      TST41: SCOPE
6869      (2)
6870 010224 004737 013056      JSR     PC,CLRCSR      ;CLEAR ALL CSRS
6871 010230 012704 000002      MOV     #2,R4          ;GROUP COUNTER
6872 010234 012737 010272      MOV     #1$,$LPERR     ;SCOPE RETURN ADDRESS
6873 010242 013700 001342      MOV     DRCSA,R0       ;CSRA ADDRESS
6874 010246 013701 001346      MOV     DRCSD,R1       ;CSRB ADDRESS
6875 010252 004737 013126      JSR     PC,CLRIRR      ;CLEAR IRR REGS WITH CHIP RESET
6876 010256 010137 001122      111$:  MOV     R1,$BDADR      ;STORE ADDRESS
6877 010262 012703 013330      MOV     #BGCHP4,R3     ;GOOD DATA PATTERN TABLE
6878 010266 012702 000030      MOV     #CSIRMR,R2     ;CLEAR SINGLE IRR+IMR BIT VALUE
6879 010272 111337 001124      1$:    MOVB    (R3),$GDDAT   ;STORE EXPECTED
6880 010276 112710 000250      MOVB    #MIRR,(R0)     ;LOAD MODE BITS TO READ IRR
6881 010302 112710 000120      MOVB    #SIRR,(R0)     ;SET ALL IRR BITS
6882 010306 112710 000060      MOVB    #SIMR,(R0)     ;SET ALL IMR BITS
6883 010312 110210              MOVB    R2,(R0)        ;CLEAR SINGLE IRR+IMR BIT
6884 010314 111115              MOVB    (R1),(R5)       ;READ DATA PORT FOR IRR
6885 010316 023715 001124      CMP     $GDDAT,(R5)
6886 010322 001401              BEQ     2$
6887 010324 104003              ERROR   3               ;IRR REG ERROR
6888 010326 112710 000244      2$:    MOVB    #MIMR,(R0)     ;SET UP TO READ IMR
6889 010332 111115              MOVB    (R1),(R5)       ;READ IMR REGISTER
6890 010334 023715 001124      CMP     $GDDAT,(R5)
6891 010340 001401              BEQ     3$
6892 010342 104005              ERROR   5               ;IMR REG ERROR
6893 010344 005723      3$:    TST     (R3)+         ;INC EXPECTED DATA TABLE
6894 010346 020327 013350      CMP     R3,#EDCHP4     ;CHECK FOR END
6895 010352 001402              BEQ     4$
6896 010354 005202              INC     R2
6897 010356 000745              BR      1$             ;SET UP TO CLEAR NEXT IRR+IMR BIT
6898 010360 005304      4$:    DEC     R4           ;CLEAR NEXT IRR+IMR BIT
6899 010362 001405              BEQ     TST42           ;DO GROUPS 1 AND 2
6900 010364 013700 001352      MOV     DRCSC,R0       ;:BR IF BOTH GROUPS TESTED
6901 010370 013701 001356      MOV     DRCSD,R1       ;:SRC ADDRESS
6902                          ;:SRD ADDRESS

```

6896 010374 000730  
6897  
6898  
(3)  
(3)  
(2) 010376 000004  
(2)  
6899 010400 004737 013056  
6900  
6901  
6902  
6903  
6904 010404 012704 000002  
6905 010410 004737 013126  
6906 010414 010337 001122  
6907 010420 112710 000300  
6908 010424 112711 000377  
6909 010430 112710 000120  
6910 010434 112710 000040  
6911 010440 112712 000254  
6912 010444 005037 001124  
6913 010450 111315  
6914 010452 023715 001124  
6915 010456 001401  
6916 010460 104004  
6917 010462 112712 000244  
6918 010466 012737 000377 001124  
6919 010474 111315  
6920 010476 023715 001124  
6921 010502 001401  
6922 010504 104005  
6923 010506 112712 000240  
6924 010512 005037 001124  
6925 010516 111315  
6926 010520 023715 001124  
6927 010524 001401  
6928 010526 104006  
6929 010530 112712 000250  
6930 010534 111315  
6931 010536 023715 001124  
6932 010542 001401  
6933 010544 104003  
6934 010546 005304  
6935 010550 001415  
6936 010552 004737 013056  
6937 010556 013700 001352  
6938 010562 013701 001356  
6939 010566 013702 001342  
6940 010572 013703 001346  
6941 010576 004737 013126  
6942 010602 000704  
6943  
6944  
(3)  
(3)  
(2) 010604 000004

BR 111\$ ;DO GROUP 2  
:\*\*\*\*\*  
:\*TEST 42 TEST GROUPS 1,2 FOR GROUP UNIQUENESS  
:\*\*\*\*\*  
TST42: SCOPE  
JSR PC,CLRCSR ;CLEAR ALL CSRS  
;CSRA = R0  
;CSRB = R1  
;CSRC = R2  
;CSRD = R3  
;COUNTER FOR TESTING TWO GROUPS  
111\$: JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
MOV R3,\$BDADR ;STORE ADDRESS  
MOVB #PACR,(R0) ;PRESELECT ACR FOR WRITING  
MOVB #377,(R1) ;WRITE INTO DATA PORT FOR ACR  
MOVB #SIRR,(R0) ;SET IRR TO ALL 1'S  
MOVB #CIMR,(R0) ;CLEAR IMR  
MOVB #MACR,(R2) ;LOAD MODE TO READ ACR  
CLR \$GDDAT ;EXPECTED  
MOVB (R3),(R5) ;STORE DATA FOR COMPARE  
CMP \$GDDAT,(R5) ;CHECK ACR RESULTS  
BEQ 1\$ ;  
ERROR 4 ;ERROR,OTHER GROUP ACR SHOULD BE CLEARED  
1\$: MOVB #MIMR,(R2) ;CHANGE TO IMR REGISTER  
MOV #377,\$GDDAT ;STORE EXPECTED  
MOVB (R3),(R5) ;READ IMR  
CMP \$GDDAT,(R5) ;SHOULD BE ALL 1'S  
BEQ 2\$ ;  
ERROR 5 ;ERROR,OTHER GROUP IMR SHOULD BE SET  
2\$: MOVB #MISR,(R2) ;LOAD MODE BITS FOR ISR  
CLR \$GDDAT ;STORE EXPECTED  
MOVB (R3),(R5) ;READ ISR  
CMP \$GDDAT,(R5) ;ISR SHOULD BE CLEARED  
BEQ 3\$ ;  
ERROR 6 ;ISR REG ERROR  
3\$: MOVB #MIRR,(R2) ;LOAD MODE BITS FOR IRR  
MOVB (R3),(R5) ;READ IRR  
CMP \$GDDAT,(R5) ;IRR SHOULD BE CLEARED  
BEQ 4\$ ;  
ERROR 3 ;ERROR,OTHER GROUP IRR SHOULD BE CLEARED  
4\$: DEC R4 ;TEST GROUPS 1 AND 2  
BEQ TST43 ;;BR IF BOTH GROUPS TESTED  
JSR PC,CLRCSR ;CLEAR ALL CSRS  
MOV DRCS,R0 ;GROUP 2 CONTROL PORT  
MOV DRCS,R1 ;GROUP 2 DATA PORT  
MOV DRCSA,R2 ;GROUP 1 CONTROL PORT  
MOV DRCSB,R3 ;GROUP 1 DATA PORT  
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET  
BR 111\$ ;TEST GROUP 2 ACR UNIQUENESS  
:\*\*\*\*\*  
:\*TEST 43 TEST STATUS BITS GINT,S2,S1,S0,GP1,2  
:\*\*\*\*\*  
TST43: SCOPE

```
(2)
6945 010606 004737 013056 JSR PC,CLRCR :CLEAR ALL CSRS
6946 010612 013700 001342 MOV DRCSA,R0
6947 010616 013701 001346 MOV DRCSB,R1
6948 010622 012703 000002 MOV #2,R3 :DO TWO GROUPS
6949 010626 012704 000120 111$: MOV #120,R4 :EXPECTED STATUS BITS
6950 010632 005077 170504 CLR @DRCSA :INIT CSRA
6951 010636 005077 170510 CLR @DRCSB :INIT CSRC
6952 010642 012702 013166 MOV #BGPAT1,R2 :EXPECTED IRR PATTERN
6953 010646 112760 000001 000001 MOVB #BIT0,1(R0) :CSR TO OUTPUT MODE
6954 010654 112777 000204 170460 MOVB #204,@DRCSA :POLLED MODE FOR CSRA,GROUP 1
6955 010662 112777 000204 170462 MOVB #204,@DRCSB :POLLED MODE FOR CSRC,GROUP 2
6956 010670 112710 000020 MOVB #CIRMR,(R0) :CLEAR IMR + IRR
6957 010674 012737 000070 001372 MOV #SSIMR,IMRLOC :STORE CODE FOR SINGLE IMR
6958 010702 012737 000130 001376 MOV #SSIRR,IRRLOC :STORE CODE FOR SINGLE IRR
6959 010710 010037 001122 112$: MOV R0,$BDADR :CSR CHIP COMMAND ADDRESS
6960 010714 010437 001124 MOV R4,$GDDAT :EXPECTED DATA
6961 010720 113710 001376 MOVB IRRLOC,(R0) :SET SINGLE IRR BIT
6962 010724 111015 MOVB (R0),(R5) :CHECK STATUS
6963 010726 042715 000007 BIC #7,(R5) :: *** VDRCA1 ADDS THIS *** ::GPA
6964 010732 023715 001124 CMP $GDDAT,(R5)
6965 010736 001401 BEQ 1$
6966 010740 104007 ERROR 7 :CHIP STATUS ERROR
6967 010742 005037 001124 1$: CLR $GDDAT
6968 010746 010137 001122 MOV R1,$BDADR :CSR CHIP DATA ADDRESS
6969 010752 111237 001124 MOVB (R2),$GDDAT
6970 010756 112710 000250 MOVB #MIRR,(R0) :READ IRR
6971 010762 111115 MOVB (R1),(R5)
6972 010764 023715 001124 CMP $GDDAT,(R5) :CHECK IRR
6973 010770 001401 BEQ 2$
6974 010772 104003 ERROR 3 :IRR ERROR
6975 010774 010037 001122 2$: MOV R0,$BDADR
6976 011000 113710 001372 MOVB IMRLOC,(R0) :SET IMR BIT
6977 011004 012737 000320 001124 MOV #320,$GDDAT :CSR EXPECTED DATA
6978 011012 111015 MOVB (R0),(R5)
6979 011014 042715 000007 BIC #7,(R5) :CLEAR UNDEFINED BITS
6980 011020 023715 001124 CMP $GDDAT,(R5)
6981 011024 001401 BEQ 3$
6982 011026 104007 ERROR 7 :CHIP STATUS ERROR
6983 011030 010137 001122 3$: MOV R1,$BDADR
6984 011034 011237 001124 MOV (R2),$GDDAT :EXPECTED DATA
6985 011040 112710 000244 MOVB #MIMR,(R0) :READ IMR BITS
6986 011044 111115 MOVB (R1),(R5) :SAVE IMR READ
6987 011046 023715 001124 CMP $GDDAT,(R5)
6988 011052 001401 BEQ 4$
6989 011054 104005 ERROR 5 :IMR ERROR
6990 011056 005722 4$: TST (R2)+ :NEXT EXPECTED FOR IMR + IRR
6991 011060 020227 013206 CMP R2,#EDCHP1 :CHECK FOR END
6992 011064 001407 BEQ 5$
6993 011066 005237 001376 INC IRRLOC :NEXT IRR BIT
6994 011072 005237 001372 INC IMRLOC :NEXT IMR BIT
6995 :;GPA INC R4 :INDEX EXPECTED STATUS
6996 011076 000240 6$: NOP :; *** VDRCA1 DELETES INC R4 *** ;;GPA
6997 011100 000137 010710 JMP 112$ :DO NEXT STATUS CHECK
6998 011104 005303 5$: DEC R3 :FINISHED BOTH GROUPS?
6999 011106 001406 BEQ TST44 :;BR IF EQUAL
```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25 K 3 PAGE 58-17  
T43 TEST STATUS BITS GINT,S2,S1,S0,GP1,2

SEQ 0036

```
7000 011110 013700 001352      MOV      DRCSC,R0
7001 011114 013701 001356      MOV      DRCSD,R1
7002 011120 000137 010626      JMP      111$          ;DO NEXT GROUP
7003
7004
7005      ;*****
(3)      ;*TEST 44      TEST POLLED MODE;CSRS A,B=OUT C,D=IN,ACTIVE LOW
(3)      ;*****
(2) 011124 000004      TST44: SCOPE
(2)
7006 011126 004737 013056      JSR      PC,LLRCSR    ;CLEAR ALL CSRS
7007      ;R0 = CSRA-GROUP 1 CONTROL
7008      ;R1 = CSRB-GROUP 1 DATA
7009      ;R2 = CSRC-GROUP 2 CONTROL
7010      ;R3 = CSRD-GROUP 2 DATA
7011 011132 012737 011174 001110      MOV      #1$,$LPERR  ;SET FOR SCOPE RETURN
7012 011140 012704 013164      MOV      #BEGPAT,R4  ;START OF PATTERN TABLE
7013 011144 112760 000001 000001      MOVB     #BIT0,1(R0) ;SET CSRA TO OUTPUT MODE
7014 011152 112761 000001 000001      MOVB     #BIT0,1(R1) ;SET CSRB TO OUTPUT MODE
7015
7016 011160 105010      CLR      (R0)        ;CHIP RESET GROUP 1 CSRA
7017 011162 105012      CLR      (R2)        ;CHIP RESET GROUP 2 CSRC
7018 011164 112710 000204      MOVB     #204,(R0)   ;LOAD MODE BITS FOR POLLED MODE,GR 1
7019 011170 112712 000204      MOVB     #204,(R2)   ;LOAD MODE BITS FOR POLLED MODE,GR2
7020 011174 011460 000002      1$:     MOV      (R4),2(R0) ;SET PATTERN DBRA FROM H TO L
7021 011200 112710 000020      MOVB     #CIRMR,(R0) ;CLEAR IMR+IRR GROUP 1
7022 011204 112712 000020      MOVB     #CIRMR,(R2) ;CLEAR IMR+IRR GROUP 2
7023 011210 005060 000002      CLR      2(R0)       ;CLEAR DBRA,ACTIVE LOW WILL SET RPLY C
7024 011214 010137 001122      2$:     MOV      R1,$BDADR  ; GROUP 1 DATA PORT
7025 011220 112710 000250      MOVB     #MIRR,(R0)  ;LOAD BITS TO READ IRR
7026 011224 111437 001124      MOVB     (R4),$GDDAT
7027 011230 111115      MOVB     (R1),(R5)   ;READ IRR,GROUP 1
7028 011232 023715 001124      MOVB     (R1),(R5)
7029 011236 001401      CMP      $GDDAT,(R5)
7030 011240 104003      BEQ      3$
7031 011242 010337 001122      3$:     MOV      R3,$BDADR  ;IRR ERROR,GROUP 1
7032 011246 112712 000250      MOVB     #MIRR,(R2)  ;GROUP 2 DATA PORT
7033 011252 116437 000001 001124      MOVB     1(R4),$GDDAT ;LOAD MODE BITS TO READ IRR GROUP2
7034 011260 042737 000360 001124      BIC      #360,$GDDAT ;BUILD EXPECTED DATA
7035 011266 052737 000100 001124      BIS      #100,$GDDAT ;SAVE BITS 0-3
7036 011274 111315      MOVB     (R3),(R5)   ;EXPECTED 0-3,URPLY 6(C)
7037 011276 023715 001124      MOVB     (R3),(R5)   ;READ IRR BITS,GROUP 2
7038 011302 001401      CMP      $GDDAT,(R5)
7039 011304 104003      BEQ      4$
7040 011306 005762 000002      4$:     ERROR 3          ;IRR ERROR,GROUP 2
7041 011312 052737 000020 001124      TST      2(R2)       ;READ DBRC FOR RPLY 4(A)
7042 011320 111315      BIS      #20,$GDDAT  ;STORE EXPECTED
7043 011322 023715 001124      MOVB     (R3),(R5)   ;READ IRR BITS,GROUP 2
7044 011326 001401      CMP      $GDDAT,(R5)
7045 011330 104003      BEQ      5$
7046 011332 005061 000002      5$:     ERROR 3          ;IRR ERROR,GROUP 2
7047 011336 052737 000200 001124      CLR      2(R1)       ;CLEAR DBRB FOR RPY 7(D)
7048 011344 111315      BIS      #200,$GDDAT ;EXPECTED
7049 011346 023715 001124      MOVB     (R3),(R5)   ;READ IRR BITS GROUP 2
7050 011352 001401      CMP      $GDDAT,(R5)
7051 011354 104003      BEQ      6$
7051      ERROR 3          ;IRR ERROR,GROUP 2
```

```

7052 011356 005763 000002      6$:  TST      2(R3)          ;READ DBRD FOR RPLY 5(B)
7053 011362 052737 000040 001124  BIS      #40,$GDDAT
7054 011370 111315          MOV      (R3),(R5)      ;READ IRR BITS GROUP2
7055 011372 023715 001124  CMP      $GDDAT,(R5)
7056 011376 001401          BEQ      7$             ;GET NEXT PATTERN
7057 011400 104003          ERROR    3             ;IRR ERROR,GROUP 2
7058 011402 005724      7$:  TST      (R4)+         ;INDEX DATA TABLE
7059 011404 020427 013464  CMP      R4,#ENDDAT    ;CHECK FOR END
7060 011410 001271          BNE      1$           ;DO NEXT PATTERN
7061
7062
7063      ;*****
      ;*TEST 45      TEST GROUPS 1,2 IN POLLED MODE,NO REPLY
      ;*****
      ;TST45: SCOPE
7064 011412 000004          JSR      PC,CLRCSR     ;CLEAR ALL CSRS
7065          ;R0 = CSRA-GROUP 1 CONTROL
7066          ;R1 = CSRB-GROUP 1 DATA
7067          ;R2 = CSRC-GROUP 2 CONTROL
7068          ;R3 = CSRD-GROUP 2 DATA
7069 011420 012704 000002  MOV      #2,R4        ;TWO PASSES
7070          ;FIRST PASS CSRA,CSRB = OUTPUT
7071          ;CSRC,CSRD = INPUT
7072          ;SEC PASS CSRC,CSRD = OUTPUT
7073          ;CSRA,CSRB = INPUT
7074 011424 112760 000001 000001 111$: MOV      #BIT0,1(R0)   ;SET CSR TO OUTPUT MODE
7075 011432 112761 000001 000001  MOV      #BIT0,1(R1)   ;SET CSR TO OUTPUT MODE
7076 011440 012760 177777 000002  MOV      #-1,2(R0)    ;SET ALL ONES DBR FOR H TO L
7077 011446 105010          CLR      (R0)         ;CHIP RESET GROUP CSR
7078 011450 105012          CLR      (R2)         ;CHIP RESET GROUP CSR
7079 011452 112710 000204  MOV      #204,(R0)    ;LOAD MODE BITS FOR POLLED MODE
7080 011456 112712 000204  MOV      #204,(R2)    ;LOAD MODE BITS FOR POLLED MODE
7081 011462 112710 000020  MOV      #CIRMR,(R0)  ;CLEAR IMR+IRR
7082 011466 112712 000020  MOV      #CIRMR,(R2)  ;CLEAR IMR+IRR
7083 011472 005760 000002  TST      2(R0)        ;READ DBR IN OUTPUT MODE,NO REPLY
7084 011476 012737 000320 001124  MOV      #320,$GDDAT  ;STORE EXPECTED
7085 011504 010037 001122  MOV      R0,$BDADR    ;STORE ADDRESS
7086 011510 111015          MOV      (R0),(R5)    ;READ STATUS
7087 011512 042715 000007  BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
7088 011516 023715 001124  CMP      $GDDAT,(R5)  ;CHECK STATUS
7089 011522 001401          BEQ      1$           ;CHIP STATUS ERROR
7090 011524 104007          ERROR    7
7091 011526 010237 001122      1$:  MOV      R2,$BDADR    ;STORE ADDRESS
7092 011532 111215          MOV      (R2),(R5)    ;READ STATUS
7093 011534 042715 000007  BIC      #7,(R5)      ;CLEAR UNDEFINED BITS
7094 011540 023715 001124  CMP      $GDDAT,(R5)
7095 011544 001401          BEQ      2$           ;CHIP STATUS ERROR
7096 011546 104007          ERROR    7
7097 011550 012762 000000 000002 2$:  MOV      #0,2(R2)     ;WRITE ONLY,DBR INPUT MODE FOR NO REPLY
7098 011556 005761 000002  TST      2(R1)        ;READ DBR OUTPUT MODE,NO REPLY
7099 011562 012763 000000 000002  MOV      #0,2(R3)     ;WRITE ONLY,DBR INPUT MODE,NO REPLY
7100 011570 010137 001122  MOV      R1,$BDADR    ;CHIP DATA PORT
7101 011574 112710 000250  MOV      #MIRR,(R0)   ;LOAD BITS TO READ IRR
7102 011600 005037 001124  CLR      $GDDAT      ;IRR SHOULD BE ZERO
7103 011604 111115          MOV      (R1),(R5)   ;READ IRR

```

```

7104 011606 023715 001124      CMP      $GDDAT,(R5)
7105 011612 001412              BEQ      3$
7106 011614 005737 017214      TST      KXTFLAG          ; ON KXT11, READ-BEFORE-WRITE...:GPA
7107 011620 001406              BEQ      100$             ;...AT 2$ UPSETS EXPECTED IRR. :GPA
7108 011622 012737 000300 001124      MOV      #300,$GDDAT      ; TRY THE ALTERNATE VALUE.   :GPA
7109 011630 023715 001124      CMP      $GDDAT,(R5)    :GPA
7110 011634 001401              BEQ      3$              :GPA
7111 011636 104003              ERROR    3                ;IRR ERROR                    :GPA
7112 011640 005037 001124      CLR      $GDDAT          :GPA
7113 011644 010337 001122      MOV      R3,$BDADR       ;CHIP DATA PORT             :GPA
7114 011650 112712 000250      MOVB    #MIRR,(R2)       ;LOAD MODE BITS TO READ IRR
7115 011654 111315      MOVB    (R3),(R5)        ;READ IRR BITS
7116 011656 023715 001124      CMP      $GDDAT,(R5)
7117 011662 001412              BEQ      4$
7118 011664 005737 017214      TST      KXTFLAG          ; ON KXT11, DITTO.           :GPA
7119 011670 001406              BEQ      101$            :GPA
7120 011672 012737 000060 001124      MOV      #60,$GDDAT      ; TRY THE ALTERNATE VALUE.   :GPA
7121 011700 023715 001124      CMP      $GDDAT,(R5)    :GPA
7122 011704 001401              BEQ      4$              :GPA
7123 011706 104003              ERROR    3                ;IRR ERROR                    :GPA
7124 011710 005304      4$: DEC      R4            ;FINISHED TWO PASSES
7125 011712 001413              BEQ      TST46           ;:BR IF EQUAL
7126 011714 004737 013056      JSR      PC,CLRCSR       ;CLEAR ALL CSRS
7127 011720 013700 001352      MOV      DRCSC,R0        ;SET UP FOR NEXT PASS
7128 011724 013701 001356      MOV      DRCSD,R1
7129 011730 013702 001342      MOV      DRCSA,R2
7130 011734 013703 001346      MOV      DRCSB,R3
7131 011740 000631      BR       111$            ;DO NEXT PASS
7132
7133      ;:*****
(3)      ;:-TEST 46      TEST POLLED MODE;CSRS C,D=OUT A,B=IN,ACTIVE LOW
(3)      ;:*****
(2) 011742 000004      TST46: SCOPE
(2)
7134 011744 004737 013056      JSR      PC,CLRCSR       ;CLEAR ALL CSRS
7135      ;R0 = CSRA-GROUP 1 CONTROL
7136      ;R1 = CSRB-GROUP 1 DATA
7137      ;R2 = CSRC-GROUP 2 CONTROL
7138      ;R3 = CSRD-GROUP 2 DATA
7139 011750 012737 012012 001110      MOV      #1$,$LPERR      ;SCOPE LOOP RETURN
7140 011756 012704 013164      MOV      #BEGPAT,R4      ;PATTERN TABLE
7141 011762 112762 000001 000001      MOVB    #BIT0,1(R2)      ;SET CSRC TO OUTPUT MODE
7142 011770 112763 000001 000001      MOVB    #BIT0,1(R3)      ;SET CSRD TO OUTPUT MODE
  
```

```

7144 011776 105010          CLRB      (R0)          ;CHIP RESET GROUP 1 CSRA
7145 012000 105012          CLRB      (R2)          ;CHIP RESET GROUP 2 CSRC
7146 012002 112710 000204  MOVB      #204,(R0)     ;LOAD MODE BITS FOR POLLED MODE,GR 1
7147 012006 112712 000204  MOVB      #204,(R2)     ;LOAD MODE BITS FOR POLLED MODE,GR2
7148 012012 011462 000002  1$:      MOV       (R4),2(R2)   ;SET PATTERN DBRC FOR H TO L
7149 012016 112710 000020  MOVB      #CIRMR,(R0)   ;CLEAR IMR+IRR GROUP 1
7150 012022 112712 000020  MOVB      #CIRMR,(R2)   ;CLEAR IMR+IRR GROUP 2
7151 012026 005062 000002  CLR       2(R2)         ;CLEAR DBRC,ACTIVE LOW WILL SET RPLY A
7152 012032 010137 001122  2$:      MOV       R1,$BDADR   ; GROUP 1 DATA PORT
7153 012036 112710 000250  MOVB      #MIRR,(R0)    ;LOAD BITS TO READ IRR
7154 012042 111437 001124  MOVB      (R4),$GDDAT
7155 012046 111115          MOVB      (R1),(R5)     ;READ IRR,GROUP 1
7156 012050 023715 001124  CMP       $GDDAT,(R5)
7157 012054 001401          BEQ       3$
7158 012056 104003          ERROR    3             ;IRR ERROR,GROUP 1
7159 012060 010337 001122  3$:      MOV       R3,$BDADR   ;GROUP 2 DATA PORT
7160 012064 112712 000250  MOVB      #MIRR,(R2)    ;LOAD MODE BITS TO READ IRR GROUP2
7161 012070 116437 000001 001124  MOVB      1(R4),$GDDAT
7162 012076 042737 000360 001124  BIC       #360,$GDDAT   ;SAVE BITS 0-3
7163 012104 052737 000020 001124  BIS       #20,$GDDAT    ;EXPECTED 0-3,URPLY 4(A)
7164 012112 111315          MOVB      (R3),(R5)     ;READ IRR BITS,GROUP 2
7165 012114 023715 001124  CMP       $GDDAT,(R5)
7166 012120 001401          BEQ       4$
7167 012122 104003          ERROR    3             ;IRR ERROR,GROUP 2
7168 012124 005760 000002  4$:      TST       2(R0)         ;READ DBRA FOR RPLY 6(C)
7169 012130 052737 000100 001124  BIS       #100,$GDDAT   ;STORE EXPECTED
7170 012136 111315          MOVB      (R3),(R5)     ;READ IRR BITS,GROUP 2
7171 012140 023715 001124  CMP       $GDDAT,(R5)
7172 012144 001401          BEQ       5$
7173 012146 104003          ERROR    3             ;IRR ERROR,GROUP 2
7174 012150 005063 000002  5$:      CLR       2(R3)         ;CLEAR DBRD FOR RPY 5(B)
7175 012154 052737 000040 001124  BIS       #40,$GDDAT    ;EXPECTED
7176 012162 111315          MOVB      (R3),(R5)     ;READ IRR BITS GROUP 2
7177 012164 023715 001124  CMP       $GDDAT,(R5)
7178 012170 001401          BEQ       6$
7179 012172 104003          ERROR    3             ;IRR ERROR,GROUP 2
7180 012174 005761 000002  6$:      TST       2(R1)         ;READ DBRB FOR RPLY 7(D)
7181 012200 052737 000200 001124  BIS       #200,$GDDAT   ;READ IRR BITS GROUP2
7182 012206 111315          MOVB      (R3),(R5)
7183 012210 023715 001124  CMP       $GDDAT,(R5)
7184 012214 001401          BEQ       7$
7185 012216 104003          ERROR    3             ;IRR ERROR,GROUP 2
7186 012220 005724  7$:      TST       (R4)+        ;INDEX DATA TABLE
7187 012222 020427 013464  CMP       R4,#ENDDAT    ;CHECK FOR END
7188 012226 001271          BNE      1$           ;DO NEXT PATTERN

```

```

7189
7190
7191
7192  (3)  ;*****
7193  (3)  ;*TEST 47 TEST IRR BITS WITH DATA PAT.,POLLED MODE,ACT. HIGH
7194  (2)  ;*****
7195  (2)  ;TST47: SCOPE
7193 012232 004737 013056  JSR      PC,CLRCR      ;CLEAR ALL CSRS
7194 012236 012703 013164  MOV      #BEGPAT,R3    ;GET DATA PATTERN TABLE
7195 012242 012737 000001 001110  MOV      #1,$LPERR     ;SET UP SCOPE ADDRESS

```



```

7196 012250 013700 001342      MOV      DRCSA,R0
7197 012254 013701 001346      MOV      DRCSB,R1
7198 012260 013702 001356      MOV      DRCSB,R1
7199 012264 112760 000001 000001      MOVB    #BIT0,1(R0)      ;CSRA OUTPUT MODE
7200 012272 112761 000001 000001      MOVB    #BIT0,1(R1)      ;CSRB OUTPUT MODE
7201 012300 112710 000224      MOVB    #224,(R0)        ;LOAD MODE BITS FOR POLLED MODE,GP1
7202 012304 112760 000224 000010      MOVB    #224,10(R0)      ;LOAD MODE BITS FOR POLLED MODE,GP2
7203 012312 005060 000002      CLR      2(R0)           ;CLEAR DBRA
7204 012316 112710 000020      MOVB    #CIRMR,(R0)      ;CLEAR IMR +IRR GROUP 1
7205 012322 112760 000020 000010      MOVB    #CIRMR,10(R0)    ;CLEAR IMR + IRR GROUP 2
7206 012330 011360 000002      MOV      (R3),2(R0)      ;STORE PATTERN INTO DBRA
7207 012334 010137 001122      MOV      R1,$BDADR      ;CSRB ADDRESS
7208 012340 112710 000250      MOVB    #MIRR,(R0)      ;LOAD BITS TO READ IRR1
7209 012344 111337 001124      MOVB    (R3),$GDDAT
7210 012350 111115      MOVB    (R1),(R5)      ;READ IRR,GP1
7211 012352 023715 001124      CMP      $GDDAT,(R5)
7212 012356 001401      BEQ      2$
7213 012360 104003      ERROR    3              ;IRR ERROR,GP1
7214 012362 010237 001122      MOV      R2,$BDADR      ;CSRD ADDRESS
7215 012366 112760 000250 000010      MOVB    #MIRR,10(R0)    ;SET MODE TO READ IRR BITS GROUP 2
7216 012374 116337 000001 001124      MOVB    1(R3),$GDDAT    ;BUILD EXPECTED DATA
7217 012402 042737 000360 001124      BIC     #360,$GDDAT     ;BITS 0-3 EXPECTED
7218 012410 052737 000100 001124      BIS     #100,$GDDAT     ;'A' REPLY EXPECTED
7219 012416 111215      MOVB    (R2),(R5)      ;READ IRR2
7220 012420 023715 001124      CMP      $GDDAT,(R5)
7221 012424 001401      BEQ      3$
7222 012426 104003      ERROR    3              ;IRR GROUP 2
7223 012430 005723      TST     (R3)+          ;INDEX DATA TABLE
7224 012432 020327 013464      CMP     R3,#ENDDAT     ;CHECK FOR PATTERN END
7225 012436 001325      BNE     1$            ;DO NEXT PATTERN
7226
7227
7228
(3)
(3)
(2) 012440 000004
(2)
(1) 012442 012737 000001 001160      MOV      #1,$TIMES      ;:DO 1 ITERATION
7229 012450 004737 013056      JSR     PC,CLRCSR      ;:CLEAR ALL CSRS
7230 012454 013700 001342      MOV      DRCSA,R0
7231 012460 013701 001346      MOV      DRCSB,R1
7232 012464 112760 001001 000001      MOVB    #IE!BIT0,1(R0)  ;:CSRA TO OUTPUT MODE
7233 012472 112761 000001 000001      MOVB    #BIT0,1(R1)    ;:SET CSRB TO OUTPUT MODE
7234 012500 010037 001122      MOV      R0,$BDADR     ;:STORE CSRA ADDRESS
7235 012504 012737 100300 001124      MOV      #100300,$GDDAT ;:RDY + CHIP STATUS EXP'D
7236 012512 000005      RESET
7237 012514 011015      MOV      (R0),(R5)     ;:INITIALIZE
7238 012516 042715 000007      BIC     #7,(R5)        ;:STORE REC'D
7239 012522 023715 001124      CMP     $GDDAT,(R5)   ;:CLEAR UNDEFINED BITS
7240 012526 001401      BEQ     1$            ;:MAKE COMPARE
7241 012530 104002      ERROR    2              ;:CSRA ERROR ON RESET
7242 012532 012737 100000 001124 1$:      MOV      #100000,$GDDAT ;:STORE EXPECTED
7243 012540 010137 001122      MOV      R1,$BDADR     ;:CHECK CSRB
7244 012544 011115      MOV      (R1),(R5)     ;:STORE IN $BDDAT
7245 012546 023715 001124      CMP     $GDDAT,(R5)   ;:MAKE COMPARE
7246 012552 001401      BEQ     TST51         ;:BR IF EQUAL
  
```

```

*****
;*TEST 50      TEST CSRA AND CSRB WITH RESET
*****
TST50: SCOPE
  
```

7247 012554 104002 ERROR 2 ;CSRB ERROR WITH RESET

7248  
7249  
7250

(3)  
(3)  
(2) 012556 000004

\*\*\*\*\*  
:TEST 51 TEST CSRC AND CSRD WITH RESET  
\*\*\*\*\*  
TST51: SCOPE

(1) 012560 012737 000001 001160 MOV #1,\$TIMES ;;DO 1 ITERATION

7251 012566 004737 013056 JSR PC,CLRCSR

7252 012572 013702 001352 MOV DRCSC,R2

7253 012576 013703 001356 MOV DRCSD,R3

7254 012602 112762 000001 000001 MOVB #BIT0,1(R2) ;SET CSRC TO OUTPUT MODE

7255 012610 112763 000001 000001 MOVB #BIT0,1(R3) ;SET CSRD TO OUTPUT MODE

7256 012616 010237 001122 MOV R2,\$BDADR ;STORE ADDRESS

7257 012622 012737 100200 001124 MOV #100200,\$GDDAT ;RDY + CHIP STATUS

7258 012630 000005 RESET ;INITIALIZE

7259 012632 011215 MOV (R2),(R5) ;STORE CSRC

7260 012634 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS

7261 012640 023715 00:124 CMP \$GDDAT,(R5) ;MAKE COMPARE

7262 012644 001401 BEQ 1\$

7263 012646 104002 ERROR 2 ;CSRC ERROR WITH RESET

7264 012650 010337 001122 1\$: MOV R3,\$BDADR ;CHECK CSRD

7265 012654 012737 100000 001124 MOV #100000,\$GDDAT ;STORE EXPECTED

7266 012662 011315 MOV (R3),(R5) ;READ CSRD

7267 012664 023715 001124 CMP \$GDDAT,(R5) ;MAKE COMPARE

7268 012670 001401 BEQ 2\$

7269 012672 104002 ERROR 2 ;CSRD ERROR ON RESET

7270 012674 2\$:

7271

```

7273
7274
7275
7276          ;DON'T REPORT 'EOP' UNTIL ALL SELECTED DRV11-J'S HAVE BEEN TESTED.
7277 012674 013701 001342 NXDEV: MOV DRCSA,R1          ;INIT TO SETUP NEXT DRV11J ADDRESSES
7278 012700 000241 NXDEV1: CLC                   ;CLEAR CARRY FOR DEVICE MAP
7279 012702 006037 001364 ROR DMAP           ;LOOK FOR NEXT DRV11J
7280 012706 001412 BEQ $EOP          ;BR IF ALL TESTED
7281 012710 162701 000020 SUB #20,R1       ;NEXT DRV11-J STARTS -20
7282 012714 005237 001202 INC $UNIT        ;UPDATE UNIT NUMBER
7283 012720 032737 000001 001364 BIT #1,DMAP       ;IS UNIT SELECTED?
7284 012726 001764 BEQ NXDEV1          ;NEXT,IF NOT SELECTED
7285 012730 000137 002032 JMP NEXPAS      ;TEST NEXT DRV11-J
7286          .SBTTL END OF PASS ROUTINE

(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 012734
(2) 012734 000240
(1) 012736 005037 001102
(1) 012742 005037 001160
(1) 012746 005237 001176
(1) 012752 042737 100000 001176
(1) 012760 005327
(1) 012762 000001
(1) 012764 003022
(1) 012766 012737
(1) 012770 000001
(1) 012772 012762
(1) 012774 104401 013041
(2) 013000 013746 001176
(2) 013004 104405
(1) 013006 104401 013036
(1) 013012 013700 000042
(1) 013016 001405
(1) 013020 000005
(1) 013022 004710
(1) 013024 000240
(1) 013026 000240
(1) 013030 000240
(1) 013032
(1) 013032 000137
(1) 013034 001766

(1)
(1)
(1) 013036 377 377 000
(1) 013041 015 042412 042116
(1) 013046 050040 051501 020123
(1) 013054 000043
  
```

7288  
7289  
7290  
7291  
7292  
7293  
7294  
7295  
7296 013056 012705 001126  
7297 013062 013700 001342  
7298 013066 013701 001346  
7299 013072 013702 001352  
7300 013076 013703 001356  
7301 013102 005037 001124  
7302 013106 005015  
7303 013110 005010  
7304 013112 105061 000001  
7305 013116 005012  
7306 013120 105063 000001  
7307 013124 000207  
7308  
7309  
7310  
7311 013126 010046  
7312 013130 013700 001342  
7313 013134 105060 000011  
7314 013140 112760 000001 000001  
7315 013146 005060 000002  
7316 013152 105010  
7317 013154 105060 000010  
7318 013160 012600  
7319 013162 000207  
7320

.SBTTL PROGRAM SUBROUTINES

\*\*\*\*\*  
:CLEAR ALL CONTROL/STATUS REGISTERS  
:INIT REGISTERS R0-R4 WITH CSRA-CSR4  
:AND STORE \$BDDAT INTO R5.  
\*\*\*\*\*

CLRCR: MOV # \$BDDAT,R5 ;BAD DATA STORAGE IN R5  
MOV DRCSA,R0 ;CSRA ADDRESS TO R0  
MOV DRCSB,R1 ;CSRB ADDRESS TO R1  
MOV DRCSA,R2 ;CSRC ADDRESS TO R2  
MOV DRCSA,R3 ;CSR4 ADDRESS TO R3  
CLR \$GDAT ;CLEAR EXPECTED  
CLR (R5) ;CLEAR REC'D  
CLR (R0) ;CLEAR CSRA;CHIP RESET GROUP 1  
CLRB 1(R1) ;CLEAR HIGH BYTE CSRB  
CLR (R2) ;CLEAR CSRC;CHIP RESET GROUP 2  
CLRB 1(R3) ;CLEAR HIGH BYTE CSR4  
RTS PC

:CLEAR IRR REGISTERS, GROUP 1, GROUP 2 WITH CHIP RESET

CLRIRR: MOV R0,-(SP)  
MOV DRCSA,R0 ;START OF CSR ADDRESS  
CLRB 1(R0) ;CSRC TO INPUT MODE  
MOV #BIT0,1(R0) ;CSRA TO OUTPUT MODE  
CLR 2(R0) ;CLEAR DBRA  
CLRB (R0) ;CHIP RESET OF GROUP1  
CLRB 10(R0) ;CHIP RESET OF GROUP 2  
MOV (SP)+,R0 ;RESTORE REGISTER  
RTS PC ;EXIT

```
7322  
7323      .SBTTL PATTERNS FOR REGISTER R/W  
7324      ;  
7325      ;PATTERNS USED FOR LOADING/READING REGISTERS  
7326  
7327 013164 000000      BEGPAT: 0      ;GROWING 1  
7328 013166 000001      BGPAT1: 1  
7329 013170 000003      3  
7330 013172 000007      7  
7331 013174 000017      17  
7332 013176 000037      37  
7333 013200 000077      77  
7334 013202 000177      177  
7335 013204 000377      377  
7336 013206 000777      EDCHP1: 777  
7337 013210 001777      1777  
7338 013212 003777      3777  
7339 013214 007777      7777  
7340 013216 017777      17777  
7341 013220 037777      37777  
7342 013222 077777      77777  
7343 013224 177777      177777  
7344 013226 177776      BGCHP2: 177776      ;GROWING 0  
7345 013230 177774      177774  
7346 013232 177770      177770  
7347 013234 177760      177760  
7348 013236 177740      177740  
7349 013240 177700      177700  
7350 013242 177600      177600  
7351 013244 177400      EDCHP2: 177400  
7352 013246 177000      177000  
7353 013250 176000      176000  
7354 013252 174000      174000  
7355 013254 170000      170000  
7356 013256 160000      160000  
7357 013260 140000      140000  
7358 013262 100000      100000  
7359  
7360 013264 000000      BGCHP3: 000000  
7361 013266 000001      1      ;WALKING 1  
7362 013270 000002      2  
7363 013272 000004      4  
7364 013274 000010      10  
7365 013276 000020      20  
7366 013300 000040      40  
7367 013302 000100      100  
7368 013304 000200      200  
7369 013306 000400      EDCHP3: 400  
7370 013310 001000      1000  
7371 013312 002000      2000  
7372 013314 004000      4000  
7373 013316 010000      10000  
7374 013320 020000      20000  
7375 013322 040000      40000  
7376 013324 100000      100000  
7377 013326 177777      177777      ;WALKING 0
```

CNDRCA DRV11J DIAG TST PRT1  
CNDRCA.P11 10-DEC-82 14:37

MACY11 30(1046) 15-DEC-82 15:25  
PATTERNS FOR REGISTER R/W

G 4  
PAGE 61-1

SEQ 0045

7378	013330	177776	BGCHP4:	177776
7379	013332	177775		177775
7380	013334	177773		177773
7381	013336	177767		177767
7382	013340	177757		177757
7383	013342	177737		177737
7384	013344	177677		177677
7385	013346	177577		177577
7386	013350	177377	EDCHP4:	177377
7387	013352	176777		176777
7388	013354	175777		175777
7389	013356	173777		173777
7390	013360	167777		167777
7391	013362	157777		157777
7392	013364	137777		137777
7393	013366	077777		077777
7394	013370	177777		177777
7395	013372	000000	ENDPAT:	000000
7396				
7397				
7398	013374	155555	:DATA PATTERNS	
7399	013376	133333	PATDAT:	155555
7400	013400	066666		133333
7401	013402	125252		066666
7402	013404	052525		125252
7403	013406	177777		052525
7404	013410	000000		177777
7405	013412	107070		000000
7406	013414	070707		107070
7407	013416	144444		070707
7408	013420	033333		144444
7409	013422	011111		033333
7410	013424	022222		011111
7411	013426	044444		022222
7412	013430	111111		044444
7413	013432	166666		111111
7414	013434	010421		166666
7415	013436	021042		010421
7416	013440	031463		021042
7417	013442	042104		031463
7418	013444	063146		042104
7419	013446	073567		063146
7420	013450	104210		073567
7421	013452	114631		104210
7422	013454	135673		114631
7423	013456	146314		135673
7424	013460	156735		146314
7425	013462	167356		156735
7426	013464	000000	ENDDAT:	167356
7427				000000



```
(1) 013642 004737 013700 JSR PC,$TYPEC ;;GO TYPE A NULL
(1) 013646 105337 013744 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
(1) 013652 000770 BR 7$ ;;LOOP
(1)
(1) ;HORIZONTAL TAB PROCESSOR
(1) 013654 112716 000040 8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
(1) 013660 004737 013700 9$: JSR PC,$TYPEC ;;TYPE A SPACE
(1) 013664 132737 000007 013744 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
(1) 013672 001372 BNE 9$ ;;TAB STOP
(1) 013674 005726 TST (SP)+ ;;POP SPACE OFF STACK
(1) 013676 000724 BR 2$ ;;GET NEXT CHARACTER
(1) 013700 105777 165244 $TYPEC: TSTB @STPS ;;WAIT UNTIL PRINTER IS READY
(1) 013704 100375 BPL $TYPEC
(1) 013706 116677 000002 165236 MOVB 2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)
(1) 013714 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
(1) 013722 001003 BNE 1$ ;;BRANCH IF NO
(1) 013724 105037 013744 CLRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
(1) 013730 000406 BR $TYPEX ;;EXIT
(1) 013732 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
(1) 013740 001402 BEQ $TYPEX ;;BRANCH IF YES
(1) 013742 105227 INCB (PC)+ ;;COUNT THE CHARACTER
(1) 013744 000000 $CHARCNT: WORD 0 ;;CHARACTER COUNT STORAGE
(1) 013746 000207 $TYPEX: RTS PC
```

7432

.SBTTL APT COMMUNICATIONS ROUTINE

```
(1)
(1)
(2) ;*****
(1) 013750 112737 000001 014214 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
(1) 013756 112737 000001 014212 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
(1) 013764 000403 BR $ATYC
(1) 013766 112737 000001 014214 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
(1) 013774 $ATYC:
(3) 013774 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 013776 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(1) 014000 105737 014212 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
(1) 014004 001450 BEQ 5$ ;;IF NOT: BR
(1) 014006 122737 000001 001210 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
(1) 014014 001031 BNE 3$ ;;IF NOT: BR
(1) 014016 132737 000100 001211 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 014024 001425 BEQ 3$ ;;IF NOT: BR
(1) 014026 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
(1) 014032 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 014040 005737 001170 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
(1) 014044 001375 BNE 1$ ;;IF NOT: WAIT
(1) 014046 010037 001204 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
(1)
(1) 014052 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 014054 001376 BNE 2$
(1) 014056 163700 001204 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 014062 006200 ASR R0 ;;GET MESSAGE LENGTH IN WORDS
(1) 014064 010037 001206 MOV R0,$MSGLEN ;;PUT LENGTH IN MAILBOX
(1) 014070 012737 000004 001170 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
```



```

(1) 014076 000413
(1) 014100 017637 000004 014124 3$: BR 5$
(1) 014106 062766 000002 000004 MOV @4(SP),4$ ::PUT MSG ADDR IN JSR LINKAGE
(3) 014114 013746 177776 ACB #2,4(SP) ::BUMP RETURN ADDRESS
(1) 014120 004737 013466 MOV 177776,-(SP) ::PUSH 177776 ON STACK
(1) 014124 000000 4$: JSR PC,$TYPE ::CALL TYPE MACRO
(1) 014126 105737 014214 5$: .WORD 0
(1) 014132 001416 10$: TSTB $FFLG ::SHOULD REPORT FATAL ERROR?
(1) 014134 005737 001210 BEQ 12$ ::IF NOT: BR
(1) 014140 001413 TST $ENV ::RUNNING UNDER APT?
(1) 014142 005737 001170 11$: BEQ 12$ ::IF NOT: BR
(1) 014146 001375 TST $MSGTYPE ::FINISHED LAST MESSAGE?
(1) 014150 017637 000004 001172 BNE 11$ ::IF NOT: WAIT
(1) 014156 062766 000002 000004 MOV @4(SP),$FATAL ::GET ERROR #
(1) 014164 005237 001170 ADD #2,4(SP) ::BUMP RETURN ADDR.
(1) 014170 105037 014214 12$: INC $MSGTYPE ::TELL APT TO TAKE ERROR
(1) 014174 105037 014213 CLRB $FFLG ::CLEAR FATAL FLAG
(1) 014200 105037 014212 CLRB $LFLG ::CLEAR LOG FLAG
(3) 014204 012601 MOV (SP)+,R1 ::POP STACK INTO R1
(3) 014206 012600 MOV (SP)+,R0 ::POP STACK INTO R0
(1) 014210 000207 RTS PC ::RETURN
(1) 014212 000 $MFLG: .BYTE 0 ::MESSG. FLAG
(1) 014213 000 $LFLG: .BYTE 0
(1) 014214 000 $FFLG: .BYTE 0 ::LOG FLAG
(1) 014216 000 .EVEN ::FATAL FLAG
(1) 000200 APTSIZE=200
(1) 000001 APTENV=001
(1) 000100 APTSPool=100
(1) 000040 APTCSUP=040

```

7433

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

(1) *****
(2) *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1) *OCTAL (ASCII) NUMBER AND TYPE IT.
(1) *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1) *CALL:
(1) *      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
(1) *      TYPOS    ::CALL FOR TYPEOUT
(1) *      .BYTE   N                ::N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1) *      .BYTE   M                ::M=1 OR 0
(1) *                                           ::1=TYPE LEADING ZEROS
(1) *                                           ::0=SUPPRESS LEADING ZEROS
(1) *
(1) *$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1) *$TYPOS OR $TYPOC
(1) *CALL:
(1) *      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
(1) *      TYPON    ::CALL FOR TYPEOUT
(1) *
(1) *$TYPOC----ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1) *CALL:
(1) *      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
(1) *      TYPOC    ::CALL FOR TYPEOUT
(1)

```

```

(1) 014216 017646 000000
(1) 014222 116637 000001 014441 $TYPOS: MOV @ (SP),-(SP) ;;PICKUP THE MODE
(1) 014230 112637 014443 MOV 1(SP),$OFILL ;;LOAD ZERO FILL SWITCH
(1) 014234 062716 000002 MOV (SP)+,$SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
(1) 014240 000406 ADD #2,(SP) ;;ADJUST RETURN ADDRESS
(1) 014242 112737 000001 014441 $TYPOC: BR $TYPON
(1) 014250 112737 000006 014443 MOV #1,$OFILL ;;SET THE ZERO FILL SWITCH
(1) 014256 112737 000005 014440 $TYPON: MOV #6,$SOMODE+1 ;;SET FOR SIX(6) DIGITS
(1) 014264 010346 MOV #5,$OCNT ;;SET THE ITERATION COUNT
(1) 014266 010446 MOV R3,-(SP) ;;SAVE R3
(1) 014270 010546 MOV R4,-(SP) ;;SAVE R4
(1) 014272 113704 014443 MOV R5,-(SP) ;;SAVE R5
(1) 014276 005404 MOV $SOMODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
(1) 014300 062704 000006 NEG R4
(1) 014304 110437 014442 ADD #6,R4 ;;SUBTRACT IT FOR MAX. ALLOWED
(1) 014310 113704 014441 MOV R4,$SOMODE ;;SAVE IT FOR USE
(1) 014314 016605 000012 MOV $OFILL,R4 ;;GET THE ZERO FILL SWITCH
(1) 014320 005003 MOV 12(SP),R5 ;;PICKUP THE INPUT NUMBER
(1) 014322 006105 1$: CLR R3 ;;CLEAR THE OUTPUT WORD
(1) 014324 000404 BR 3$ ;;GO DO MSB
(1) 014326 006105 2$: ROL R5 ;;ROTATE MSB INTO 'C'
(1) 014330 006105 ROL R5 ;;FORM THIS DIGIT
(1) 014332 006105 ROL R5
(1) 014334 010503 MOV R5,R3
(1) 014336 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
(1) 014340 105337 014442 DECB $SOMODE ;;TYPE THIS DIGIT?
(1) 014344 100016 BPL 7$ ;;BR IF NO
(1) 014346 042703 177770 BIC #177770,R3 ;;GET RID OF JUNK
(1) 014352 001002 BNE 4$ ;;TEST FOR 0
(1) 014354 005704 TST R4 ;;SUPPRESS THIS 0?
(1) 014356 001403 BEQ 5$ ;;BR IF YES
(1) 014360 005204 4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
(1) 014362 052703 000060 BIS #'0,R3 ;;MAKE THIS DIGIT ASCII
(1) 014366 052703 000040 5$: BIS #' ,R3 ;;MAKE ASCII IF NOT ALREADY
(1) 014372 110337 014436 MOV R3,8$ ;;SAVE FOR TYPING
(1) 014376 104401 014436 TYPE ,8$ ;;GO TYPE THIS DIGIT
(1) 014402 105337 014440 7$: DECB $OCNT ;;COUNT BY 1
(1) 014406 003347 BGT 2$ ;;BR IF MORE TO DO
(1) 014410 002402 BLT 6$ ;;BR IF DONE
(1) 014412 005204 INC R4 ;;INSURE LAST DIGIT ISN'T A BLANK
(1) 014414 000744 BR 2$ ;;GO DO THE LAST DIGIT
(1) 014416 012605 6$: MOV (SP)+,R5 ;;RESTORE R5
(1) 014420 012604 MOV (SP)+,R4 ;;RESTORE R4
(1) 014422 012603 MOV (SP)+,R3 ;;RESTORE R3
(1) 014424 016666 000002 000004 MOV 2(SP),4(SP) ;;SET THE STACK FOR RETURNING
(1) 014432 012616 MOV (SP)+,(SP)
(1) 014434 000002 RTI ;;RETURN
(1) 014436 000 8$: .BYTE 0 ;;STORAGE FOR ASCII DIGIT
(1) 014437 000 .BYTE 0 ;;TERMINATOR FOR TYPE ROUTINE
(1) 014440 000 $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
(1) 014441 000 $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
(1) 014442 000000 $SOMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
7434 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

\*\*\*\*\*  
 ;\*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT

```

(1) ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT DEPENDING ON WHETHER THE
(1) ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1) ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1) ;*REPLACED WITH SPACES.
(1) ;*CALL:
(1) ;*   MOV   NUM,-(SP)   ;;PUT THE BINARY NUMBER ON THE STACK
(1) ;*   TYPDS   ;;GO TO THE ROUTINE
(1)
(1) 014444 STYPDS:
(3) 014444 010046 MOV   R0,-(SP)   ;;PUSH R0 ON STACK
(3) 014446 010146 MOV   R1,-(SP)   ;;PUSH R1 ON STACK
(3) 014450 010246 MOV   R2,-(SP)   ;;PUSH R2 ON STACK
(3) 014452 010346 MOV   R3,-(SP)   ;;PUSH R3 ON STACK
(3) 014454 010546 MOV   R5,-(SP)   ;;PUSH R5 ON STACK
(1) 014456 012746 020200 MOV   #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
(1) 014462 016605 000020 MOV   20(SP),R5   ;;GET THE INPUT NUMBER
(1) 014466 100004 BPL   1$         ;;BR IF INPUT IS POS.
(1) 014470 005405 NEG   R5         ;;MAKE THE BINARY NUMBER POS.
(1) 014472 112766 000055 000001 MOVB  #'-,1(SP)   ;;MAKE THE ASCII NUMBER NEG.
(1) 014500 015000 1$: CLR   R0         ;;ZERO THE CONSTANT'S INDEX
(1) 014502 112703 014660 MOV   #SDBLK,R3   ;;SETUP THE OUTPUT POINTER
(1) 014506 112723 000040 MOVB  #' ,(R3)+   ;;SET THE FIRST CHARACTER TO A BLANK
(1) 014512 005002 2$: CLR   R2         ;;CLEAR THE BCD NUMBER
(1) 014514 016001 014650 MOV   $DTBL(R0),R1 ;;GET THE CONSTANT
(1) 014520 160105 3$: SUB   R1,R5         ;;FORM THIS BCD DIGIT
(1) 014522 002402 BLT   4$         ;;BR IF DONE
(1) 014524 005202 INC   R2         ;;INCREASE THE BCD DIGIT BY 1
(1) 014526 000774 BR    3$
(1) 014530 060105 4$: ADD   R1,R5         ;;ADD BACK THE CONSTANT
(1) 014532 005702 TST   R2         ;;CHECK IF BCD DIGIT=0
(1) 014534 001002 BNE   5$         ;;FALL THROUGH IF 0
(1) 014536 105716 TSTB  (SP)       ;;STILL DOING LEADING 0'S?
(1) 014540 100407 BMI   7$         ;;BR IF YES
(1) 014542 106316 5$: ASLB  (SP)       ;;MSD?
(1) 014544 103003 BCC   6$         ;;BR IF NO
(1) 014546 116663 000001 177777 MOVB  1(SP),-1(R3) ;;YES--SET THE SIGN
(1) 014554 052702 000060 6$: BIS  #'0,R2     ;;MAKE THE BCD DIGIT ASCII
(1) 014560 052702 000040 7$: BIS  #' ,R2     ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1) 014564 110223 MOVB  R2,(R3)+   ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1) 014566 005720 TST   (R0)+     ;;JUST INCREMENTING
(1) 014570 020027 000010 CMP   R0,#10    ;;CHECK THE TABLE INDEX
(1) 014574 002746 BLT   2$         ;;GO DO THE NEXT DIGIT
(1) 014576 003002 BGT   8$         ;;GO TO EXIT
(1) 014600 010502 MOV   R5,R2     ;;GET THE LSD
(1) 014602 000764 BR    6$
(1) 014604 105726 8$: TSTB  (SP)+   ;;GO CHANGE TO ASCII
(1) 014606 100003 BPL   9$         ;;WAS THE LSD THE FIRST NON-ZERO?
(1) 014610 116663 177777 177776 MOVB  -1(SP),-2(R3) ;;BR IF NO
(1) 014616 105013 9$: CLRB  (R3)     ;;YES--SET THE SIGN FOR TYPING
(3) 014620 012605 MOV   (SP)+,R5   ;;SET THE TERMINATOR
(3) 014622 012603 MOV   (SP)+,R3   ;;POP STACK INTO R5
(3) 014624 012602 MOV   (SP)+,R2   ;;POP STACK INTO R3
(3) 014626 012601 MOV   (SP)+,R1   ;;POP STACK INTO R2
(3) 014630 012600 MOV   (SP)+,R0   ;;POP STACK INTO R1
(1) 014632 104401 014660 TYPE  $SDBLK     ;;NOW TYPE THE NUMBER
(1) 014636 016666 000002 000004 MOV   2(SP),4(SP) ;;ADJUST THE STACK

```

(1) 014644 012616  
 (1) 014646 000002  
 (1) 014650 023420  
 (1) 014652 001750  
 (1) 014654 000144  
 (1) 014656 000012  
 (1) 014660 000004  
 7435  
 (1)  
 (2)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1)  
 (1) 014670  
 (1) 014670 104407  
 (2) 014672 104407  
 (1) 014674 105237 001103  
 (1) 014700 001775  
 (1) 014702 013777 001102 164232  
 (1) 014710 005237 001112  
 (1) 014714 011637 001116  
 (1) 014720 162737 000002 001116  
 (1) 014726 117737 164164 001114  
 (1) 014734 032777 020000 164176  
 (1) 014742 001004  
 (1) 014744 004737 015044  
 (1) 014750 104401 001165  
 (1) 014754  
 (1) 014754 122737 000001 001210  
 (1) 014762 001007  
 (1) 014764 113737 001114 014776  
 (1) 014772 004737 013766  
 (1) 014776 000  
 (1) 014777 000  
 (1) 015000 000777  
 (1) 015002 005777 164132  
 (1) 015006 100002  
 (1) 015010 000000  
 (1) 015012 104407  
 (1) 015014 032777 001000 164116  
 (1) 015022 001402  
 (1) 015024 013716 001110  
 (1) 015030 005737 001162  
 (1) 015034 001402  
 (1) 015036 013716 001162  
 (1) 015042  
 (1) 015042 000002  
 7436  
 7437

```

MOV (SP)+,(SP)
RTI ;:RETURN TO USER
SDTBL: 10000.
      1000.
      100.
      10.
SDBLK: .BLKW 4
.SBTTL ERROR HANDLER ROUTINE

:*****
:THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
:SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
:AND GO TO SWRCK ON ERROR
:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:SW15=1 HALT ON ERROR
:SW13=1 INHIBIT ERROR TYPEOUTS
:SW09=1 LOOP ON ERROR
:CALL
: * ERROR N ;:ERROR=EMT AND N=ERROR ITEM NUMBER

$ERROR:
  CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
  CKSWR ;GO LOOK FOR SWR CHANGE
7$: INCB $ERFLG ;:SET THE ERROR FLAG
    BEQ 7$ ;:DON'T LET THE FLAG GO TO ZERO
    MOV $TSTNM,@DISPLAY ;:DISPLAY TEST NUMBER AND ERROR FLAG
    INC $ERTTL ;:INC THE ERROR COUNT
    MOV (SP),$ERRPC ;:GET ADDRESS OF ERROR INSTRUCTION
    SUB #2,$ERRPC
    MOVB @$ERRPC,$ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
    BIT #BIT13,@SWR ;:SKIP TYPEOUT IF SET
    BNE 20$ ;:SKIP TYPEOUTS
    JSR PC,SWRCK ;:GO TO USER ERROR ROUTINE
    TYPE $.CRLF

20$: CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
     BNE 2$ ;:NO SKIP APT ERROR REPORT
     MOVB $ITEMB,21$ ;:SET ITEM NUMBER AS ERROR NUMBER
     JSR PC,$ATY4 ;:REPORT FATAL ERROR TO APT

21$: .BYTE 0
     .BYTE 0

22$: BR 22$ ;:APT ERROR LOOP
2$: TST @SWR ;:HALT ON ERROR
    BPL 3$ ;:SKIP IF CONTINUE
    HALT ;:HALT ON ERROR!

3$: CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
    BIT #BIT09,@SWR ;:LOOP ON ERROR SWITCH SET?
    BEQ 4$ ;:BR IF NO
    MOV $LPERR,(SP) ;:FUDGE RETURN FOR LOOPING
    TST $ESCAPE ;:CHECK FOR AN ESCAPE ADDRESS
    BEQ 5$ ;:BR IF NONE
    MOV $ESCAPE,(SP) ;:FUDGE RETURN ADDRESS FOR ESCAPE

5$: RTI ;:RETURN
:*****
;GO TYPE ERROR
  
```

7438  
7439  
7440 015044 113737 001102  
7441 015052 004737 015062  
7442 015056 104407  
7443 015060 000207  
7444  
(1)  
(2)  
(1)  
(1)  
(1)  
(1)  
(1) 015062  
(1) 015062 104401 001165  
(1) 015066 010046  
(1) 015070 005000  
(1) 015072 153700 001114  
(1) 015076 001004  
(1)  
(2) 015100 013746 001116  
(2)  
(2) 015104 104402  
(1) 015106 000426  
(1) 015110 005300  
(1) 015112 006300  
(1) 015114 006300  
(1) 015116 006300  
(1) 015120 062700 001252  
(1) 015124 012037 015134  
(1) 015130 001404  
(1) 015132 104401  
(1) 015134 000000  
(1) 015136 104401 001165  
(1) 015142 012037 015152  
(1) 015146 001404  
(1) 015150 104401  
(1) 015152 000000  
(1) 015154 104401 001165  
(1) 015160 011000  
(1) 015162 001004  
(1) 015164 012600  
(1)  
(1) 015166 104401 001165  
(1) 015172 000207  
(1) 015174  
(2) 015174 013046  
(2) 015176 104402  
(1) 015200 005710  
(1) 015202 001770  
(1) 015204 104401 015212  
(1) 015210 000771  
(1) 015212 020040 000  
(1)  
7445  
(1)

```
;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'  
;*****  
SWRCK:  MOV  $TSTNM,TSTNUM ;SET UP TEST # ON ER  
        JSR  PC,$ERRTYP   ;GO TYPE ERROR  
        CKSWR ;GO LOOK FOR SWR CHANGE  
        RTS  PC           ;RETURN TO ERROR HANDLER  
.SBTTL  ERROR MESSAGE TYPEOUT ROUTINE  
;*****  
;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH  
;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),  
;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.  
$ERRTYP:  
        TYPE  , $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'  
        MOV   RO,-(SP)     ;:SAVE RO  
        CLR  RO           ;:PICKUP THE ITEM INDEX  
        BISB @#$ITEMB,RO  
        BNE  1$           ;:IF ITEM NUMBER IS ZERO, JUST  
                        ;:TYPE THE PC OF THE ERROR  
        MOV  $ERRPC,-(SP) ;:SAVE $ERRPC FOR TYPEOUT  
                        ;:ERROR ADDRESS  
        TYPC ;GO TYPE--OCTAL ASCII(ALL DIGITS)  
        BR   6$           ;:GET OUT  
1$:     DEC  RO           ;:ADJUST THE INDEX SO THAT IT WILL  
        ASL RO           ;:WORK FOR THE ERROR TABLE  
        ASL RO  
        ASL RO  
        ADD  #$ERRTB,RO   ;:FORM TABLE POINTER  
        MOV  (RO)+,2$     ;:PICKUP "ERROR MESSAGE" POINTER  
        BEQ  3$           ;:SKIP TYPEOUT IF NO POINTER  
        TYPE ;TYPE THE "ERROR MESSAGE"  
                        ;:"ERROR MESSAGE" POINTER GOES HERE  
2$:     .WORD 0           ;:'CARRIAGE RETURN' & 'LINE FEED'  
        TYPE  , $CRLF      ;:PICKUP "DATA HEADER" POINTER  
3$:     MOV  (RO)+,4$     ;:SKIP TYPEOUT IF 0  
        BEQ  5$           ;:TYPE THE "DATA HEADER"  
        TYPE ;"DATA HEADER" POINTER GOES HERE  
4$:     .WORD 0           ;:'CARRIAGE RETURN' & 'LINE FEED'  
        TYPE  , $CRLF      ;:PICKUP "DATA TABLE" POINTER  
5$:     MOV  (RO),RO     ;:GO TYPE THE DATA  
        BNE  7$           ;:RESTORE RO  
6$:     MOV  (SP)+,RO  
        TYPE  , $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'  
        RTS  PC           ;:RETURN  
7$:     MOV  @(RO)+,-(SP) ;:SAVE @(RO)+ FOR TYPEOUT  
        TYPC ;GO TYPE--OCTAL ASCII(ALL DIGITS)  
        TST  (RO)        ;:IS THERE ANOTHER NUMBER?  
        BEQ  6$         ;:BR IF NO  
        TYPE  , 8$       ;:TYPE TWO(2) SPACES  
        BR   7$         ;:LOOP  
8$:     .ASCIZ  / /      ;:TWO(2) SPACES  
        .EVEN  
.SBTTL  SCOPE HANDLER ROUTINE  
;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
```

```
(1) ;*AND LOAD THE TEST NUMBER($STSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1) ;*AND LOAD THE ERROR FLAG ($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) $SCOPE:
(1) 015216 104407 CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
(1) 015220 032777 040000 163712 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
(1) 015226 001114 BNE $OVER ;:YES IF SW14=1
(1) ;*#####START OF CODE FOR THE XOR TESTER#####
(1) 015230 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
(1) ;:THIS INSTRUCTION TO A 'NOP' (NOP=240)
(1) 015232 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
(1) 015236 012737 015256 000004 MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
(1) 015244 005737 177060 TST @#177060 ;:TIME OUT ON XOR?
(1) 015250 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
(1) 015254 000463 BR $SVLAD ;:GO TO THE NEXT TEST
(1) 015256 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
(1) 015260 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
(1) 015264 000423 BR 7$ ;:LOOP ON THE PRESENT TEST
(1) 015266 6$::#####END OF CODE FOR THE XOR TESTER#####
(1) 015266 032777 000400 163644 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
(1) 015274 001404 BEQ 2$ ;:BR IF NO
(1) 015276 127737 163636 001102 CMPB @SWR,$STSTNM ;:ON THE RIGHT TEST? SWR<7:0>
(1) 015304 001465 BEQ $OVER ;:BR IF YES
(1) 015306 105737 001103 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
(1) 015312 001421 BEQ 3$ ;:BR IF NO
(1) 015314 123737 001115 001103 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
(1) 015322 101015 BHI 3$ ;:BR IF NO
(1) 015324 032777 001000 163606 BIT #BIT09,@SWR ;:LOOP ON ERROR?
(1) 015332 001404 BEQ 4$ ;:BR IF NO
(1) 015334 013737 001110 001106 7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
(1) 015342 000446 BR $OVER
(1) 015344 105037 001103 4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
(1) 015350 005037 001160 CLR $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1) 015354 000415 BR 1$ ;:ESCAPE TO THE NEXT TEST
(1) 015356 032777 004000 163554 3$: BIT #BIT11,@SWR ;:INHIBIT ITERATIONS?
(1) 015364 001011 BNE 1$ ;:BR IF YES
(1) 015366 005737 001176 TST $PASS ;:IF FIRST PASS OF PROGRAM
(1) 015372 001406 BEQ 1$ ;: INHIBIT ITERATIONS
(1) 015374 005237 001104 INC $ICNT ;:INCREMENT ITERATION COUNT
(1) 015400 023737 001160 001104 CMP $TIMES,$ICNT ;:CHECK THE NUMBER OF ITERATIONS MADE
(1) 015406 002024 BGE $OVER ;:BR IF MORE ITERATION REQUIRED
(1) 015410 012737 000001 001104 1$: MOV #1,$ICNT ;:REINITIALIZE THE ITERATION COUNTER
(1) 015416 013737 015474 001160 MOV $MXCNT,$TIMES ;:SET NUMBER OF ITERATIONS TO DO
(1) 015424 105237 001102 $SVLAD: INCB $STSTNM ;:COUNT TEST NUMBERS
(1) 015430 113737 001102 001174 MOVB $STSTNM,$TESTN ;:SET TEST NUMBER IN APT MAILBOX
(1) 015436 011637 001106 MOV (SP),$LPADR ;:SAVE SCOPE LOOP ADDRESS
(1) 015442 011637 001110 MOV (SP),$LPERR ;:SAVE ERROR LOOP ADDRESS
(1) 015446 005037 001162 CLR $ESCAPE ;:CLEAR THE ESCAPE FROM ERROR ADDRESS
(1) 015452 112737 000001 001115 MOVB #1,$ERMAX ;:ONLY ALLOW ONE(1) ERROR ON NEXT TEST
```

(1)	015460	013777	001102	163454	\$OVER:	MOV	\$STNM,@DISPLAY	::DISPLAY TEST NUMBER
(1)	015466	013716	001106			MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS
(1)	015472	000002				RTI		::FIXES PS
(1)	015474	000062			\$MXCNT:	50.		::MAX. NUMBER OF ITERATIONS
7446					.SBTTL		TTY INPUT ROUTINE	
(1)					::*****			
(2)								
(1)								
(1)					.ENABL	LSB		
(1)					::*****			
(2)					::**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)	015476	022737	000176	001140	\$CKSWR:	CMP	#SWREG,SWR	::IS THE SOFT-SWR SELECTED?
(1)	015504	001074				BNE	15\$	::BRANCH IF NO
(1)	015506	105777	163432			TSTB	@\$TKS	::CHAR THERE?
(1)	015512	100071				BPL	15\$	::IF NO, DON'T WAIT AROUND
(1)	015514	117746	163426			MOVB	@\$TKB,-(SP)	::SAVE THE CHAR
(1)	015520	042716	177600			BIC	#^C177,(SP)	::STRIP-OFF THE ASCII
(1)	015524	022726	000007			CMP	#7,(SP)+	::IS IT A CONTROL G?
(1)	015530	001062				BNE	15\$	::NO, RETURN TO USER
(1)	015532	123727	001134	000001		CMPB	\$AUTOB,#1	::ARE WE RUNNING IN AUTO-MODE?
(1)	015540	001456				BEQ	15\$	::BRANCH IF YES
(1)								
(1)	015542	104401	016223		\$GTSWR:	TYPE	,\$CNTLG	::ECHO THE CONTROL-G (^G)
(1)	015546	104401	016230			TYPE	,\$MSWR	::TYPE CURRENT CONTENTS
(2)	015552	013746	000176			MOV	SWREG,-(SP)	::SAVE SWREG FOR TYPEOUT
(2)	015556	104402				TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)	015560	104401	016241			TYPE	,\$MNEW	::PROMPT FOR NEW SWR
(1)	015564	005046			19\$:	CLR	-(SP)	::CLEAR COUNTER
(1)	015566	005046				CLR	-(SP)	
(1)							::THE NEW SWR	
(1)	015570	105777	163350		7\$:	TSTB	@\$TKS	::CHAR THERE?
(1)	015574	100375				BPL	7\$	::IF NOT TRY AGAIN
(1)								
(1)	015576	117746	163344			MOVB	@\$TKB,-(SP)	::PICK UP CHAR
(1)	015602	042716	177600			BIC	#^C177,(SP)	::MAKE IT 7-BIT ASCII
(1)								
(1)								
(1)	015606	021627	000025		9\$:	CMP	(SP),#25	::IS IT A CONTROL-U?
(1)	015612	001005				BNE	10\$	::BRANCH IF NOT
(1)	015614	104401	016216			TYPE	,\$CNTLU	::YES, ECHO CONTROL-U (^U)
(1)	015620	062706	000006		20\$:	ADD	#6,SP	::IGNORE PREVIOUS INPUT
(1)	015624	000757				BR	19\$	::LET'S TRY IT AGAIN
(1)								
(1)								
(1)	015626	021627	000015		10\$:	CMP	(SP),#15	::IS IT A <CR>?
(1)	015632	001022				BNE	16\$	::BRANCH IF NO
(1)	015634	005766	000004			TST	4(SP)	::YES, IS IT THE FIRST CHAR?
(1)	015640	001403				BEQ	11\$	::BRANCH IF YES
(1)	015642	016677	000002	163270		MOV	2(SP),@SWR	::SAVE NEW SWR
(1)	015650	062706	000006		11\$:	ADD	#6,SP	::CLEAR UP STACK

```

(1) 015654 104401 001165 14$: TYPE $CRLF ;;ECHO <CR> AND <LF>
(1) 015660 123727 001135 000001 CMPB $INTAG,#1 ;;RE-ENABLE TTY KBD INTERRUPTS?
(1) 015666 001003 BNE 15$ ;;BRANCH IF NOT
(1) 015670 012777 000100 163246 MOV #100,@$TKS ;;RE-ENABLE TTY KBD INTERRUPTS
(1) 015676 000002 15$: RTI ;;RETURN
(1) 015700 004737 013700 16$: JSR PC,$TYPEC ;;ECHO CHAR
(1) 015704 021627 000060 CMP (SP),#60 ;;CHAR < 0?
(1) 015710 002420 BLT 18$ ;;BRANCH IF YES
(1) 015712 021627 000067 CMP (SP),#67 ;;CHAR > 7?
(1) 015716 003015 BGT 18$ ;;BRANCH IF YES
(1) 015720 042726 000060 BIC #60,(SP)+ ;;STRIP-OFF ASCII
(1) 015724 005766 000002 TST 2(SP) ;;IS THIS THE FIRST CHAR
(1) 015730 001403 BEQ 17$ ;;BRANCH IF YES
(1) 015732 006316 ASL (SP) ;;NO, SHIFT PRESENT
(1) 015734 006316 ASL (SP) ;; CHAR OVER TO MAKE
(1) 015736 006316 ASL (SP) ;; ROOM FOR NEW ONE.
(1) 015740 005266 000002 17$: INC 2(SP) ;;KEEP COUNT OF CHAR
(1) 015744 056616 177776 BIS -2(SP),(SP) ;;SET IN NEW CHAR
(1) 015750 000707 BR 7$ ;;GET THE NEXT ONE
(1) 015752 104401 001164 18$: TYPE $QUES ;;TYPE ?<CR><LF>
(1) 015756 000720 BR 20$ ;;SIMULATE CONTROL-U
(1) .DSABL LSB

*****
*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:
* RDCHR ;;INPUT A SINGLE CHARACTER FROM THE TTY
* RETURN HERE ;;CHARACTER IS ON THE STACK
* ;;WITH PARITY BIT STRIPPED OFF
*

(1) 015760 011646 $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC
(1) 015762 016666 000004 000002 MOV 4(SP),2(SP) ;;SAVE THE PS
(1) 015770 105777 163150 1$: TSTB @$TKS ;;WAIT FOR
(1) 015774 100375 BPL 1$ ;;A CHARACTER
(1) 015776 117766 163144 000004 MOVB @$TKB,4(SP) ;;READ THE TTY
(1) 016004 042766 177600 000004 BIC #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
(1) 016012 026627 000004 000023 CMP 4(SP),#2$ ;;IS IT A CONTROL-S?
(1) 016020 001013 BNE 3$ ;;BRANCH IF NO
(1) 016022 105777 163116 2$: TSTB @$TKS ;;WAIT FOR A CHARACTER
(1) 016026 100375 BPL 2$ ;;LOOP UNTIL ITS THERE
(1) 016030 117746 163112 MOVB @$TKB,-(SP) ;;GET CHARACTER
(1) 016034 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
(1) 016040 022627 000021 CMP (SP)+,#21 ;;IS IT A CONTROL-Q?
(1) 016044 001366 BNE 2$ ;;IF NOT DISCARD IT
(1) 016046 000750 BR 1$ ;;YES, RESUME
(1) 016050 026627 000004 000140 3$: CMP 4(SP),#140 ;;IS IT UPPER CASE?
(1) 016056 002407 BLT 4$ ;;BRANCH IF YES
(1) 016060 026627 000004 000175 CMP 4(SP),#175 ;;IS IT A SPECIAL CHAR?
(1) 016066 003003 BGT 4$ ;;BRANCH IF YES
(1) 016070 042766 000040 000004 BIC #40,4(SP) ;;MAKE IT UPPER CASE
(1) 016076 000002 4$: RTI ;;GO BACK TO USER

*****
*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
*CALL:

```



```
(1)          : *      RDLIN          :: INPUT A STRING FROM THE TTY
(1)          : *      RETURN HERE     :: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)          : *                               :: TERMINATOR WILL BE A BYTE OF ALL 0'S

(1) 016100 010346 $RDLIN: MOV R3,-(SP)      :: SAVE R3
(1) 016102 012703 016206 1$: MOV #$TTYIN,R3  :: GET ADDRESS
(1) 016106 022703 016216 2$: CMP #$TTYIN+8.,R3  :: BUFFER FULL?
(1) 016112 101405 BLOS 4$          :: BR IF YES
(1) 016114 104410 RDCHR          :: GO READ ONE CHARACTER FROM THE TTY
(1) 016116 112613 MOVB (SP)+,(R3)  :: GET CHARACTER
(1) 016120 122713 000177 10$: CMPB #177,(R3)  :: IS IT A RUBOUT
(1) 016124 001003 BNE 3$          :: SKIP IF NOT
(1) 016126 104401 001164 4$: TYPE ,SQUES  :: TYPE A '?'
(1) 016132 000763 BR 1$          :: CLEAR THE BUFFER AND LOOP
(1) 016134 111337 016204 3$: MOVB (R3),9$  :: ECHO THE CHARACTER
(1) 016140 104401 016204 TYPE ,9$
(1) 016144 122723 000015 CMPB #15,(R3)+  :: CHECK FOR RETURN
(1) 016150 001356 BNE 2$          :: LOOP IF NOT RETURN
(1) 016152 105063 177777 CLPB -1(R3)      :: CLEAR RETURN (THE 15)
(1) 016156 104401 001166 TYPE ,SLF      :: TYPE A LINE FEED
(1) 016162 012603 MOV (SP)+,R3    :: RESTORE R3
(1) 016164 011646 MOV (SP),-(SP)  :: ADJUST THE STACK AND PUT ADDRESS OF THE
(1) 016166 016666 000004 000002 MOV 4(SP),2(SP)  :: FIRST ASCII CHARACTER ON IT
(1) 016174 012766 016206 000004 MOV #$TTYIN,4(SP)
(1) 016202 000002 RTI          :: RETURN
(1) 016204 000 9$: .BYTE 0      :: STORAGE FOR ASCII CHAR. TO TYPE
(1) 016205 000 .BYTE 0      :: TERMINATOR
(1) 016206 000010 $TTYIN: .BLK8 8.  :: RESERVE 8 BYTES FOR TTY INPUT
(1) 016216 052536 005015 000 $CNTLU: .ASCIIZ /^L/<15><12>  :: CONTROL 'U'
(1) 016223 136 006507 000012 $CNTLG: .ASCIIZ /^G/<15><12>  :: CONTROL 'G'
(1) 016230 005015 053523 020122 $MSWR: .ASCIIZ <15><12>/SWR = /
(1) 016236 020075 000
(1) 016241 040 047040 053505 $MNEW: .ASCII? / NEW = /
(1) 016246 036440 000040

7447 .SBTTL POWER DOWN AND UP ROUTINES
(1)
(2) :: *****
(1) : POWER DOWN ROUTINE
(1) 016252 012737 016416 000024 $PWRDN: MOV $ILLUP,@#PWRVEC  :: SET FOR FAST UP
(1) 016260 012737 000300 000026 MOV #PR6,@#PWRVEC+2  :: PRIO:6
(3) 016266 010046 MOV R0,-(SP)  :: PUSH R0 ON STACK
(3) 016270 010146 MOV R1,-(SP)  :: PUSH R1 ON STACK
(3) 016272 010246 MOV R2,-(SP)  :: PUSH R2 ON STACK
(3) 016274 010346 MOV R3,-(SP)  :: PUSH R3 ON STACK
(3) 016276 010446 MOV R4,-(SP)  :: PUSH R4 ON STACK
(3) 016300 010546 MOV R5,-(SP)  :: PUSH R5 ON STACK
(3) 016302 017746 162632 MOV @SWR,-(SP)  :: PUSH @SWR ON STACK
(1) 016306 010637 016422 MOV SP,$SAVR6  :: SAVE SP
(1) 016312 012737 016324 000024 MOV #$PWRUP,@#PWRVEC  :: SET UP VECTOR
(1) 016320 000000 HALT
(1) 016322 000776 BR -2  :: HANG UP
(1)
(2) :: *****
(1) : POWER UP ROUTINE
(1) 016324 012737 016416 000024 $PWRUP: MOV $ILLUP,@#PWRVEC  :: SET FOR FAST DOWN
(1) 016332 013706 016422 MOV $SAVR6,SP  :: GET SP
```

```

(1) 016336 005037 016422          CLR    $SAVR6          ;;WAIT LOOP FOR THE TTY
(1) 016342 005237 016422    1$:  INC    $SAVR6          ;;WAIT FOR THE INC
(1) 016346 001375          BNE    1$              ;;CF WORD
(3) 016350 012677 162564    MOV    (SP)+,@SWR      ;;POP STACK INTO @SWR
(3) 016354 012605          MOV    (SP)+,R5        ;;POP STACK INTO R5
(3) 016356 012604          MOV    (SP)+,R4        ;;POP STACK INTO R4
(3) 016360 012603          MOV    (SP)+,R3        ;;POP STACK INTO R3
(3) 016362 012602          MOV    (SP)+,R2        ;;POP STACK INTO R2
(3) 016364 012601          MOV    (SP)+,R1        ;;POP STACK INTO R1
(3) 016366 012600          MOV    (SP)+,R0        ;;POP STACK INTO R0
(1) 016370 012737 016252 000024  MOV    #PWRDN,@PWRVEC  ;;SET UP THE POWER DOWN VECTOR
(1) 016376 012737 000300 000026  MOV    #PR6,@PWRVEC+2  ;;PRIO:6
(1) 016404 104401          TYPE                                ;;REPORT THE POWER FAILURE
(1) 016406 016424          $PWRMG: .WORD    PWRMSG  ;;POWER FAIL MESSAGE POINTER
(1) 016410 012716          MOV    (PC)+,(SP)      ;;RESTART AT START1
(1) 016412 001766          $PWRAD: .WORD    START1  ;;RESTART ADDRESS
(1) 016414 000002          RTI
(1) 016416 000000          $ILLUP: HALT           ;;THE POWER UP SEQUENCE WAS STARTED
(1) 016420 000776          BR     .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1) 016422 000000          $SAVR6: 0              ;;PUT THE SP HERE
7448 016424 005015 042522 052123  PWRMSG: .ASCIZ <15><12>/RESTARTED FROM PWR FAIL/
      016432 051101 042524 020104
      016440 051106 046517 050040
      016446 051127 043040 044501
      016454 000114

7449
7450          .EVEN
          .SBTTL TRAP DECODER

          ;;*****
          ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
          ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
          ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
          ;;*GO TO THAT ROUTINE.

(1) 016456 010046          $TRAP: MOV    R0,-(SP)    ;;SAVE R0
(1) 016460 016600 000002    MOV    2(SP),R0        ;;GET TRAP ADDRESS
(1) 016464 005740          TST    -(R0)           ;;BACKUP BY 2
(1) 016466 111000          MOV    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1) 016470 006300          ASL    R0              ;;POSITION FOR INDEXING
(1) 016472 016000 016512    MOV    $TRPAD(R0),R0   ;;INDEX TO TABLE
(1) 016476 000200          RTS    R0              ;;GO TO ROUTINE

          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

(1) 016500 011646          $TRAP2: MOV   (SP),-(SP)  ;;MOVE THE PC DOWN
(1) 016502 016666 000004 000002  MOV   4(SP),2(SP)      ;;MOVE THE PSW DOWN
(1) 016510 000002          RTI                    ;;RESTORE THE PSW

(3)          .SBTTL TRAP TABLE

(3)          ;;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)          ;;*BY THE "TRAP" INSTRUCTION.

(3)          :          ROUTINE
(3)          :          -----
(3)          :
  
```

(3)	016512	016500			\$TRPAD: .WORD	\$TRAP2		
(3)	016514	015466			\$TYPE	::CALL=TYPE	TRAP+1(104401)	TTY TYPEOUT ROUTINE
(3)	016516	014242			\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)	016520	014216			\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)	016522	014256			\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)	016524	014444			\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
(1)								
(3)	016526	015546			\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
(1)								
(3)	016530	015476			\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
(3)	016532	015760			\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
(3)	016534	016100			\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
7451								
7452								
7453								
7454	016536	005015	047103	051104	.SBTTL ASCII MESSAGES			
	016544	040503	020040	051104	::GPA TITLED: .ASCIZ <15><12>/CNDRCA DRV11J DIAG TEST PART 1 /<15><12>			
	016552	030526	045061	042040	TI LED: .ASCIZ <15><12>/CNDRCA DRV11J DIAG TEST PART 1 /<15><12> ;:GPA			
	016560	040511	020107	042524				
	016566	052123	050040	051101				
	016574	020124	020061	006440				
	016602	000012						
7455	016604	005015	051104	030526	TI CABL:	.ASCIZ	<15><12>/DRV11J CABLE REQ'D/<15><12>	
	016612	045061	041440	041101				
	016620	042514	051040	050505				
	016626	042047	005015	000				
7456								
7457	016633	122	043505	052040	EM1:	.ASCIZ	/REG TIMEOUT ER/	
	016640	046511	047505	052125				
	016646	042440	000122					
7458	016652	042522	020107	042522	EM2:	.ASCIZ	'REG READ/WRITE ER'	
	016660	042101	053457	044522				
	016666	042524	042440	000122				
7459	016674	051111	020122	042522	EM3:	.ASCIZ	/IRR REG ER/	
	016702	020107	051105	000				
7460	016707	101	051103	051040	EM4:	.ASCIZ	/ACR REG ER/	
	016714	043505	042440	000122				
7461	016722	046511	020122	042522	EM5:	.ASCIZ	/IMR REG ER/	
	016730	020107	051105	000				
7462	016735	111	051123	051040	EM6:	.ASCIZ	/ISR REG ER/	
	016742	043505	042440	000122				
7463	016750	044103	050111	051440	EM7:	.ASCIZ	/CHIP STAT ER/	
	016756	040524	020124	051105				
	016764	000						
7464	016765	105	051122	041520	DH1:	.ASCIZ	/ERRPC TSTNUM BUSADR EXPCT RCVD/	
	016772	020040	052040	052123				
	017000	052516	020115	041040				
	017006	051525	042101	020122				
	017014	042440	050130	052103				
	017022	020040	051040	053103				
	017030	000104						
7465	017032	051105	050122	020103	DH2:	.ASCIZ	/ERRPC TSTNUM BUSADR ADRS EXPCT RCVD/	
	017040	020040	051524	047124				
	017046	046525	020040	052502				
	017054	040523	051104	020040				
	017062	042101	051522	020040				

	017070	020040	054105	041520		
	017076	020124	020040	041522		
	017104	042126	000			
7466						
7467		017110				
7468	017110	001116	001362	001122	DT1:	.EVEN
	017116	001124	001126	000000		\$ERRPC,TSTNUM,\$BDADR,\$GDDAT,\$BDDAT,0
7469	017124	001116	001362	001122	DT2:	\$ERRPC,TSTNUM,\$BDADR,\$GDADR,\$GDDAT,\$BDDAT,0
	017132	001120	001124	001126		
	017140	000000				

7558  
7559  
7560 017142  
(1) 000100 000100  
(1) 000102 017142  
(1) 000102 000300  
(1) 000140 000140  
(1) 000142 170000  
(1) 000142 000300  
(1) 017142 017142  
(1) 017142 104401 017150  
(1) 017146 000000  
(1) 017150 005015 045514 042526  
(1) 017156 020103 047111 042524  
(1) 017164 051122 050125 020124  
(1) 017172 020055 044504 041523  
(1) 017200 047117 042516 052103  
(1) 017206 046040 041524 000040  
7561  
7562 017214 000001  
7563 000001

:THE FOLLOWING .INIT CALL WAS ADDED TO INITIALIZE ODT  
:VECTOR AND LKVEC.  
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 /  
  
:THE FOLLOWING FLAG WAS ADDED TO INDICATE IT IS FALCON  
KXTFLAG:1  
.END











SW11	=	004000	5693#			
SW12	=	010000	5693#			
SW13	=	020000	5693#			
SW14	=	040000	5693#			
SW15	=	100000	5693#			
SW2	=	000004	5693#			
SW3	=	000010	5693#			
SW4	=	000020	5693#			
SW5	=	000040	5693#			
SW6	=	000100	5693#			
SW7	=	000200	5693#			
SW8	=	000400	5693#			
SW9	=	001000	5693#			
TBITVE	=	000014	5693#			
TILED	=	016536	5847	7454#		
TKVEC	=	000060	5693#			
TLCABL	=	016604	5848	7455#		
TPVEC	=	000064	5693#			
TRAPVE	=	000034	5693#	5814*		
TRTVEC	=	000014	5693#			
TSTNUM	=	001362	5803#	7440*	7468	7469
TST1	=	002074	5874#			
TST10	=	003260	6063	6066#		
TST11	=	003354	6085#			
TST12	=	003450	6105#			
TST13	=	003544	6125#			
TST14	=	003640	6146#			
TST15	=	004010	6178	6181#		
TST16	=	004156	6212	6216#		
TST17	=	004456	6271#			
TST2	=	002152	5890#			
TST20	=	004744	6323#			
TST21	=	005244	6377#			
TST22	=	005532	6428#			
TST23	=	005666	6453	6458#		
TST24	=	006076	6497	6502#		
TST25	=	006302	6541	6546#		
TST26	=	006506	6584	6590#		
TST27	=	006624	6610	6615#		
TST3	=	002344	5924	5927#		
TST30	=	006742	6635	6640#		
TST31	=	007034	6654	6659#		
TST32	=	007134	6674	6679#		
TST33	=	007262	6702	6708#		
TST34	=	007410	6731	6736#		
TST35	=	007510	6751	6757#		
TST36	=	007612	6773	6778#		
TST37	=	007744	6802	6807#		
TST4	=	002454	5945	5948#		
TST40	=	010076	6831	6836#		
TST41	=	010222	6858	6863#		
TST42	=	010376	6893	6898#		
TST43	=	010604	6935	6944#		
TST44	=	011124	6999	7005#		
TST45	=	011412	7063#			
TST46	=	011742	7125	7133#		











.INIT	5658#	5693#	7560
.SETUP	1213#	5680#	5812
.SWRHI	104#	5681#	5692
.SWRLO	5692#		
.\$ACT1	5064#	5683#	5737
.\$APT8	5109#	5683#	5743#
.\$APTH	5370#	5683#	5742
.\$APTY	5547#	5683#	7432
.\$ASTA	5417#		
.\$CATC	932#	5681#	5732
.\$CMTA	1047#	5681#	5743
.\$DB2D	4686#		
.\$DB2O	4812#		
.\$DIV	4587#		
.\$EOP	2214#	5682#	7286
.\$ERRO	2700#	5682#	7435
.\$ERRT	2896#	5682#	7444
.\$MULT	4523#		
.\$POWE	4229#	5681#	7447
.\$RAND	4307#		
.\$RDDE	3891#		
.\$RDOC	3797#	5680#	
.\$READ	3395#	5682#	7446
.\$R2AZ	4958#		
.\$SAVE	3969#		
.\$SB2D	4771#		
.\$SB2O	4874#		
.\$SCOP	2454#	5682#	7445
.\$SIZE	4361#		
.\$SUPR	4913#		
.\$STRAP	4073#	5680#	7450
.\$TYPB	3287#		
.\$TYPD	3209#	5682#	7434
.\$TYPE	2985#	5682#	7431
.\$TYPO	3112#	5681#	7433
.\$4OCA	972#		

. ABS. 017216 000

ERRORS DETECTED: 0

CNDRCA,CNDRCA/CR/NL:TOC=CNMAC2.SML,CNDRCA.P11  
RUN-TIME: 14 15 1 SECONDS  
RUN-TIME RATIO: 68/30=2.2  
CORE USED: 33K (66 PAGES)