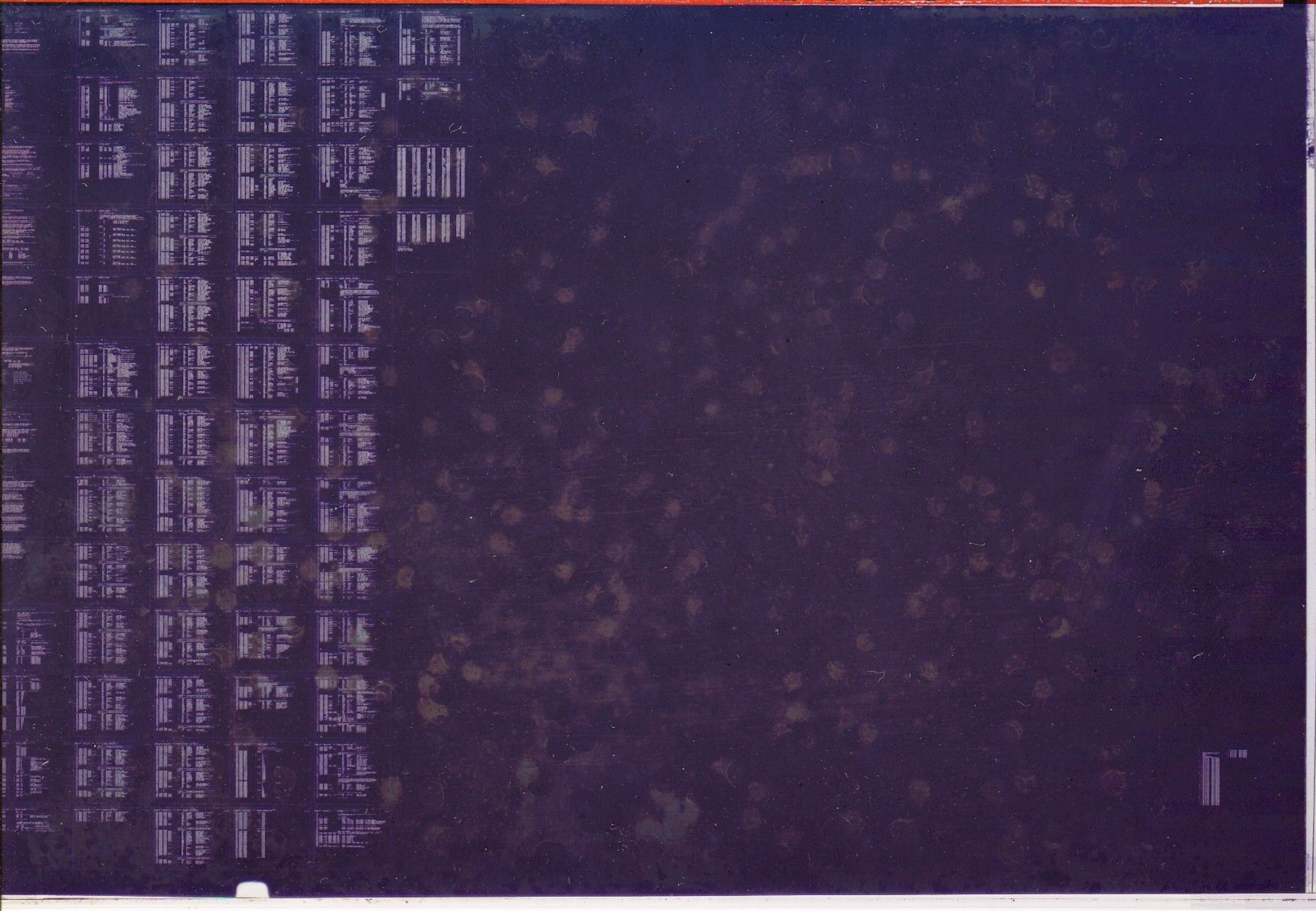
DRV11-J DIAG TST PRT 1 CVDRCBO AH-F758B-MC FICHE 1 OF 1 JUL 1982 COPYRIGHT@ 79-82 MADE IN USA



IDENTIFICATION

PRODUCT CODE: AC-F756B-MC

PRODUCT NAME: CVDRCB0 DRV11J DIAG TST PRT1

DATE CREATED: 17-FFB-82

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THE DRV11-J IS A GENERAL PURPOSE PARALLEL INTERFACE FOR THE LSI-11 BUS. IT HAS A BASIC CONFIGURATION OF 64 TRI-STATE IN/OUT LINES DIVIDED INTO FOUR GROUPS OR PORTS OF 16 BIT WORDS.

THERE ARE TWO 4K DIAGNOSTICS FOR THE DRV11-J OPTION. THE DRV11-J DIAGNOSTIC TEST PART 1 OF 2 CONTAINS A SERIES OF TESTS WITHOUT DRV11J INTERRUPTS DESIGNED TO TEST ALL LOGIC FUNCTIONS AND DATA PATHS MADE ACCESSIBLE WITH THE LOOPBACK CABLE INSERTED INTO THE DPV11-J I/O CONNECTORS.

THE DRV11-J DIAGNOSTIC PART 2 OF 2 IS A SERIES OF TESTS WITH DRV11J INTERRUPTS DESIGNED TO TEST ALL LOGIC AND DATA PATHS MADE ACCESSIBLE WITH THE LOOPBACK CABLE INSERTED INTO THE I/O CONNECTORS.

THE DRV11-J IS CONTAINED ON A DOUBLE HEIGHT MODULE. THE MODULE CONTAINS TWO 50 PIN CONNECTORS FOR INTERFACING TO EXTERNAL USER DEVICES.

FOR DIAGNOSTIC TESTING,THE DPV11-J CABLE,BC05W-02) MUST BE INSTALLED WITH 1/2 TWIST BETWEEN THE 50 PIN CONNECTORS.

```
**************************************************************************
*                                                                ::GPA   *
* NOTE: THIS DIAGNOSTIC HAS BEEN MODIFIED TO RUN IN KXT11 (SBC 11/21)    *
* BASED SYSTEMS.  THE PROGRAM WILL AUTOMATICALLY ADJUST ITSELF TO RUN    *
* IN THE APPROPRIATE ENVIRONMENT AS FOLLOWS:                             *
*                                                                        *
*                LSI-11, 11/2, AND 11/23        SBC 11/21              *
*                ------------------------       ---------              *
* CSR RANGE:         160010 TO 177760         174000 TO 177760         *
* PROGRAMMABLE...                                                        *
* ...VECTOR RANGE:     0000 TO 1774               000 TO 374            *
*                                                                ::GPA   *
**************************************************************************
```

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP11/03,11/23 COMPUTER OR LSI-11 PROCESSOR WITH A MINIMUM OF 4K MEMORY.
2. SERIAL LINE INTERFACE AND CONSOLE TERMINAL
3. DRV11-J OPTION WITH A BC05W-02 CABLE

2.2 STORAGE

THE PROGPAM USES THE LOWER 4K OF MEMORY.

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE

BINARY FORMATTED PAPERTAPES OR XXDP MEDIA
(FILES WITH .BIC OR .BIN EXTENSIONS ONLY).

4.0 STARTING PROCEDURE

1. MAKE SURE THE DRV11-J CABLE IS INSERTED WITH 1/2 A TWIST ON THE I/O CONNECTORS OF THE DRV11-J OPTION. THIS WILL CONNECT PORT A TO PORT C AND PORT B TO PORT D.

2. MAKE SURE THE DEVICE BUS ADDRESSES AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 7.1. IF NOT,CHANGE LOCATION(S) AS DESIRED VIA THE 'ADDRESS/' ODT COMMAND.

3. THE PROGRAM SHOULD ALWAYS BE STARTED AT 200. STARTING AT 200(200G OR .R CVDRCB UNDER XXDP+),THE PROGRAM INITIALIZES ITSELF,PRINTS ITS ID(FIRST TIME ONLY) AND THEN PRINTS THAT THE DRV11-J CABLE IS REQUIRED(FIRST TIME ONLY) AND THEN PRINTS: SWR=XXXXXX NEW=

   WHERE XXXXXX REPRESENTS THE CURRENT VALUE OF THE SOFTWARE SWITCH REGISTER. IF NO CHANGES ARE REQUIRED IN THE SWITCH REGISTER THEN JUST HIT CARRIAGE RETURN.
   IF CHANGES ARE REQUIRED,THEN A NEW VALUE MAY BE TYPED FOLLOWED BY A CARRIAGE RETURN.
   REFER TO SECTION 5.0 FOR SWITCH REGISTER OPTIONS.

4. IF THE FOLLOWING ERROR OCCURS RIGHT AFTER STARTUP IT IS POSSIBLE THAT THE DRV11J LOOPBACK CABLE MAY NOT BE INSTALLED OR WAS NOT INSTALLED PROPERLY:

   REG READ/WRITE ER
   ERRPC TSTNUM BUSADR EXPCT RCVD
   002224 000002 *164160 100700 000700
   *BUSADR MAY BE DIFFERENT DEPENDING ON THE CSR OF THE DRV11J.

5.0 SOFTWARE SWITCH REGISTER

5.1 OPTIONS

THE PROGRAM SWITCH DEFAULT MODE IS SWR = 000000
IF USING A VIDEO TERMINAL ,BIT 15 = 1(HALT ON ERROR)
MAY BE HELPFUL IN KEEPING THE ERROR ON THE SCREEN.

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR <7-0>

5.2 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE CHANGED BY USING THE ODT FACILITIES.

2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM

CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD OPERATION WILL PRINT OUT THE CURRENT CONTENTS AND ACCEPT NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE RETURN.

3. ONCE THE ODT MODE HAS BEEN ENTERED BECAUSE OF AN ERROR CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE SOFTWARE SWITCH REGISTER IF DESIRED BEFORE TYPING 'P' (CONTINUE).

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF.

TO CONSERVE MEMORY SPACE FOR THE 4K PROGRAM MEMORY REQUIREMENT,REGISTER 5 (R5) IS RESERVED FOR THE $BDDAT LOCATION (1126).

EXAMPLE: CMP $GDDAT,(R5) IS THE SAME AS CMP $GDDAT,$BDDAT.

6.2.1 ERROR DATA

```
ERROR TITLE HEADING
ERRPC   TSTNUM  BUSADR  EXPCT   RCVD
XXXXXX  XXXXXX  XXXXXX  XXXXXX  XXXXXX

ERRPC           LISTING ADDRESS WHERE THE ERROR WAS DETECTED
TSTNUM          TEST NUMBER WHERE THE ERROR OCCURRED
BUSADR          DRV11J BUS REG ADDRESS OF CONCERNED OPERATION
EXPCT           DATA THAT WAS EXPECTED
RCVD            DATA THAT WAS RECEIVED
```

6.2.2 ERROR TITLES

```
REG TIMEOUT ER          ;REGISTER TIMEOUT ERROR
REG READ/WRITE ER       ;REGISTER READ/WRITE ERROR
IRR REG ER              ;INTERRUPT REQUEST REGISTER ERROR
ACR REG ER              ;AUTOCLEAR REGISTER ERROR
IMR REG ER              ;INTERRUPT MASK REGISTER ERROR
ISR REC ER              ;INTERRUPT SERVICE REGISTER ERROR
CHIP STAT ER            ;CHIP STATUS ERROR
```

7.0 MISCELLANEOUS

7.1 DRV11-J BUS ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' (ADDR: 1244) IF BASE BUS ADDRESS
IS NOT 164160.

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC DOES SUPPORT ACT AND APT ENVIRONMENTS.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT
WHICH TIME THE PROGRAM IS RESTARTED (ONLY ON SYSTEMS WITH
NON-VOLATILE MEMORY AND WITH APPROPRIATE HARDWARE).

7.4 MULTIPLE DRV11J INTERFACE TESTING

THIS PROGRAM DOES NOT "AUTO-SIZE" THE NUMBER OF DRV11J'S CONNECTED.
THIS DIAGNOSTIC WILL TEST SEQUENTIALLY UP TO 4 DRV11J INTERFACES
WITH CONTIGUOUS BUS ADDRESSES. THIS IS ACCOMPLISHED
BY THE USER SETTING UP LOCATION 'DEVM' (ADDR: 1246)
WITH A BIT MAP INDICATING WHAT INTERFACES ARE TO BE TESTED.
I.E. BIT0 = 1 SAYS TEST 1ST DRV11J,
BIT1 = 1 SAYS TEST 2ND DRV11J
BIT2 = 1 SAYS TEST 3RD DRV11J
BIT3 = 1 SAYS TEST 4TH DRV11J

1ST UNIT = STARTING CSRA	164160	$DEVM = 1
2ND UNIT = STARTING CSRA	164140	$DEVM = 3
3RD UNIT = STARTING CSRA	164120	$DEVM = 7
4TH UNIT = STARTING CSRA	164100	$DEVM = 17

7.5 RESTRICTIONS

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT <5 SECONDS ON FIRST
PASS WITH NO ITERATIONS TO <20 SECONDS WITH ITERATIONS
ENABLED AFTER FIRST PASS,PER DRV11J UNIT CONNECTED.
AN 'END PASS" MESSAGE INDICATES ALL TESTS HAVE COMPLETED
ON ALL SELECTED UNITS.

9.0 PROGRAM TEST DESCRIPTIONS

GENERAL

THIS DIAGNOSTIC CONTAINS A SERIES OF INDEPENDENT TESTS DESIGNED
TO TEST LOGIC FUNCTIONS AND DATA PATHS OF THE DRV11J OPTION.
TESTING IS ACCOMPLISHED WITH THE AID OF THE
DRV11J LOOP BACK CABLE PROVIDED FOR DIAGNOSTIC TESTING.
A COMPLETE LIST OF TESTS IS AVAILABLE IN THE TABLE OF CONTENTS
AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN
EACH TEST CAN BE BENEFICIAL IN TEST UNDERSTANDING.

TESTING WITH THE DRV11J CABLE ALLOWS FOR TESTING
OF PORT A WITH PORT C AND THE TESTING OF PORT B
WITH PORT D.

9.1 TESTS T1-T22 REGISTER CHECKING

THESE TESTS WILL WRITE/READ PORTS A TO C
AND WRITE/READ PORTS B TO D WITH FLOATING
ONES AND FLOATING ZEROS,GROWING ONES AND
GROWING ZEROS AND DATA PATTERNS.
TESTS ARE PERFORMED TO INSURE INTERACTION WITH
CONNECTED PORTS AND NO INTERACTION BETWEEN
UNCONNECTED PORTS.

9.2 T22-T42 INTERRUPT CONTROL REGISTER CHECKING

TESTS ARE MADE TO THE INTERRUPT CONTROL CHIP
REGISTERS IRR,ACR,IMR AND LIMITED TESTING OF
ISR REGISTER UNTIL INTERRUPTS ARE PERFORMED.
TESTS ARE PERFORMED ON THE REGISTERS WITH
FLOATING ONES,FLOATING ZEROS,GROWING ONES
AND GROWING ZEROS. CHIP RESET CAPABILITIES
ARE TESTED AS WELL AS UNIQUENESS OF REGISTERS
WITHIN A CHIP GROUP AND THE UNIQUENESS OF
ONE GROUP'S REGISTERS TO ANOTHER GROUP'S
REGISTERS.

9.3 TEST 43 - TEST STATUS BITS GINT,S2,S1,S0,GP1,GP2

EXERCISE THE STATUS BITS S2,S1,S0 AND GINT FOR
EACH GROUP CHIP BY SETTING SINGLE IRR BITS TO CAUSE THE
STATUS BITS TO GO FROM 120 TO 127. EACH IRR BIT
IS THEN MASKED TO RETURN STATUS BACK TO ORIGINAL
STATUS WITH NO IRR BITS SET.

9.4 T44-T47 POLLED MODE TESTING

THIS TEST WILL WRITE PATTERNS INTO DBRA WITH
EITHER LOW TO HIGH OR HIGH TO LOW POLARITIES.
AFTER PLACING ALL ONES IN DBRA AND THEN SELECTING
ACTIVE LOW,CLEARING DBRA WILL NOW SET IRR BITS
0-7,GROUP 1 AND IRR BITS0-3,GROUP2.
THE RPLY SIGNALS WILL SET IRR BITS 4-7
WHEN WRITING DBRA IN OUTPUT MODE,THIS WILL
SET THE RPLY FOR DBRC.(IRR6,GROUP 2)

WHEN READING DBRC IN INPUT MODE WILL SET
THE RPLY FOR DBRA.(IRR BIT 4,GROUP 2)
WHEN WRITING DBRB IN OUTPUT MODE,THIS WILL
SET THE RPLY FOR DBRD(IRR BIT7,GROUP2)
WHEN READING DBRD IN INPUT MODE.THIS WILL
SET THE RPLY FOR DBRB(IRR BIT5,GROUP 2).
TEST ALL DATA PATTERNS TO SET IRR BITS
GROUP1 AND GROUP2 AND FOR THE SETTING OF RPLY
BITS IN THE GROUP 2 IRR REGISTER BY WRITING IN
OUTPUT MODE AND READING IN INPUT MODE.
TEST THAT NO RPLY BITS SET WHEN READING IN OUTPUT
MODE AND WHEN WRITING IN INPUT MODE.

9.5 T50-T51 CSR'S WITH RESET
SET UP CSR'S IN OUTPUT MODE AND CLEAR
DIRECTION BIT OUT OF EACH CSR EXCEPT FOR
CSRA WHICH WILL CLEAR DIR BIT AND I/E BIT.

10.0 LISTING

```
                                ;;GPA   .HEADER ^/CVDRCA  DRV11J DIAG TST PRT1/,1979,^/BILL HEAVEY/
                                .TITLE  CVDRCB  DRV11J DIAG TST PRT1
                                ;*COPYRIGHT (C) 1979
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY BILL HEAVEY
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
                                ;* EDITED TO PERMIT DRV'S TO RUN ON FALCON (KXT11).              ;;GPA
                                ;*              G. PASQUANTONIO, JULY '81                      ;;GPA
                                ;*
165400                          $SWR=165400
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
```

```
(1)             000006                  SP=     %6              ;;STACK POINTER
(1)             000007                  PC=     %7              ;;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)             0004C0                  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)                                   ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)          000004                   ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)          000010                   RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)          000014                   TBITVEC=14              ;;"T" BIT
(1)          000014                   TRTVEC= 14              ;;TRACE TRAP
(1)          000014                   BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)          000020                   IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)          000024                   PWRVEC= 24              ;;POWER FAIL
(1)          000030                   EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)          000034                   TRAPVEC=34              ;;"TRAP" TRAP
(1)          000060                   TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)          000064                   TPVEC=  64              ;;TTY PRINTER VECTOR
(1)          000240                   PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
 20          164160                   ABASE=  164160          ;BASE ADDRESS
 21          000001                   ADEVM=  1               ;DEFAULT TO ONE DRV11J
 22          100000                   RDY=    BIT15
 23          000400                   DIR=    BIT8
 24          001000                   IE=     BIT9
 25
 26                                   ;CHIP COMMAND SUMMARY
 27          000020                   CIRMR=  20              ;CLEAR IRR AND IMR
 28          000030                   CSIRMR= 30              ;CLEAR SINGLE IRR AND IMR BIT
 29
 30          000040                   CIMR=   40              ;CLEAR IMR
 31          000050                   CSIMR=  50              ;CLEAR SINGLE IMR BIT
 32          000060                   SIMR=   60              ;SET ALL IMR BITS
 33          000070                   SSIMR=  70              ;SET SINGLE IMR BITS
 34
 35          000100                   CIRR=   100             ;CLEAR IRR
 36          000110                   CSIRR=  110             ;CLEAR SINGLE IRR BITS
 37          000120                   SIRR=   120             ;SET ALL IRR BITS
 38          000130                   SSIRR=  130             ;SET SINGLE IRR BITS
 39
 40          000140                   CHPISR= 140             ;CLEAR HIGHEST PRIORITY ISR BIT
 41          000160                   CISR=   160             ;CLEAR ISR
 42          000170                   CSISR=  170             ;CLEAR SINGLE ISR BIT
 43
 44          000200                   LMD04=  200             ;LOAD MODE BITS M0-M4
 45          000240                   LMD57=  240             ;LOAD MODE BITS M5-M7
 46
 47                                   ;CHIP MODE BIT PRESELECTION
 48          000240                   MISR    240
 49          000244                   MIMR=   244
```

```
                        000250                  MIRR=   250
                        000254                  MACR=   254

                                                ;CHIP WRITE PRESELECTION
                        000300                  PACR=   300             ;PRESELECT AUTO CLEAR REG. FOR WRITING
                        000260                  PIMR=   260             ;PRESELECT IMR REG. FOR WRITING
                        000340                  PVMA=   340             ;PRESELECT VECTOR MEMORY ADDRESS

                                                .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)                                             .SBTTL  STARTING ADDRESS(ES)
(1)  000200  000137  001402                             JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                        000100                          .=100
                000100  000104  000200  000002          .WORD   104,200,2       ;IF 'B EVENT' ON Q BUS IS CONNECTED
                                                                                ;IGNORE IT'S INTERRUPT - JUST DO A RTI
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000106                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  013070                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000106                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000
                                                                ;LONGEST TEST TIME
                                                                ;1ST PASS RUN TIME
                                                                ;ADDITIONAL RUN TIME
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
001000                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
001000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001170                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000006                  $TSTM:  .WORD   6.      ;;RUN TIM OF LONGEST TEST
001006  000024                  $PASTM: .WORD   20.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000024                  $UNITM: .WORD   20.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
 69                                     .SBTTL  COMMON TAGS
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)                                     ;*USED IN THE PROGRAM.
(1)
(1)                 001100                      .=1100
(1)  001100                             $CMTAG:                 ;;START OF COMMON TAGS
(1)  001100  000000                             .WORD   0
(1)  001102     000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
(1)  001103     000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
(1)  001104  000000                     $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
(1)  001106  000000                     $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
(1)  001110  000000                     $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
(1)  001112  000000                     $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
(1)  001114     000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
(1)  001115     001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
(1)  001116  000000                     $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)  001120  000000                     $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
(1)  001122  000000                     $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
(1)  001124  000000                     $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
(1)  001126  000000                     $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
(1)  001130  000000                             .WORD   0       ;;RESERVED--NOT TO BE USED
(1)  001132  000000                             .WORD   0
(1)  001134     000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
(1)  001135     000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
(1)  001136  000000                             .WORD   0
(1)  001140  177570                     SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
(1)  001142  177570                     DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
(1)  001144  177560                     $TKS:   177560          ;;TTY KBD STATUS
(1)  001146  177562                     $TKB:   177562          ;;TTY KBD BUFFER
(1)  001150  177564                     $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
(1)  001152  177566                     $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
(1)  001154     000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
(1)  001155     002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)  001156     012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
(1)  001157     000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(1)  001160  000000                     $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
(1)  001162  000000                     $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
(1)  001164     077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
(1)  001165     015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
(1)  001166  000012                     $LF:    .ASCIZ  <12>    ;;LINE FEED
(2)                                     ;;*****************************************************************
(2)                                     .SBTTL  APT MAILBOX-ETABLE
(2)
(3)                                     ;;*****************************************************************
(2)                                     .EVEN
(2)  001170                             $MAIL:                  ;;APT MAILBOX
(2)  001170  000000                     $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
(2)  001172  000000                     $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
(2)  001174  000000                     $TESTN: .WORD   ATESTN  ;;TEST NUMBER
(2)  001176  000000                     $PASS:  .WORD   APASS   ;;PASS COUNT
(2)  001200  000000                     $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
(2)  001202  000000                     $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001204  000000                     $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(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          ;;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  164160                  $BASE:   .WORD   ABASE          ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001246  000001                  $DEVM:   .WORD   ADEVM          ;;DEVICE MAP
(2)  001250  000000                  $CDW1:   .WORD   ACDW1          ;;CONTROLLER DESCRIPTION WORD#1
(2)  001252                          $ETEND:
(2)                                  .MEXIT
```

```
(1)                                     .SBTTL  ERROR POINTER TABLE
(1)
(1)                                     ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                     ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                     ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                     ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                     ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                     ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                     ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                     ;*      DT              ;;POINTS TO THE DATA
(1)                                     ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001252                             $ERRTB:
 70                                     ;ERROR  1
 71  001252  016763                             EM1             ;REG TIMEOUT ER
 72  001254  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 73  001256  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
 74  001260  000000                             0
 75
 76                                     ;ERROR  2
 77  001262  017002                             EM2             ;REG READ/WRITE ER
 78  001264  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 79  001266  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
 80  001270  000000                             0
 81
 82                                     ;ERROR  3
 83  001272  017024                             EM3             ;IRR REG ER
 84  001274  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 85  001276  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
 86  001300  000000                             0
 87
 88                                     ;ERROR  4
 89  001302  017037                             EM4             ;ACR REG ER
 90  001304  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 91  001306  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
 92  001310  000000                             0
 93
 94                                     ;ERROR  5
 95  001312  017052                             EM5             ;IMR REG ERROR
 96  001314  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 97  001316  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
 98  001320  000000                             0
 99
100                                     ;ERROR  6
101  001322  017065                             EM6             ;ISR REG ERROR
102  001324  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
103  001326  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
104  001330  000000                             0
105
106
107                                     ;ERROR  7
108  001332  017100                             EM7             ;CHIP STAT ER
109  001334  017115                             DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
110  001336  017240                             DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
```

```
001340 000000                          0


                                ; BUS REGISTER ADDRESS POINTERS


001342 164160                   DRCSA:  ABASE
001344 164162                   DRDBA:  ABASE+2
001346 164164                   DRCSB:  ABASE+4
001350 164166                   DRDBB:  ABASE+6
001352 164170                   DRCSC:  ABASE+10
001354 164172                   DRDBC:  ABASE+12
001356 164174                   DRCSD:  ABASE+14
001360 164176                   DRDBD:  ABASE+16


                                ;COMMON PROGRAM LOCATION(S)

001362 000000                   TSTNUM: 0                       ;CONTAINS TEST NUMBER ON ERROR
001364 000001                   DMAP:   1
001366 000000                   INTFLG: .WORD   0
001370 000000                   XXDP:   .WORD   0
001372 000000                   IMRLOC: .WORD   0
001374 000000                   ISRLOC: .WORD   0
001376 000000                   IRRLOC: .WORD   0
001400 000000                   ACRLOC: .WORD   0
```

```
                                        .SBTTL  PROGRAM START
001402                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001402  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001406  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001410  022706  001140                  CMP     #SWR,R6 ;;DONE?
001414  001374                          BNE     .-6             ;;LOOP BACK IF NO
001416  012706  001100                  MOV     #1100,SP                ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001422  012737  015336  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001430  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001436  012737  015010  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001444  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001452  012737  016606  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001460  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001466  012737  016402  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001474  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001502  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001506  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001512  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001520  012737  001520  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001526  012737  001526  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001534  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001540  012737  001574  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001546  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001554  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001562  022777  177777  177350          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001570  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001572  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001574  012716  001602          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001600  000002                          RTI
001602  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001610  012737  000174  001142          MOV     #DISPREG,DISPLAY
001616  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001622  005037  001176                  CLR     $PASS           ;;CLEAR PASS COUNT
001626  132737  000200  001211          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
001634  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
001636  012737  001212  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
001644                          67$:
001644  005037  001172                  CLR     $FATAL          ;CLEAR ERROR NUMBER
001650  005037  001170                  CLR     $MSGTYP         ;CLEAR MESSAGE TYPE
001654  005037  001174                  CLR     $TESTN          ;CLEAR TEST NUMBER
001660  004737  017272                  CALL    FALCON          ; CHECK FOR FALCON (KXT11)      ;;GPA
001664  001420                          BEQ     1000$           ; BR IF NOT KXT11 SYSTEM        ;;GPA
001666  042737  000040  000022          BIC     #40,@#22        ; YES, STRAP IOT...             ;;GPA
001674  042737  000040  000032          BIC     #40,@#32        ;...EMT...                      ;;GPA
001702  042737  000040  000036          BIC     #40,@#36        ;...AND TRAP TO LEVEL 6.        ;;GPA
001710  023727  001244  164160          CMP     $BASE,#ABASE    ; IS $BASE VIRGIN ??            ;;GPA
001716  001003                          BNE     1000$           ; BR IF NOT.                    ;;GPA
001720  012737  174160  001244          MOV     #174160,$BASE   ; YES, USE ENGINEERING DEFAULT  ;;GPA
001726                          1000$:                                                          ;;GPA
```

```
                                ;CHECK OPERATING ENVIRONMENT
001726  0C5737  000042          TST     @#42            ;ARE WE IN ACT/XXDP AUTO MODE?
001732  001410                  BEQ     1$              ;BRANCH IF NO
001734  023737  000042  000046  CMP     @#42,@#46       ;IS IT ACT AUTO MODE?
001742  001410                  BEQ     2$              ;BRANCH IF YES
001744  012737  177777  001370  MOV     #-1,XXDP        ;SET XXDP CHAIN MODE INDICATOR
001752  000404                  BR      2$
001754  123727  001210  000001  1$:     CMPB    $ENV,#1         ;ARE WE IN APT AUTO MODE?
001762  001003                  BNE     3$              ;BRANCH IF NO
001764  112737  000001  001134  2$:     MOVB    #1,$AUTOB       ;SET AUTO MODE INDICATOR

                                ;PRINT TITLE IF NOT IN ACT OR APT AUTO MODE
001772  005227  177777          3$:     INC     #-1             ;FIRST TIME?
001776  001012                  BNE     5$              ;SKIP TITLE IF NO
002000  005737  001134          TST     $AUTOB          ;ARE WE IN AUTO MODE?
002004  001403                  BEQ     4$              ;BRANCH TO TITLE TYPEOUT IF NOT
002006  005737  001370          TST     XXDP            ;IS THE AUTO MODE UNDER XXDP?
002012  001404                  BEQ     5$              ;SKIP TITLE IF NOT
002014  104401  016666          4$:     TYPE    ,TITLED         ;PRINT OUT THE TITLE
002020  104401  016774          TYPE    ,TLCABL         ;PRINT 'DRV11J CABLE REQ'D''

                                ;GET THE VALUE IN THE SOFTWARE SWITCH REGISTER
002024  005737  001134          5$:     TST     $AUTOB          ;ARE WE IN AUTOMATIC MODE?
002030  001001                  BNE     START1          ;BRANCH IF YES
002032  104406                  GTSWR                   ;ASK FOR SWR INPUT FROM CONSOLE
002034  005037  001202          START1: CLR     .UNIT           ;CLEAR UNIT NUMBER
002040  013737  001246  001364  MOV     $DEVM,DMAP      ;POSITION OF DRV11-J'S
002046  042737  177760  001364  BIC     #177760,DMAP    ;UP TO 4 DRV11-J'S ONLY
002054  013701  001244          MOV     $BASE,R1        ;GET BASE ADDRESS
002060  010137  001342          MOV     R1,DRCSA        ;MAKE BUS REG. POINTER = $BASE
002064  032737  000001  001364  BIT     #1,DMAP         ;IS FIRST DRV11-J SELECTED?
002072  001002                  BNE     NEXPAS          ;YES
002074  000137  012746          JMP     NXDEV1          ;ADVANCE BASE DRV11-J ADDRESS
002100  012700  001342          NEXPAS: MOV     #DRCSA,R0       ;SET UP REGISTER ADDRESS POINTERS
002104  010120                  NEXPA1: MOV     R1,(R0)+        ;LOAD EM,R1 = DRV11-J CSRA ADDRESS
002106  062701  000002          ADD     #2,R1           ;DO POINTERS FOR ALL REGS, A THRU D
002112  022700  001362          CMP     #DRDBD+2,R0     ;ALL DONE?
002116  001372                  BNE     NEXPA1          ;BR IF NOT
002120  012706  001100          MOV     #STACK,SP       ;ALWAYS RESET STACK
002124  012705  001126          MOV     #$BDDAT,R5      ;INIT R5 WITH $BDDAT
002130  013737  001202  001200  MOV     $UNIT,$DEVCT    ;LOAD APT COUNTER WITH UNIT NO.
002136  106427  000340          MTPS    #PR7            ;SET PRIORITY TO 7


                                ;;*******************************************************************
                                ;*TEST 1         TEST THAT ALL  REGISTERS ARE ADDRESSABLE
                                ;;*******************************************************************
002142  000004                  TST1:   SCOPE
002144  005037  001124          CLR     $GDDAT          ;NO DATA COMPARE
002150  005015                  CLR     (R5)            ;NO DATA COMPARE
002152  012737  002206  000004  MOV     #2$,@#ERRVEC    ;SET UP TIMEOUT RETURN ADDRS
002160  013700  001342          MOV     DRCSA,R0        ;SET UP 1ST DRV11 BUS ADRS
002164  012701  000010          MOV     #8.,R1          ;SET UP REG COUNT
002170  010037  001122          1$:     MOV     R0,$BDADR       ;SET UP CURRENT DRV BUS ADRS
002174  005010                  CLR     (R0)            ;SEE IF THERE
```

```
002176  005720                              TST     (R0)+              ;BUMP TO NEXT
002200  005301                              DEC     R1                 ;COUNT 8 OF THEM
002202  001403                              BEQ     3$                 ;BR IF ALL DONE
002204  000771                              BR      1$                 ;TRY NEXT
002206  022626                      2$:     CMP     (SP)+,(SP)+        ;FIX STACK SINCE NO RTI
002210  104001                              ERROR   1                  ;BUS ADRS INDICATED DID NOT RESPOND
002212  012737  000006  000004      3$:     MOV     #ERRVEC+2,@#ERRVEC  ;RESTORE LOC 4

                                    ;;*****************************************************************
                                    ;*TEST 2        TEST CSRA W/R DIR BIT,INT. CHIP RESET STATUS
                                    ;;*****************************************************************
002220  000004                      TST2:   SCOPE
002222  013700  001342                      MOV     DRCSA,R0           ;GET CSR ADDRESS
002226  013701  001352                      MOV     DRCSC,R1           ;STORE CSRC ADDRESS
002232  005010                              CLR     (R0)               ;INIT CSRA
002234  005011                              CLR     (R1)               ;INIT CSRC
002236  010037  001122                      MOV     R0,$BDADR          ;STORE CSR ADDRESS
                                                                       ;SET UP EXPECTED DATA
002242  012737  100700  001124              MOV     #RDY!DIR!BIT7!BIT6,$GDDAT
002250  112760  000001  000001              MOVB    #BIT0,1(R0)        ;SET DIRECTION BIT
002256  011015                              MOV     (R0),(R5)          ;GET CSR DATA
002260  042715  000007                      BIC     #7,(R5)            ;CLEAR UNDEFINED BITS
002264  023715  001124                      CMP     $GDDAT,(R5)
002270  001401                              BEQ     100$
002272  104002                              ERROR   2                  ;CSRA ERROR
002274  010137  001122              100$:   MOV     R1,$BDADR          ;STORE CSRC ADDRESS
002300  012737  000200  001124              MOV     #BIT7,$GDDAT       ;STORE EXPECTED
002306  011115                              MOV     (R1),(R5)          ;STORE CSRC
002310  042715  000007                      BIC     #7,(R5)            ;CLEAR UNDEFINED BITS
002314  023715  001124                      CMP     $GDDAT,(R5)
002320  001401                              BEQ     1$
002322  104002                              ERROR   2                  ;CSRC ERROR
002324  010037  001122              1$:     MOV     R0,$BDADR          ;CSRA ADDRESS
002330  012737  100300  001124              MOV     #RDY!BIT7!BIT6,$GDDAT
002336  105060  000001                      CLRB    1(R0)              ;CLEAR CSRA DIR BIT
002342  011015                              MOV     (R0),(R5)          ;READ CSR
002344  042715  000007                      BIC     #7,(R5)            ;CLEAR UNDEFINED BITS
002350  023715  001124                      CMP     $GDDAT,(R5)
002354  001401                              BEQ     2$                 ;;BR IF EQUAL
002356  104002                              ERROR   2
002360  012737  000300  001124      2$:     MOV     #BIT7!BIT6,$GDDAT
002366  112761  000001  000001              MOVB    #BIT0,1(R1)        ;CSRC TO OUTPUT MODE
002374  011015                              MOV     (R0),(R5)          ;READ CSRA
002376  042715  000007                      BIC     #7,(R5)            ;CLEAR UNDEFINED BITS
002402  023715  001124                      CMP     $GDDAT,(R5)        ;CHECK IF RDY BIT CLEARED
002406  001401                              BEQ     TST3               ;;BR IF EQUAL
002410  104002                              ERROR   2                  ;CSRA REG ERROR

                                    ;;*****************************************************************
                                    ;*TEST 3        TEST CSRA INT. ENABLE BIT
                                    ;;*****************************************************************
002412  000004                      TST3:   SCOPE
002414  004737  013124                      JSR     PC,CLRCSR          ;CLEAR CSR REGISTER
002420  013700  001342                      MOV     DRCSA,R0           ;GET CSRA ADDRESS
002424  013701  001352                      MOV     DRCSC,R1           ;GET CSRC ADDRESS
002430  112761  000001  000001              MOVB    #BIT0,1(R1)        ;SET DIRECTION BIT CSRC
```

```
   256  002436  010037  001122                  MOV     R0,$BDADR       ;STORE CSRA ADDRESS
   257  002442  012737  001300  001124          MOV     #BIT9!BIT7!BIT6,$GDDAT
   258  002450  012710  001000                  MOV     #IE,(R0)        ;SET INTERRUPT ENABLE BIT
   259  002454  011015                          MOV     (R0),(R5)
   260  002456  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   261  002462  023715  001124                  CMP     $GDDAT,(R5)
   262  002466  001401                          BEQ     1$
   263  002470  104002                          ERROR   2               ;INT. ENABLE ERROR,CSRA
   264  002472  012737  000300  001124  1$:     MOV     #BIT7!BIT6,$GDDAT
   265  002500  105060  000001                  CLRB    1(R0)           ;CLEAR I/E BIT
   266  002504  011015                          MOV     (R0),(R5)
   267  002506  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   268  002512  023715  001124                  CMP     $GDDAT,(R5)
   269  002516  001401                          BEQ     TST4            ;;BR IF EQUAL
   270  002520  104002                          ERROR   2               ;CSRA ERROR
   271
   272                                  ;;*****************************************************************
   (3)                                  ;*TEST 4         TEST CSRA I/E,DIR BIT
   (3)                                  ;;*****************************************************************
   (2)  002522  000004                  TST4:   SCOPE
   273  002524  004737  013124                  JSR     PC,CLRCSR       ;CLEAR CSR REGISTERS
   274  002530  013700  001342                  MOV     DRCSA,R0        ;GET CSRA ADDRESS
   275  002534  010037  001122                  MOV     R0,$BDADR       ;SAVE CSRA ADDRESS
   276  002540  012737  101700  001124          MOV     #101700,$GDDAT  ;EXPECTED I/E,DIR
   277  002546  112760  000003  000001          MOVB    #BIT1!BIT0,1(R0)
   278                                                                  ;SET I/E AND DIR BIT
   279  002554  011015                          MOV     (R0),(R5)
   280  002556  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   281  002562  023715  001124                  CMP     $GDDAT,(R5)
   282  002566  001401                          BEQ     1$
   283  002570  104002                          ERROR   2               ;CSRA ERROR
   284  002572  012737  100300  001124  1$:     MOV     #RDY!BIT7!BIT6,$GDDAT
   285  002600  005010                          CLR     (R0)
   286  002602  011015                          MOV     (R0),(R5)
   287  002604  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   288  002610  023715  001124                  CMP     $GDDAT,(R5)
   289  002614  001401                          BEQ     TST5            ;;BR IF EQUAL
   290  002616  104002                          ERROR   2               ;CSRA ERROR
   291
   292
   293                                  ;;*****************************************************************
   (3)                                  ;*TEST 5         TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS
   (3)                                  ;;*****************************************************************
   (2)  002620  000004                  TST5:   SCOPE
   294  002622  004737  013124                  JSR     PC,CLRCSR       ;CLEAR CSR REGISTERS
   295  002626  013700  001346                  MOV     DRCSB,R0        ;GET CSR ADDRESS
   296  002632  013701  001356                  MOV     DRCSD,R1        ;GET CSRD ADDRESS
   297  002636  010037  001122                  MOV     R0,$BDADR       ;STORE CSR ADDRESS
   298  002642  012737  100400  001124          MOV     #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
   299  002650  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET DIRECTION BIT
   300  002656  011015                          MOV     (R0),(R5)       ;GET CSRB DATA
   301  002660  023715  001124                  CMP     $GDDAT,(R5)
   302  002664  001401                          BEQ     100$
   303  002666  104002                          ERROR   2               ;CSRB ERROR
   304  002670  010137  001122          100$:   MOV     R1,$BDADR       ;STORE CSRD ADDRESS
   305  002674  005037  001124                  CLR     $GDDAT          ;STORE EXPECTED
```

```
   306  002700  011115                          MOV     (R1),(R5)       ;READ CSRD
   307  002702  023715  001124                  CMP     $GDDAT,(R5)
   308  002706  001401                          BEQ     1$
   309  002710  104002                          ERROR   2               ;CSRD ERROR
   310  002712  010037  001122          1$:     MOV     R0,$BDADR       ;STORE CSRB ADDRESS
   311  002716  012737  100000  001124          MOV     #RDY,$GDDAT     ;STORE EXPECTED
   312  002724  105060  000001                  CLRB    1(R0)           ;CLEAR CSR DIR BIT
   313  002730  011015                          MOV     (R0),(R5)       ;READ CSR
   314  002732  023715  001124                  CMP     $GDDAT,(R5)
   315  002736  001401                          BEQ     2$              ;;BR IF EQUAL
   316  002740  104002                          ERROR   2
   317  002742  005037  001124          2$:     CLR     $GDDAT          ;STORE EXPECTED
   318  002746  112761  000001  000001          MOVB    #BIT0,1(R1)     ;CSRD TO OUTPUT MODE
   319  002754  011015                          MOV     (R0),(R5)       ;READ CSRB
   320  002756  023715  001124                  CMP     $GDDAT,(R5)     ;RDY BIT CLEARED
   321  002762  001401                          BEQ     TST6            ;;BR IF EQUAL
   322  002764  104002                          ERROR   2               ;CSRB REG ERROR
   323
   324                                  ;;*******************************************************************
   (3)                                  ;*TEST 6        TEST CSRC W/R DIR BIT,INT CHIP RESET STATUS
   (3)                                  ;;*******************************************************************
   (2)  002766  000004                  TST6:   SCOPE
   325  002770  004737  013124                  JSR     PC,CLRCSR       ;CLEAR CSR REGISTERS
   326  002774  013700  001352                  MOV     DRCSC,R0        ;GET CSR ADDRESS
   327  003000  013701  001342                  MOV     DRCSA,R1        ;GET CSRA ADDRESS
   328  003004  010037  001122                  MOV     R0,$BDADR       ;STORE CSR ADDRESS
   329  003010  012737  100600  001124          MOV     #RDY!DIR!BIT7,$GDDAT
   330  003016  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET DIRECTION BIT
   331  003024  011015                          MOV     (R0),(R5)       ;GET CSR DATA
   332  003026  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   333  003032  023715  001124                  CMP     $GDDAT,(R5)
   334  003036  001401                          BEQ     100$
   335  003040  104002                          ERROR   2
   336  003042  010137  001122          100$:   MOV     R1,$BDADR       ;STORE CSR ADDRESS
   337  003046  012737  000300  001124          MOV     #BIT7!BIT6,$GDDAT
   338  003054  011115                          MOV     (R1),(R5)       ;READ CSRA
   339  003056  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   340  003062  023715  001124                  CMP     $GDDAT,(R5)
   341  003066  001401                          BEQ     1$              ;CHECK CSRA
   342  003070  104002                          ERROR   2               ;CSRA ERROR
   343  003072  010037  001122          1$:     MOV     R0,$BDADR       ;STORE CSR ADDRESS
   344  003076  012737  100200  001124          MOV     #RDY!BIT7,$GDDAT;STORE EXPECTED DATA
   345  003104  105060  000001                  CLRB    1(R0)           ;CLEAR CSR DIR BIT
   346  003110  011015                          MOV     (R0),(R5)       ;READ CSR
   347  003112  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   348  003116  023715  001124                  CMP     $GDDAT,(R5)
   349  003122  001401                          BEQ     2$              ;;BR IF EQUAL
   350  003124  104002                          ERROR   2
   351  003126  012737  000200  001124  2$:     MOV     #BIT7,$GDDAT    ;EXPECTED DATA
   352  003134  112761  000001  000001          MOVB    #BIT0,1(R1)     ;CSRA TO OUTPUT MODE
   353  003142  011015                          MOV     (R0),(R5)       ;READ CSRC
   354  003144  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   355  003150  023715  001124                  CMP     $GDDAT,(R5)     ;RDY BIT CLEARED
   356  003154  001401                          BEQ     TST7            ;;BR IF EQUAL
   357  003156  104002                          ERROR   2               ;CSRC REG ERROR
   358
```

```
359                                     ;;*****************************************************************
(3)                                     ;*TEST 7         TEST CSRD W/R DIR BIT,INT CHIP RESET STATUS
(3)                                     ;;*****************************************************************
(2)  003160  000004                     TST7:   SCOPE
360  003162  004737  013124                    JSR     PC,CLRCSR       ;CLEAR CSR REGISTERS
361  003166  013700  001356                    MOV     DRCSD,R0        ;GET CSR ADDRESS
362  003172  013701  001346                    MOV     DRCSB,R1        ;CSRB ADDRESS
363  003176  010037  001122                    MOV     R0,$BDADR       ;STORE CSR ADDRESS
364  003202  012737  100400  001124            MOV     #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
365  003210  112760  000001  000001            MOVB    #BIT0,1(R0)     ;SET DIRECTION BIT
366  003216  011015                            MOV     (R0),(R5)       ;GET CSR DATA
367  003220  023715  001124                    CMP     $GDDAT,(R5)
368  003224  001401                            BEQ     100$
369  003226  104002                            ERROR   2
370  003230  010137  001122            100$:   MOV     R1,$BDADR       ;STORE CSRB ADDRESS
371  003234  005037  001124                    CLR     $GDDAT          ;STORE EXPECTED
372  003240  011115                            MOV     (R1),(R5)
373  003242  023715  001124                    CMP     $GDDAT,(R5)
374  003246  001401                            BEQ     1$
375  003250  104002                            ERRCR   2               ;CSRB ERROR
376  003252  010037  001122            1$:     MOV     R0,$BDADR       ;STORE CSRD ADDRESS
377  003256  012737  100000  001124            MOV     #RDY,$GDDAT     ;STORE EXPECTED
378  003264  105060  000001                    CLRB    1(R0)           ;CLEAR CSR DIR BIT
379  003270  011015                            MOV     (R0),(R5)       ;READ CSR
380  003272  023715  001124                    CMP     $GDDAT,(R5)
381  003276  001401                            BEQ     2$              ;;BR IF EQUAL
382  003300  104002                            ERROR   2
383  003302  005037  001124            2$:     CLR     $GDDAT          ;EXPECTED
384  003306  112761  000001  000001            MOVB    #BIT0,1(R1)     ;CSRD TO OUTPUT MODE
385  003314  011015                            MOV     (R0),(R5)       ;READ CSR
386  003316  023715  001124                    CMP     $GDDAT,(R5)     ;RDY BIT CLEARED
387  003322  001401                            BEQ     TST10           ;;BR IF EQUAL
388  003324  104002                            ERROR   2               ;CSRD REG ERROR
389
390                                     ;;*****************************************************************
(3)                                     ;*TEST 10        TEST DBRA W/R IN OUTPUT MODE
(3)                                     ;;*****************************************************************
(2)  003326  000004                     TST10:  SCOPE
391  003330  004737  013124                    JSR     PC,CLRCSR       ;CLEAR ALL CSRS
392  003334  012703  013232                    MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
393  003340  012737  003370  001110            MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
394  003346  013700  001342                    MOV     DRCSA,R0        ;GET CSRA
395  003352  013701  001344                    MOV     DRDBA,R1        ;GET DBRA ADDRESS
396  003356  010137  001122                    MOV     R1,$BDADR       ;STORE DBRA ADDRESS
397  003362  112760  000001  000001            MOVB    #BIT0,1(R0)     ;SET CSRA IN OUTPUT MODE
398  003370  011337  001124            1$:     MOV     (R3),$GDDAT     ;SAVE EXPECTED DATA
399  003374  005011                            CLR     (R1)            ;CLEAR DBRA
400  003376  051311                            BIS     (R3),(R1)       ;WRITE INTO DBRA
401  003400  011115                            MOV     (R1),(R5)       ;READ DBRA
402  003402  023715  001124                    CMP     $GDDAT,(R5)     ;CHECK W/R DBRA
403  003406  001401                            BEQ     2$              ;NEXT PAT. IF EQUAL
404  003410  104002                            ERROR   2               ;DBRA W/R ERROR
405  003412  005723                    2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
406  003414  020327  013532                    CMP     R3,#ENDDAT      ;CHECK FOR END
407  003420  001363                            BNE     1$              ;DO NEXT PATTERN
408
```

```
                                ;;*****************************************************************
                                ;*TEST 11        TEST DBRB W/R IN OUTPUT MODE
                                ;;*****************************************************************
003422  000004                  TST11:  SCOPE
003424  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
003430  012703  013232                  MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
003434  012737  003464  001110          MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
003442  013700  001346                  MOV     DRCSB,R0        ;GET CSRB
003446  013701  001350                  MOV     DRDBB,R1        ;GET DBRB ADDRESS
003452  010137  001122                  MOV     R1,$BDADR       ;STORE DBRB ADDRESS
003456  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRB IN OUTPUT MODE
003464  011337  001124          1$:     MOV     (R3),$GDDAT     ;SAVE EXPECTED DATA
003470  005011                          CLR     (R1)            ;CLEAR DBRB
003472  051311                          BIS     (R3),(R1)       ;WRITE INTO DBRB
003474  011115                          MOV     (R1),(R5)       ;READ DBRB
003476  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK W/R DBRB
003502  001401                          BEQ     2$              ;NEXT PAT. IF EQUAL
003504  104002                          ERROR   2               ;DBRB W/R ERROR
003506  005723                  2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
003510  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR END
003514  001363                          BNE     1$              ;DO NEXT PATTERN

                                ;;*****************************************************************
                                ;*TEST 12        TEST DBRC W/R IN OUTPUT MODE
                                ;;*****************************************************************
003516  000004                  TST12:  SCOPE
003520  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
003524  012703  013232                  MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
003530  012737  003560  001110          MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
003536  013700  001352                  MOV     DRCSC,R0        ;GET CSRC
003542  013701  001354                  MOV     DRDBC,R1        ;GET DBRC ADDRESS
003546  010137  001122                  MOV     R1,$BDADR       ;STORE DBRC ADDRESS
003552  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRC IN OUTPUT MODE
003560  011337  001124          1$:     MOV     (R3),$GDDAT     ;SAVE EXPECTED DATA
003564  005011                          CLR     (R1)            ;CLEAR DBRC
003566  051311                          BIS     (R3),(R1)       ;WRITE INTO DBRC
003570  011115                          MOV     (R1),(R5)       ;READ DBRC
003572  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK W/R DBRC
003576  001401                          BEQ     2$              ;NEXT PAT. IF EQUAL
003600  104002                          ERROR   2               ;DBRC W/R ERROR
003602  005723                  2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
003604  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR END
003610  001363                          BNE     1$              ;DO NEXT PATTERN

                                ;;*****************************************************************
                                ;*TEST 13        TEST DBRD W/R IN OUTPUT MODE
                                ;;*****************************************************************
003612  000004                  TST13:  SCOPE
003614  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
003620  012703  013232                  MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
003624  012737  003654  001110          MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
003632  013700  001356                  MOV     DRCSD,R0        ;GET CSRD
003636  013701  001360                  MOV     DRDBD,R1        ;GET DBRD ADDRESS
003642  010137  001122                  MOV     R1,$BDADR       ;STORE DBRD ADDRESS
```

SEQ 0026

```
  456  003646  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRD IN OUTPUT MODE
  457  003654  011337  001124          1$:     MOV     (R3),$GDDAT     ;SAVE EXPECTED DATA
  458  003660  005011                          CLR     (R1)            ;CLEAR DBRD
  459  003662  051311                          BIS     (R3),(R1)       ;WRITE INTO DBRD
  460  003664  011115                          MOV     (R1),(R5)       ;READ DBRD
  461  003666  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK W/R DBRD
  462  003672  001401                          BEQ     2$              ;NEXT PAT. IF EQUAL
  463  003674  104002                          ERROR   2               ;DBRD W/R ERROR
  464  003676  005723                  2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
  465  003700  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR END
  466  003704  001363                          BNE     1$              ;DO NEXT PATTERN
```

```
                                        ;;*****************************************************************
                                        ;*TEST 14        TEST CSR UNIQUENESS,CSRS (A-B),(C-D)
                                        ;;*****************************************************************
003706  000004                  TST14:  SCOPE
003710  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
                                                                ;CSRA = R0
                                                                ;CSRB = R1
                                                                ;CSRC = R2
                                                                ;CSRD = R3
003714  112760  000003  000001          MOVB    #3,1(R0)        ;CSRA OUTPUT MODE,I/E
003722  112763  000001  000001          MOVB    #BIT0,1(R3)     ;CSRD OUTPUT MODE
003730  010037  001122                  MOV     R0,$BDADR
003734  012737  101700  001124          MOV     #101700,$GDDAT  ;EXPECTED DATA
003742  011015                          MOV     (R0),(R5)
003744  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
003750  023715  001124                  CMP     $GDDAT,(R5)
003754  001401                          BEQ     1$
003756  104002                          ERROR   2               ;CSRA ERROR
003760  010137  001122          1$:     MOV     R1,$BDADR       ;CHECK CSRB
003764  005037  001124                  CLR     $GDDAT
003770  011115                          MOV     (R1),(R5)
003772  023715  001124                  CMP     $GDDAT,(R5)
003776  001401                          BEQ     2$
004000  104002                          ERROR   2               ;CSRB ERROR
004002  010337  001122          2$:     MOV     R3,$BDADR
004006  012737  100400  001124          MOV     #100400,$GDDAT
004014  011315                          MOV     (R3),(R5)
004016  023715  001124                  CMP     $GDDAT,(R5)
004022  001401                          BEQ     3$
004024  104002                          ERROR   2               ;CSRD ERROR
004026  010237  001122          3$:     MOV     R2,$BDADR       ;CHECK CSRC
004032  012737  000200  001124          MOV     #BIT7,$GDDAT    ;EXPECTED
004040  011215                          MOV     (R2),(R5)
004042  042715  000007                  BIC     #7,(R5)
004046  023715  001124                  CMP     $GDDAT,(R5)
004052  001401                          BEQ     TST15           ;;BR IF EQUAL
004054  104002                          ERROR   2               ;CSRC ERROR

                                        ;;*****************************************************************
                                        ;*TEST 15        TEST CSR UNIQUENESS,CSRS (A-D),(C-B)
                                        ;;*****************************************************************
004056  0000                    TST15:  SCOPE
004060  0074  013124                    JSR     PC,CLRCSR       ;CLR ALL CSRS
                                                                ;CSRA = R0
                                                                ;CSRB = R1
                                                                ;CSRC = R2
                                                                ;CSRD = R3
004064  112760  000003  000001          MOVB    #3,1(R0)        ;CSRA OUTPUT,I/E
004072  112761  000001  000001          MOVB    #BIT0,1(R1)     ;CSRB OUTPUT MODE
004100  010037  001122                  MOV     R0,$BDADR       ;CSRA ADDRESS
004104  012737  101700  001124          MOV     #101700,$GDDAT
004112  011015                          MOV     (R0),(R5)
004114  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
004120  023715  00112                   CMP     $GDDAT,(R5)
```

```
   518  004124  001400                          BEQ     1$
   519  004126  010137  001122          1$:     MOV     R1,$BDADR       ;CHECK CSRB
   520  004132  012737  100400  001124          MOV     #100400,$GDDAT
   521  004140  011115                          MOV     (R1),(R5)
   522  004142  023715  001124                  CMP     $GDDAT,(R5)
   523  004146  001401                          BEQ     2$
   524  004150  104002                          ERROR   2               ;CSRB ERROR
   525  004152  010237  001122          2$:     MOV     R2,$BDADR       ;CHECK CSRC
   526  004156  012737  000200  001124          MOV     #BIT7,$GDDAT
   527  004164  011215                          MOV     (R2),(R5)
   528  004166  042715  000007                  BIC     #7,(R5)
   529  004172  023715  001124                  CMP     $GDDAT,(R5)
   530  004176  001401                          BEQ     3$
   531  004200  104002                          ERROR   2               ;CSRC ERROR
   532  004202  010337  001122          3$:     MOV     R3,$BDADR
   533  004206  005037  001124                  CLR     $GDDAT
   534  004212  011315                          MOV     (R3),(R5)
   535  004214  023715  001124                  CMP     $GDDAT,(R5)
   536  004220  001401                          BEQ     TST16           ;;BR IF EQUAL
   537  004222  104002                          ERROR   2               ;CSRD ERROR
   538
   539                                  ;;*****************************************************************
   540                                  ;;
   (3)                                  ;*TEST 16       TEST PORT A TO PORT C INTERACTION
   (3)                                  ;;
   (2)  004224  000004                  TST16:  SCOPE
   541  004226  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
   542  004232  012703  013232                  MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
   543  004236  012737  004266  001110          MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
   544  004244  013700  001342                  MOV     DRCSA,R0        ;GET PORT A, CSRA ADDRESS
   545  004250  013701  001352                  MOV     DRCSC,R1        ;GET PORT C, CSRC ADDRESS
   546  004254  013702  001346                  MOV     DRCSB,R2        ;CSRB ADDRESS
   547  004260  112762  000001  000001          MOVB    #BIT0,1(R2)     ;CSRB IN OUTPUT MODE
   548  004266  010037  001122          1$:     MOV     R0,$BDADR       ;STORE CSRA ADDRESS
   549  004272  005062  000002                  CLR     2(R2)           ;CLEAR DBRB
   550  004276  105061  000001                  CLRB    1(R1)           ;PORT C,CSRC,INPUT MODE
   551  004302  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRA IN OUTPUT MODE
   552  004310  011360  000002                  MOV     (R3),2(R0)      ;WRITE INTO DBRA
   553  004314  016104  000002                  MOV     2(R1),R4        ;READ DBRC
   554  004320  012737  100700  001124          MOV     #RDY!DIR!BIT7!BIT6,$GDDAT
   555                                                                  ;CSRA DIR SHOULD STAY SET
   556  004326  011015                          MOV     (R0),(R5)       ;READ CSRA
   557  004330  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   558  004334  023715  001124                  CMP     $GDDAT,(R5)
   559  004340  001401                          BEQ     100$
   560  004342  104002                          ERROR   2               ;CSRA ERROR
   561  004344  012737  000200  001124  100$:  MOV     #BIT7,$GDDAT
   562  004352  010137  001122                  MOV     R1,$BDADR       ;CSRC ADDRESS
   563  004356  011115                          MOV     (R1),(R5)       ;READ CSRC
   564  004360  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
   565  004364  023715  001124                  CMP     $GDDAT,(R5)
   566  004370  001401                          BEQ     101$
   567  004372  104002                          ERROR   2               ;CSRC ERROR
   568  004374  013737  001354  001122  101$:  MOV     DRDBC,$BDADR    ;STORE DBRC ADDRESS
   569  004402  011337  001124                  MOV     (R3),$GDDAT     ;SAVE EXPECTED
   570  004406  010415                          MOV     R4,(R5)         ;DBRC CONTENTS
```

```
                                                CMP     $GDDAT,(R5)     ;CHECK PORT C,DBRC
                                                BEQ     2$              ;BEQ TO NEXT SUBTEST
                                                ERROR   2               ;DBRC REG ERROR
                                        2$:     COM     $GDDAT          ;SET UP TO WRITE COM DATA FROM
                                                                        ;PORT C TO PORT A
                                                MOV     DRDBA,$BDADR    ;STORE DBRA ADDRESS
                                        3$:     MOV     $GDDAT,2(R1)    ;WRITE PORT C,INPUT MODE
                                                CLRB    1(R0)           ;MAKE PORT A INPUT MODE
                                                MOVB    #BIT0,1(R1)     ;MAKE PORT C OUTPUT MODE
                                                MOV     2(R0),(R5)      ;READ PORT A,DBRA
                                                CMP     $GDDAT,(R5)     ;PORT A =PORT C?
                                                BEQ     4$              ;CHECK DBRB FOR NO DATA CHANGE
                                                ERROR   2               ;PORT A,DBRA REG ERROR
                                        4$:     MOV     DRDBB,$BDADR    ;STORE DBRB ADDRESS
                                                CLR     $GDDAT          ;CONTENTS SHOULD STAY ZERO
                                                MOV     2(R2),(R5)      ;READ DBRB FOR CLEAR
                                                CMP     $GDDAT,(R5)     ;CHECK FOR DBRB CLEAR
                                                BEQ     5$              ;YES,CONTINUE
                                                ERROR   2               ;DBRB INTERACTION ERROR
                                        5$:     TST     (R3)+           ;INC FOR NEXT PATTERN
                                                CMP     R3,#ENDDAT      ;CHECK FOR END
                                                BNE     1$              ;DO NEXT PATTERN

                                ;;*************************************************************
                                ;*TEST 17       TEST PORT B TO PORT D INTERACTION
                                ;;*************************************************************
                                TST17:  SCOPE
                                        JSR     PC,CLRCSR       ;CLEAR ALL CSRS
                                        MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
                                        MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
                                        MOV     DRCSB,R0        ;GET PORT B, CSRB ADDRESS
                                        MOV     DRCSD,R1        ;GET PORT D, CSRD ADDRESS
                                        MOV     DRCSA,R2        ;STORE CSRA ADDRESS
                                        MOVB    #BIT0,1(R2)     ;CSRA IN OUTPUT MODE
                                1$:     MOV     R0,$BDADR       ;STORE CSRB ADDRESS
                                        CLR     2(R2)           ;CLEAR DBRA
                                        CLRB    1(R1)           ;PORT D,CSRD,INPUT MODE
                                        MOVB    #BIT0,1(R0)     ;SET CSRB IN OUTPUT MODE
                                        MOV     (R3),2(R0)      ;WRITE INTO DBRB
                                        MOV     2(R1),R4        ;DBRD CONTENTS
                                        MOV     #RDY!DIR,$GDDAT ;SAVE EXPECTED
                                        MOV     (R0),(R5)       ;READ CSRB
                                        CMP     $GDDAT,(R5)
                                        BEQ     100$
                                        ERROR   2               ;CSRB ERROR
                                100$:   CLR     $GDDAT          ;SAVE EXPECTED
                                        MOV     R1,$BDADR       ;SAVE CSRD ADDRESS
                                        MOV     (R1),(R5)
                                        CMP     $GDDAT,(R5)
                                        BEQ     101$
                                        ERROR   2               ;CSRD ERROR
                                101$:   MOV     DRDBD,$BDADR    ;SAVE DBRD ADDRESS
                                        MOV     (R3),$GDDAT     ;SAVE EXPECTED
                                        MOV     R4,(R5)         ;DBRD CONTENTS
                                        CMP     $GDDAT,(R5)     ;CHECK PORT D,DBRD
```

```
                                                BEQ     2$              ;BEQ TO NEXT SUBTEST
                                                ERROR   2               ;DBRD REG ERROR
                                        2$:     COM     $GDDAT          ;SET UP TO WRITE COM DATA FROM
                                                                        ;PORT D TO PORT B
                                                MOV     DRDBB,$BDADR    ;STORE DBRB ADDRESS
                                        3$:     MOV     $GDDAT,2(R1)    ;WRITE PORT D,INPUT MODE
                                                CLRB    1(R0)           ;MAKE PORT B INPUT MODE
                                                MOVB    #BIT0,1(R1)     ;MAKE PORT D OUTPUT MODE
                                                MOV     2(R0),(R5)      ;READ PORT B,DBRB
                                                CMP     $GDDAT,(R5)     ;PORT B =PORT D?
                                                BEQ     4$              ;CHECK DBRA FOR NO DATA CHANGE
                                                ERROR   2               ;PORT B,DBRB REG ERROR
                                        4$:     MOV     DRDBA,$BDADR    ;STORE DBRA ADDRESS
                                                CLR     $GDDAT          ;CONTENTS = 0
                                                MOV     2(R2),(R5)      ;READ DBRA FOR CLEAR
                                                CMP     $GDDAT,(R5)     ;DBRA CLEAR?
                                                BEQ     5$              ;YES,CONTINUE
                                                ERROR   2               ;DBRA INTERACTION ERROR
                                        5$:     TST     (R3)+           ;INC FOR NEXT PATTERN
                                                CMP     R3,#ENDDAT      ;CHECK FOR END
                                                BNE     1$              ;DO NEXT PATTERN

                                ;;*****************************************************************
                                ;*TEST 20       TEST PORT C TO PORT A INTERACTION
                                ;;*****************************************************************
                                TST20:  SCOPE
                                                JSR     PC,CLRCSR       ;CLEAR ALL CSRS
                                                MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
                                                MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
                                                MOV     DRCSC,R0        ;GET PORT C, CSRC ADDRESS
                                                MOV     DRCSA,R1        ;GET PORT A, CSRA ADDRESS
                                                MOV     DRCSD,R2        ;STORE CSRD ADDRESS
                                                MOVB    #BIT0,1(R2)     ;CSRD IN OUTPUT MODE
                                        1$:     MOV     R0,$BDADR       ;STORE CSRC ADDRESS
                                                CLR     2(R2)           ;CLEAR DBRC
                                                CLRB    1(R1)           ;PORT A,CSRA,INPUT MODE
                                                MOVB    #BIT0,1(R0)     ;SET CSRC IN OUTPUT MODE
                                                MOV     (R3),2(R0)      ;WRITE INTO DBRC
                                                MOV     2(R1),R4        ;READ DBRA
                                                MOV     #RDY!DIR!BIT7,$GDDAT
                                                                        ;CSRC DIR SHOULD BE SET
                                                MOV     (R0),(R5)       ;READ CSRC
                                                BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
                                                CMP     $GDDAT,(R5)
                                                BEQ     100$
                                                ERROR   2               ;CSRC ERROR
                                        100$:   MOV     #BIT7!BIT6,$GDDAT
                                                MOV     R1,$BDADR       ;CSRA ADDRESS
                                                MOV     (R1),(R5)       ;READ CSRA
                                                BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
                                                CMP     $GDDAT,(R5)
                                                BEQ     101$
                                                ERROR   2               ;CSRA ERROR
                                        101$:   MOV     DRDBA,$BDADR    ;STORE DBRA ADDRESS
                                                MOV     (R3),$GDDAT     ;SAVE EXPECTED
```

```
   677  005174  010415                          MOV     R4,(R5)         ;DBRA CONTENTS
   678  005176  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK PORT A,DBRA
   679  005202  001401                          BEQ     2$              ;BEQ TO NEXT SUBTEST
   680  005204  104002                          ERROR   2               ;DBRA REG ERROR
   681  005206  005137  001124          2$:     COM     $GDDAT          ;SET UP TO WRITE COM DATA FROM
   682                                                                  ;PORT A TO PORT C
   683  005212  013737  001354  001122          MOV     DRDBC,$BDADR    ;STORE DBRC ADDRESS
   684  005220  013761  001124  000002  3$:     MOV     $GDDAT,2(R1)    ;WRITE PORT A,INPUT MODE
   685  005226  105060  000001                  CLRB    1(R0)           ;MAKE PORT C INPUT MODE
   686  005232  112761  000001  000001          MOVB    #BIT0,1(R1)     ;MAKE PORT A OUTPUT MODE
   687  005240  016015  000002                  MOV     2(R0),(R5)      ;READ PORT C,DBRC
   688  005244  023715  001124                  CMP     $GDDAT,(R5)     ;PORT C =PORT A?
   689  005250  001401                          BEQ     4$              ;CHECK DBRD FOR NO DATA CHANGE
   690  005252  104002                          ERROR   2               ;PORT C,DBRC REG ERROR
   691  005254  013737  001360  001122  4$:     MOV     DRDBD,$BDADR    ;STORE DBRD ADDRESS
   692  005262  005037  001124                  CLR     $GDDAT          ;DBRD = 0 EXPECTED
   693  005266  016215  000002                  MOV     2(R2),(R5)      ;READ DBRD FOR CLEAR
   694  005272  023715  001124                  CMP     $GDDAT,(R5)     ;IS DBRD CLEAR?
   695  005276  001401                          BEQ     5$              ;YES,CONTINUE
   696  005300  104002                          ERROR   2               ;DBRD INTERACTION ERROR
   697  005302  005723                  5$:     TST     (R3)+           ;INC FOR NEXT PATTERN
   698  005304  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR END
   699  005310  001261                          BNE     1$              ;DO NEXT PATTERN
   700
   701                                  ;;*****************************************************************
   (3)                                 ;*TEST 21       TEST PORT D TO PORT B INTERACTION
   (3)                                 ;;*****************************************************************
   (2)  005312  000004                  TST21:  SCOPE
   702  005314  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
   703  005320  012703  013232                  MOV     #BEGPAT,R3      ;GET DATA PATTERN TABLE
   704  005324  012737  005354  001110          MOV     #1$,$LPERR      ;SET UP SCOPE ADDRESS
   705  005332  013700  001356                  MOV     DRCSD,R0        ;GET PORT D, CSRD ADDRESS
   706  005336  013701  001346                  MOV     DRCSB,P1        ;GET PORT B, CSRB ADDRESS
   707  005342  013702  001352                  MOV     DRCSC,R2        ;STORE CSRC ADDRESS
   708  005346  112762  000001  000001          MOVB    #BIT0,1(R2)     ;CSRC IN OUTPUT MODE
   709  005354  010037  001122          1$:     MOV     R0,$BDADR       ;STORE CSRD ADDRESS
   710  005360  005062  000002                  CLR     2(R2)           ;CLEAR DBRC
   711  005364  105061  000001                  CLRB    1(R1)           ;PORT B,CSRB,INPUT MODE
   712  005370  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRD IN OUTPUT MODE
   713  005376  011360  000002                  MOV     (R3),2(R0)      ;WRITE INTO DBRD
   714  005402  016104  000002                  MOV     2(R1),R4        ;READ DBRB
   715  005406  012737  100400  001124          MOV     #RDY!DIR,$GDDAT ;SAVE EXPECTED
   716  005414  011015                          MOV     (R0),(R5)       ;READ CSRD
   717  005416  023715  001124                  CMP     $GDDAT,(R5)
   718  005422  001401                          BEQ     100$
   719  005424  104002                          ERROR   2               ;CSRD ERROR
   720  005426  005037  001124          100$:   CLR     $GDDAT          ;SAVE EXPECTED
   721  005432  010137  001122                  MOV     R1,$BDADR       ;SAVE CSRB ADDRESS
   722  005436  011115                          MOV     (R1),(R5)
   723  005440  023715  001124                  CMP     $GDDAT,(R5)
   724  005444  001401                          BEQ     101$
   725  005446  104002                          ERROR   2               ;CSRB ERROR
   726  005450  013737  001350  001122  101$:   MOV     DRDBB,$BDADR    ;SAVE DBRB ADDRESS
   727  005456  011337  001124                  MOV     (R3),$GDDAT     ;SAVE EXPECTED
   728  005462  010415                          MOV     R4,(R5)         ;DBRB CONTENTS
   729  005464  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK PORT B,DBRB
```

```
   730  005470  001401                          BEQ     2$              ;BEQ TO NEXT SUBTEST
   731  005472  104002                          ERROR   2               ;DBRB REG ERROR
   732  005474  005137  001124          2$:     COM     $GDDAT          ;SET UP TO WRITE COM DATA FROM
   733                                                                  ;PORT B TO PORT D
   734  005500  013737  001360  001122          MOV     DRDBD,$BDADR    ;STORE  DBRD ADDRESS
   735  005506  013761  001124  000002  3$:     MOV     $GDDAT,2(R1)    ;WRITE PORT B,INPUT MODE
   736  005514  105060  000001                  CLRB    1(R0)           ;MAKE PORT D INPUT MODE
   737  005520  112761  000001  000001          MOVB    #BIT0,1(R1)     ;MAKE PORT B OUTPUT MODE
   738  005526  016015  000002                  MOV     2(R0),(R5)      ;READ PORT D,DBRD
   739  005532  023715  001124                  CMP     $GDDAT,(R5)     ;PORT B =PORT D?
   740  005536  001401                          BEQ     4$              ;CHECK DBRC FOR NO DATA CHANGE
   741  005540  104002                          ERROR   2               ;PORT D,DBRD REG ERROR
   742  005542  013737  001354  001122  4$:     MOV     DRDBC,$BDADR    ;STORE DBRC ADDRESS
   743  005550  005037  001124                  CLR     $GDDAT          ;EXPECT DBRC = 0
   744  005554  016215  000002                  MOV     2(R2),(R5)      ;READ DBRC
   745  005560  023715  001124                  CMP     $GDDAT,(R5)     ;IS DBRC = 0
   746  005564  001401                          BEQ     5$              ;YES,CONTINUE
   747  005566  104002                          ERROR   2               ;DBRC INTERACTION ERROR
   748  005570  005723                  5$:     TST     (R3)+           ;INC FOR NEXT PATTERN
   749  005572  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR END
   750  005576  001266                          BNE     1$              ;DO NEXT PATTERN
   751
   752                                          ;;*************************************************************************
   (3)                                          ;*TEST 22        TEST GROUP 1,2 IMR,IRR,ACR WITH CHIP RESET
   (3)                                          ;;*************************************************************************
   (2)  005600  000004                  TST22:  SCOPE
   753  005602  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
   754  005606  012702  000002                  MOV     #2,R2           ;TWO GROUPS TO TEST
   755  005612  013700  001342                  MOV     DRCSA,R0        ;STORE CSRA
   756  005616  013701  001346                  MOV     DRCSB,R1        ;STORE CSRB
   757  005622  004737  013174                  JSR     PC,CLRIRR       ;CLEAR IRR REGISTERS
   758  005626  010137  001122          111$:   MOV     R1,$BDADR       ;STORE ADDRESS
   759  005632  012737  000377  001124          MOV     #377,$GDDAT     ;STORE EXPECTED DATA
   760  005640  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
   761  005644  111115                          MOVB    (R1),(R5)       ;READ IMR REGISTER
   762  005646  023715  001124                  CMP     $GDDAT,(R5)
   763  005652  001401                          BEQ     1$
   764  005654  104005                          ERROR   5               ;IMR REG ERROR
   765  005656  005037  001124          1$:     CLR     $GDDAT          ;STORE EXPECTED
   766  005662  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
   767  005666  111115                          MOVB    (R1),(R5)       ;READ IRR REGISTER
   768  005670  023715  001124                  CMP     $GDDAT,(R5)
   769  005674  001401                          BEQ     2$
   770  005676  104003                          ERROR   3               ;IRR REG ERROR
   771  005700  112710  000254          2$:     MOVB    #MACR,(R0)      ;LOAD MODE BITS FOR ACR
   772  005704  111115                          MOVB    (R1),(R5)       ;READ ACR REGISTER
   773  005706  023715  001124                  CMP     $GDDAT,(R5)
   774  005712  001401                          BEQ     3$
   775  005714  104004                          ERROR   4               ;ACR REG ERROR
   776  005716  005302                  3$:     DEC     R2              ;FINISHED BOTH GROUPS?
   777  005720  001405                          BEQ     TST23           ;;BR IF EQUAL
   778  005722  013700  001352                  MOV     DRCSC,R0
   779  005726  013701  001356                  MOV     DRCSD,R1
   780  005732  000735                          BR      111$
   781
   782                                          ;;*************************************************************************
```

```
    (3)                                 ;*TEST 23       TEST GROUPS 1 AND 2 ACR UNIQUENESS
    (3)                                 ;;*****************************************************************
    (2)  005734  000004                 TST23:  SCOPE
    783  005736  004737  013124                   JSR     PC,CLRCSR       ;CLEAR ALL CSRS
    784  005742  013700  001342                   MOV     DRCSA,R0        ;CSRA ADDRESS
    785  005746  013701  001346                   MOV     DRCSB,R1        ;CSRB ADDRESS
    786  005752  012703  000002                   MOV     #2,R3           ;COUNTER FOR TESTING TWO GROUPS
    787  005756  004737  013174                   JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
    788  005762  010137  001122           111$:   MOV     R1,$BDADR       ;STORE ADDRESS
    789  005766  012737  000252  001124         MOV     #252,$GDDAT     ;STORE EXPECTED
    790  005774  112710  000254                 MOVB    #MACR,(R0)      ;LOAD MODE BITS FOR ACR
    791  006000  112710  000300                 MOVB    #PACR,(R0)      ;PRESELECT ACR FOR WRITING
    792  006004  113711  001124                 MOVB    $GDDAT,(R1)     ;WRITE INTO DATA PORT
    793  006010  111115                         MOVB    (R1),(R5)       ;STORE DATA FOR COMPARE
    794  006012  023715  001124                 CMP     $GDDAT,(R5)     ;CHECK ACR RESULTS
    795  006016  001401                         BEQ     1$              ;
    796  006020  104004                         ERROR   4               ;ACR ERROR
    797  006022  112710  000244           1$:     MOVB    #MIMR,(R0)      ;CHANGE TO IMR REGISTER
    798  006026  012737  000377  001124         MOV     #377,$GDDAT     ;STORE EXPECTED
    799  006034  111115                         MOVB    (R1),(R5)       ;READ IMR
    800  006036  023715  001124                 CMP     $GDDAT,(R5)     ;SHOULD STILL BE ALL 1'S
    801  006042  001401                         BEQ     2$
    802  006044  104005                         ERROR   5               ;IMR REG ERROR
    803  006046  112710  000240           2$:     MOVB    #MISR,(R0)      ;LOAD MODE BITS FOR ISR
    804  006052  005037  001124                 CLR     $GDDAT          ;STORE EXPECTED
    805  006056  111115                         MOVB    (R1),(R5)       ;READ ISR
    806  006060  023715  001124                 CMP     $GDDAT,(R5)     ;ISR SHOULD BE CLEARED
    807  006064  001401                         BEQ     3$
    808  006066  104006                         ERROR   6               ;ISR REG ERROR
    809  006070  112710  000250           3$:     MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
    810  006074  111115                         MOVB    (R1),(R5)       ;READ IRR
    811  006076  023715  001124                 CMP     $GDDAT,(R5)     ;IRR SHOULD BE CLEARED
    812  006102  001401                         BEQ     4$
    813  006104  104003                         ERROR   3               ;IRR REG ERROR
    814  006106  105010           4$:     CLRB    (R0)            ;CHIP RESET GROUP 1
    815  006110  112710  000254                 MOVB    #MACR,(R0)      ;LOAD MODE BITS FOR ACR
    816  006114  111115                         MOVB    (R1),(R5)       ;READ ACR
    817  006116  023715  001124                 CMP     $GDDAT,(R5)     ;ACR SHOULD BE CLEARED
    818  006122  001401                         BEQ     5$
    819  006124  104004                         ERROR   4               ;ACR REG ERROR
    820  006126  005303           5$:     DEC     R3              ;TEST GROUPS 1 AND 2
    821  006130  001405                         BEQ     TST24           ;;BR IF BOTH GROUPS TESTED
    822  006132  013700  001352                 MOV     DRCSC,R0        ;GROUP 2 CONTROL PORT
    823  006136  013701  001356                 MOV     DRCSD,R1        ;GROUP 2 DATA PORT
    824  006142  000707                         BR      111$            ;TEST GROUP 2 ACR UNIQUENESS
    825
    826                                 ;;*****************************************************************
    (3)                                 ;*TEST 24       TEST GROUPS 1 AND 2 IMR UNIQUENESS
    (3)                                 ;;*****************************************************************
    (2)  006144  000004                 TST24:  SCOPE
    827                                 ;;GPA   JSR     PC,CLRCSR       ;CLEAR ALL CSRS
    828  006146  004737  013174                   JSR     PC,CLRIRR       ;; *** VDRCA1 CHANGES THIS ***  ;;GPA
    829  006152  013700  001342                   MOV     DRCSA,R0        ;CSRA ADDRESS
    830  006156  013701  001346                   MOV     DRCSB,R1        ;CSRB ADDRESS
    831  006162  012703  000002                   MOV     #2,R3           ;COUNTER FOR TWO GROUP TESTING
    832  006166  010137  001122           111$:   MOV     R1,$BDADR       ;STORE ADDRESS
```

```
  833  006172  012737  000252  001124          MOV     #252,$GDDAT     ;STORE EXPECTED
  834  006200  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
  835  006204  112710  000260                  MOVB    #PIMR,(R0)      ;PRESELECT IMR FOR WRITING
  836  006210  113711  001124                  MOVB    $GDDAT,(R1)     ;WRITE INTO DATA PORT
  837  006214  111115                          MOVB    (R1),(R5)       ;STORE DATA FOR COMPARE
  838  006216  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK IMR RESULTS
  839  006222  001401                          BEQ     1$              ;
  840  006224  104005                          ERROR   5               ;IMR ERROR
  841  006226  112710  000254          1$:     MOVB    #MACR,(R0)      ;CHANGE TO ACR REGISTER
  842  006232  005037  001124                  CLR     $GDDAT          ;STORE EXPECTED
  843  006236  111115                          MOVB    (R1),(R5)       ;READ ACR
  844  006240  023715  001124                  CMP     $GDDAT,(R5)     ;SHOULD STILL BE CLEARED
  845  006244  001401                          BEQ     2$
  846  006246  104004                          ERROR   4               ;ACR REG ERROR
  847  006250  112710  000240          2$:     MOVB    #MISR,(R0)      ;LOAD MODE BITS FOR ISR
  848  006254  111115                          MOVB    (R1),(R5)       ;READ ISR
  849  006256  023715  001124                  CMP     $GDDAT,(R5)     ;ISR SHOULD BE CLEARED
  850  006262  001401                          BEQ     3$
  951  006264  104006                          ERROR   6               ;ISR REG ERROR
  852  006266  112710  000250          3$:     MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  853  006272  111115                          MOVB    (R1),(R5)       ;READ IRR
  854  006274  023715  001124                  CMP     $GDDAT,(R5)     ;IRR SHOULD BE CLEARED
  855  006300  001401                          BEQ     4$
  856  006302  104003                          ERROR   3               ;IRR REG ERROR
  857  006304  105010          4$:     CLRB    (R0)            ;CHIP RESET GROUP 1
  858  006306  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
  859  006312  012737  000377  001124          MOV     #377,$GDDAT     ;STORE EXPECTED
  860  006320  111115                          MOVB    (R1),(R5)       ;READ IMR
  861  006322  023715  001124                  CMP     $GDDAT,(R5)     ;IMR SHOULD BE ALL ONES
  862  006326  001401                          BEQ     5$              ;DO NEXT GROUP
  863  006330  104005                          ERROR   5               ;IMR REG ERROR
  864  006332  005303          5$:     DEC     R3              ;DO GROUPS 1 AND GROUP 2
  865  006334  001405                          BEQ     TST25           ;;BR IF BOTH GROUPS TESTED
  866  006336  013700  001352                  MOV     DRCSC,R0        ;SET UP TO TEST GROUP 2
  867  006342  013701  001356                  MOV     DRCSD,R1        ;GROUP 2 DATA PORT
  868  006346  000707                          BR      111$            ;DO GROUP 2 IMR UNIQUENESS
  869
  870                          ;;*************************************************************************
  (3)                          ;*TEST 25       TEST GROUPS 1 AND 2 IRR UNIQUENESS
  (3)                          ;;*************************************************************************
  (2)  006350  000004          TST25:  SCOPE
  871  006352  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
  872  006356  013700  001342                  MOV     DRCSA,R0        ;CSRA ADDRESS
  873  006362  013701  001346                  MOV     DRCSB,R1        ;CSRB ADDRESS
  874  006366  012703  000002                  MOV     #2,R3           ;COUNTER FOR TESTING TWO GROUPS
  875  006372  004737  013174                  JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
  876  006376  010137  001122          111$:   MOV     R1,$BDADR       ;STORE ADDRESS
  877  006402  012737  000200  001124          MOV     #200,$GDDAT     ;STORE EXPECTED
  878  006410  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  879  006414  112710  000137                  MOVB    #137,(R0)       ;SET SINGLE IRR BIT7
  880  006420  111115                          MOVB    (R1),(R5)       ;STORE DATA FOR COMPARE
  881  006422  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK IRR RESULTS
  882  006426  001401                          BEQ     1$              ;
  883  006430  104003                          ERROR   3               ;IRR ERROR
  884  006432  112710  000244          1$:     MOVB    #MIMR,(R0)      ;CHANGE TO IMR REGISTER
  885  006436  012737  000377  001124          MOV     #377,$GDDAT     ;STORE EXPECTED
```

```
  886  006444  111115                          MOVB    (R1),(R5)       ;READ IMR
  887  006446  023715  001124                  CMP     $GDDAT,(R5)     ;SHOULD STILL BE ALL 1'S
  888  006452  001401                          BEQ     2$
  889  006454  104005                          ERROR   5               ;IMR REG ERROR
  890  006456  112710  000240          2$:     MOVB    #MISR,(R0)      ;LOAD MODE BITS FOR ISR
  891  006462  005037  001124                  CLR     $GDDAT          ;STORE EXPECTED
  892  006466  111115                          MOVB    (R1),(R5)       ;READ ISR
  893  006470  023715  001124                  CMP     $GDDAT,(R5)     ;ISR SHOULD BE CLEARED
  894  006474  001401                          BEQ     3$
  895  006476  104006                          ERROR   6               ;ISR REG ERROR
  896  006500  112710  000254          3$:     MOVB    #MACR,(R0)      ;LOAD MODE BITS FOR ACR
  897  006504  111115                          MOVB    (R1),(R5)       ;READ ACR
  898  006506  023715  001124                  CMP     $GDDAT,(R5)     ;ACR SHOULD BE CLEARED
  899  006512  001401                          BEQ     4$
  900  006514  104004                          ERROR   4               ;ACR REG ERROR
  901  006516  105010                  4$:     CLRB    (R0)            ;CHIP RESET GROUP 1
  902  006520  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  903  006524  111115                          MOVB    (R1),(R5)       ;READ IRR
  904  006526  023715  001124                  CMP     $GDDAT,(R5)     ;IRR SHOULD BE CLEARED
  905  006532  001401                          BEQ     5$
  906  006534  104003                          ERROR   3               ;IRR REG ERROR
  907  006536  005303                  5$:     DEC     R3              ;TEST GROUPS 1 AND 2
  908  006540  001405                          BEQ     TST26           ;;BR IF BOTH GROUPS TESTED
  909  006542  013700  001352                  MOV     DRCSC,R0        ;GROUP 2 CONTROL PORT
  910  006546  013701  001356                  MOV     DRCSD,R1        ;GROUP 2 DATA PORT
  911  006552  000711                          BR      111$            ;TEST GROUP 2 ACR UNIQUENESS
  912
  913
  914                                  ;;*******************************************************************
  (3)                                  ;*TEST 26        TEST GROUPS 1,2 ACR WITH PATTERNS
  (3)                                  ;;*******************************************************************
  (2)  006554  000004                  TST26:  SCOPE
  915  006556  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
  916  006562  012702  000002                  MOV     #2,R2           ;TEST TWO GROUPS
  917  006566  012737  006614  001110          MOV     #1$,$LPERR      ;SET UP LOOP RETURN
  918  006574  013700  001342                  MOV     DRCSA,R0        ;STORE CSRA ADDRESS
  919  006600  013701  001346                  MOV     DRCSB,R1        ;STORE CSRB ADDRESS
  920  006604  012703  013232          111$:   MOV     #BEGPAT,R3      ;SET UP PATTERN TABLE
  921  006610  010137  001122                  MOV     R1,$BDADR       ;STORE ADDRESS
  922  006614  112710  000254          1$:     MOVB    #MACR,(R0)      ;LOAD MODE BITS FOR ACR
  923  006620  112710  000300                  MOVB    #PACR,(R0)      ;PRESELECT ACR FOR WRITING
  924  006624  111337  001124                  MOVB    (R3),$GDDAT     ;STORE EXPECTED
  925  006630  111311                          MOVB    (R3),(R1)       ;WRITE INTO ACR
  926  006632  111115                          MOVB    (R1),(R5)       ;READ OUT OF ACR
  927  006634  023715  001124                  CMP     $GDDAT,(R5)
  928  006640  001401                          BEQ     2$
  929  006642  104004                          ERROR   4               ;ACR REGISTER ERROR
  930  006644  005723                  2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
  931  006646  020327  013532                  CMP     R3,#ENDDAT      ;CHECK FOR TABLE END
  932  006652  001360                          BNE     1$              ;W/R NEXT PATTERN
  933  006654  005302                          DEC     R2              ;FINISHED BOTH GROUPS?
  934  006656  001405                          BEQ     TST27           ;;BR IF EQUAL
  935  006660  013700  001352                  MOV     DRCSC,R0
  936  006664  013701  001356                  MOV     DRCSD,R1
  937  006670  000745                          BR      111$            ;DO NEXT GROUP
  938
```

```
939                                     ;;*****************************************************************
(3)                                     ;*TEST 27       TEST GROUPS 1,2 IMR WITH PATTERNS
(3)                                     ;;*****************************************************************
(2)  006672  000004                     TST27:  SCOPE
940  006674  004737  013124                       JSR     PC,CLRCSR       ;CLEAR ALL CSRS
941  006700  012702  000002                       MOV     #2,R2           ;TEST TWO GROUPS
942  006704  012737  006732  001110              MOV     #1$,$LPERR      ;SET UP LOOP RETURN
943  006712  013700  001342                       MOV     DRCSA,R0        ;STORE CSRA ADDRESS
944  006716  013701  001346                       MOV     DRCSB,R1        ;STORE CSRB ADDRESS
945  006722  012703  013232              111$:    MOV     #BEGPAT,R3      ;SET UP PATTERN TABLE
946  006726  010137  001122                       MOV     R1,$BDADR       ;STORE ADDRESS
947  006732  112710  000244              1$:      MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
948  006736  112710  000260                       MOVB    #PIMR,(R0)      ;PRESELECT IMR FOR WRITING
949  006742  111337  001124                       MOVB    (R3),$GDDAT     ;STORE EXPECTED
950  006746  111311                             MOVB    (R3),(R1)       ;WRITE INTO IMR
951  006750  111115                             MOVB    (R1),(R5)       ;READ OUT OF IMR
952  006752  023715  001124                     CMP     $GDDAT,(R5)
953  006756  001401                             BEQ     2$
954  006760  104005                             ERROR   5               ;IMR REGISTER ERROR
955  006762  005723                     2$:     TST     (R3)+           ;INC FOR NEXT PATTERN
956  006764  020327  013532                     CMP     R3,#ENDDAT      ;CHECK FOR TABLE END
957  006770  001360                             BNE     1$              ;W/R NEXT PATTERN
958  006772  005302                             DEC     R2              ;FINISHED BOTH GROUPS?
959  006774  001405                             BEQ     TST30           ;;BR IF EQUAL
960  006776  013700  001352                     MOV     DRCSC,R0
961  007002  013701  001356                     MOV     DRCSD,R1
962  007006  000745                             BR      111$            ;DO NEXT GROUP
963
964                                     ;;*****************************************************************
(3)                                     ;*TEST 30       TEST GROUP 1,2 CLEAR IMR INSTR.
(3)                                     ;;*****************************************************************
(2)  007010  000004                     TST30:  SCOPE
965  007012  004737  013124                       JSR     PC,CLRCSR       ;CLEAR ALL CSRS
966  007016  012704  000002                       MOV     #2,R4           ;COUNTER FOR TWO GROUPS
967  007022  013700  001342                       MOV     DRCSA,R0        ;CSRA ADDRESS
968  007026  013701  001346                       MOV     DRCSB,R1        ;CSRB ADDRESS
969  007032  010137  001122              111$:    MOV     R1,$BDADR       ;STORE ADDRESS
970  007036  005037  001124                       CLR     $GDDAT          ;EXPECTED DATA
971  007042  112710  000244                       MOVB    #M.MR,(R0)      ;LOAD MODE BITS TO READ IMR
972  007046  112710  000040                       MOVB    #CIMR,(R0)      ;CLEAR IMR COMMAND
973  007052  111115                             MOVB    (R1),(R5)       ;READ DATA PORT
974  007054  023715  001124                     CMP     $GDDAT,(R5)
975  007060  001401                             BEQ     1$
976  007062  104005                             ERROR   5               ;ERROR ,IMR SHOULD BE CLEARED
977  007064  005304                     1$:     DEC     R4              ;FINISHED BOTH GROUPS?
978  007066  001405                             BEQ     TST31           ;;BR IF BOTH GROUPS TESTED
979  007070  013700  001352                     MOV     DRCSC,R0        ;SETUP FOR GROUP 2
980  007074  013701  001356                     MOV     DRCSD,R1
981  007100  000754                             BR      111$            ;DO IMR TEST WITH GROUP 2
982
983                                     ;;*****************************************************************
(3)                                     ;*TEST 31       TEST GROUP 1,2 SET IMR INSTR.
(3)                                     ;;*****************************************************************
(2)  007102  000004                     TST31:  SCOPE
984  007104  004737  013124                       JSR     PC,CLRCSR       ;CLEAR ALL CSRS
985  007110  012704  000002                       MOV     #2,R4           ;COUNTER FOR TWO GROUPS
```

```
  986  007114  013700  001342                    MOV     DRCSA,R0        ;CSRA ADDRESS
  987  007120  013701  001346                    MOV     DRCSB,R1        ;CSRB ADDRESS
  988  007124  010137  001122            111$:   MOV     R1,$BDADR       ;STORE ADDRESS
  989  007130  012737  000377  001124            MOV     #377,$GDDAT     ;EXPECTED DATA
  990  007136  112710  000244                    MOVB    #MIMR,(R0)      ;LOAD MODE BITS TO READ IMR
  991  007142  112710  000040                    MOVB    #CIMR,(R0)      ;CLEAR IMR
  992  007146  112710  000060                    MOVB    #SIMR,(R0)      ;SET IMR COMMAND
  993  007152  111115                            MOVB    (R1),(R5)       ;READ DATA PORT
  994  007154  023715  001124                    CMP     $GDDAT,(R5)
  995  007160  001401                            BEQ     1$
  996  007162  104005                            ERROR   5               ;ERROR ,IMR SHOULD BE SET
  997  007164  005304                    1$:     DEC     R4              ;FINISHED BOTH GROUPS?
  998  007166  001405                            BEQ     TST32           ;;BR IF BOTH GROUPS TESTED
  999  007170  013700  001352                    MOV     DRCSC,R0        ;SETUP FOR GROUP 2
 1000  007174  013701  001356                    MOV     DRCSD,R1
 1001  007200  000751                            BR      111$            ;DO IMR TEST WITH GROUP 2
 1002
 1003                                    ;;*******************************************************************
  (3)                                    ;*TEST 32       TEST GROUP 1,2  CLEAR SINGLE IMR BIT INSTR.
  (3)                                    ;;*******************************************************************
  (2)  007202  000004                    TST32:  SCOPE
 1004  007204  004737  013124                    JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1005  007210  012704  000002                    MOV     #2,R4           ;GROUP COUNTER
 1006  007214  012737  007252  001110            MOV     #1$,$LPERR      ;SCOPE RETURN ADDRESS
 1007  007222  013700  001342                    MOV     DRCSA,R0        ;CSRA ADDRESS
 1008  007226  013701  001346                    MOV     DRCSB,R1        ;CSRB ADDRESS
 1009  007232  010137  001122            111$:   MOV     R1,$BDADR       ;STORE ADDRESS
 1010  007236  012703  013376                    MOV     #BGCHP4,R3      ;GOOD DATA PATTERN TABLE
 1011  007242  112710  000244                    MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
 1012  007246  012702  000050                    MOV     #CSIMR,R2       ;CLEAR SINGLE IMR BIT VALUE
 1013  007252  111337  001124            1$:     MOVB    (R3),$GDDAT     ;STORE EXPECTED
 1014  007256  112710  000060                    MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
 1015  007262  110210                            MOVB    R2,(R0)         ;CLEAR SINGLE IMR BIT
 1016  007264  111115                            MOVB    (R1),(R5)       ;READ DATA PORT
 1017  007266  023715  001124                    CMP     $GDDAT,(R5)
 1018  007272  001401                            BEQ     2$
 1019  007274  104005                            ERROR   5               ;IMR REG ERROR
 1020  007276  005723                    2$:     TST     (R3)+           ;INC EXPECTED DATA TABLE
 1021  007300  020327  013416                    CMP     R3,#EDCHP4      ;CHECK FOR END
 1022  007304  001402                            BEQ     3$
 1023  007306  005202                            INC     R2              ;SET UP TO CLEAR NEXT IMR BIT
 1024  007310  000760                            BR      1$              ;CLEAR NEXT IMR BIT
 1025  007312  005304                    3$:     DEC     R4              ;DO GROUPS 1 AND 2
 1026  007314  001405                            BEQ     TST33           ;;BR IF BOTH GROUPS TESTED
 1027  007316  013700  001352                    MOV     DRCSC,R0        ;CSRC ADDRESS
 1028  007322  013701  001356                    MOV     DRCSD,R1        ;CSRD ADDRESS
 1029  007326  000741                            BR      111$            ;DO GROUP 2
 1030
 1031
 1032                                    ;;*******************************************************************
  (3)                                    ;*TEST 33       TEST GROUP 1,2  SET SINGLE IMR BIT INSTR.
  (3)                                    ;;*******************************************************************
  (2)  007330  000004                    TST33:  SCOPE
 1033  007332  004737  013124                    JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1034  007336  012704  000002                    MOV     #2,R4           ;GROUP COUNTER
 1035  007342  012737  007400  001110            MOV     #1$,$LPERR      ;SCOPE RETURN ADDRESS
```

```
 1036  007350  013700  001342                MOV     DRCSA,R0        ;CSRA ADDRESS
 1037  007354  013701  001346                MOV     DRCSB,R1        ;CSRB ADDRESS
 1038  007360  010137  001122        111$:   MOV     R1,$BDADR       ;STORE ADDRESS
 1039  007364  012703  013334                MOV     #BGCHP3,R3      ;GOOD DATA PATTERN TABLE
 1040  007370  112710  000244                MOVB    #MIMR,(R0)      ;LOAD MODE BITS FOR IMR
 1041  007374  012702  000070                MOV     #SSIMR,R2       ;SET SINGLE IMR BIT VALUE
 1042  007400  111337  001124        1$:     MOVB    (R3),$GDDAT     ;STORE EXPECTED
 1043  007404  112710  000040                MOVB    #CIMR,(R0)      ;CLEAR ALL IMR BITS
 1044  007410  110210                        MOVB    R2,(R0)         ;SET SINGLE IMR BIT
 1045  007412  111115                        MOVB    (R1),(R5)       ;READ DATA PORT
 1046  007414  023715  001124                CMP     $GDDAT,(R5)
 1047  007420  001401                        BEQ     2$
 1048  007422  104005                        ERROR   5               ;IMR REG ERROR
 1049  007424  005723                2$:     TST     (R3)+           ;INC EXPECTED DATA TABLE
 1050  007426  020327  013354                CMP     R3,#EDCHP3      ;CHECK FOR END
 1051  007432  001402                        BEQ     3$
 1052  007434  005202                        INC     R2              ;SET UP TO SET NEXT IMR BIT
 1053  007436  000760                        BR      1$              ;SET NEXT IMR BIT
 1054  007440  005304                3$:     DEC     R4              ;DO GROUPS 1 AND 2
 1055  007442  001405                        BEQ     TST34           ;;BR IF BOTH GROUPS TESTED
 1056  007444  013700  001352                MOV     DRCSC,R0        ;CSRC ADDRESS
 1057  007450  013701  001356                MOV     DRCSD,R1        ;CSRD ADDRESS
 1058  007454  000741                        BR      111$            ;DO GROUP 2
 1059
 1060                                ;;*******************************************************************
  (3)                               ;*TEST 34       TEST GROUP 1,2 SET IRR INSTR.
  (3)                               ;;*******************************************************************
  (2)  007456  000004               TST34:  SCOPE
 1061  007460  004737  013124                JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1062  007464  012704  000002                MOV     #2,R4           ;COUNTER FOR TWO GROUPS
 1063  007470  013700  001342                MOV     DRCSA,R0        ;CSRA ADDRESS
 1064  007474  013701  001346                MOV     DRCSB,R1        ;CSRB ADDRESS
 1065  007500  004737  013174                JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
 1066  007504  010137  001122        111$:   MOV     R1,$BDADR       ;STORE ADDRESS
 1067  007510  012737  000377  001124        MOV     #377,$GDDAT     ;EXPECTED DATA
 1068  007516  112710  000250                MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
 1069  007522  112710  000120                MOVB    #SIRR,(R0)      ;SET IRR COMMAND
 1070  007526  111115                        MOVB    (R1),(R5)       ;READ DATA PORT
 1071  007530  023715  0011_4                CMP     $GDDAT,(R5)
 1072  007534  001401                        BEQ     1$
 1073  007536  104003                        ERROR   3               ;ERROR ,IRR SHOULD BE SET
 1074  007540  005304                1$:     DEC     R4              ;FINISHED BOTH GROUPS?
 1075  007542  001405                        BEQ     TST35           ;;BR IF BOTH GROUPS TESTED
 1076  007544  013700  001352                MOV     DRCSC,R0        ;SETUP FOR GROUP 2
 1077  007550  013701  001356                MOV     DRCSD,R1
 1078  007554  000753                        BR      111$            ;DO IRR TEST WITH GROUP 2
 1079
 1080
 1081                                ;;*******************************************************************
  (3)                               ;*TEST 35       TEST GROUP 1,2 CLEAR IRR INSTR.
  (3)                               ;;*******************************************************************
  (2)  007556  000004               TST35:  SCOPE
 1082  007560  004737  013124                JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1083  007564  012704  000002                MOV     #2,R4           ;COUNTER FOR TWO GROUPS
 1084  007570  013700  001342                MOV     DRCSA,R0        ;CSRA ADDRESS
 1085  007574  013701  001346                MOV     DRCSB,R1        ;CSRB ADDRESS
```

```
1086  007600  004737  013174            JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
1087  007604  010137  001122    111$:   MOV     R1,$BDADR       ;STORE ADDRESS
1088  007610  005037  001124            CLR     $GDDAT          ;EXPECTED DATA
1089  007614  112710  000250            MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
1090  007620  112710  000120            MOVB    #SIRR,(R0)      ;SET IRR BITS
1091  007624  112710  000100            MOVB    #CIRR,(R0)      ;CLEAR IRR COMMAND
1092  007630  111115                    MOVB    (R1),(R5)       ;READ DATA PORT
1093  007632  023715  001124            CMP     $GDDAT,(R5)
1094  007636  001401                    BEQ     1$
1095  007640  104003                    ERROR   3               ;ERROR ,IRR SHOULD BE CLEARED
1096  007642  005304            1$:     DEC     R4              ;FINISHED BOTH GROUPS?
1097  007644  001405                    BEQ     TST36           ;;BR IF BOTH GROUPS TESTED
1098  007646  013700  001352            MOV     DRCSC,R0        ;SETUP FOR GROUP 2
1099  007652  013701  001356            MOV     DRCSD,R1
1100  007656  000752                    BR      111$            ;DO IRR TEST WITH GROUP 2
1101
1102                                    ;;*************************************************************************
 (3)                                    ;*TEST 36        TEST GROUP 1,2  CLEAR SINGLE IRR BIT INSTR.
 (3)                                    ;;*************************************************************************
 (2)  007660  000004            TST36:  SCOPE
1103  007662  004737  013124            JSR     PC,CLRCSR       ;CLEAR ALL CSRS
1104  007666  012704  000002            MOV     #2,R4           ;GROUP COUNTER
1105  007672  012737  007734  001110    MOV     #1$,$LPERR      ;SCOPE RETURN ADDRESS
1106  007700  013700  001342            MOV     DRCSA,R0        ;CSRA ADDRESS
1107  007704  013701  001346            MOV     DRCSB,R1        ;CSRB ADDRESS
1108  007710  004737  013174            JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
1109  007714  010137  001122    111$:   MOV     R1,$BDADR       ;STORE ADDRESS
1110  007720  012703  013376            MOV     #BGCHP4,R3      ;GOOD DATA PATTERN TABLE
1111  007724  112710  000250            MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
1112  007730  012702  000110            MOV     #CSIRR,R2       ;CLEAR SINGLE IRR BIT VALUE
1113  007734  111337  001124    1$:     MOVB    (R3),$GDDAT     ;STORE EXPECTED
1114  007740  112710  000120            MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
1115  007744  110210                    MOVB    R2,(R0)         ;CLEAR SINGLE IRR BIT
1116  007746  111115                    MOVB    (R1),(R5)       ;READ DATA PORT
1117  007750  023715  001124            CMP     $GDDAT,(R5)
1118  007754  001401                    BEQ     2$
1119  007756  104003                    ERROR   3               ;IRR REG ERROR
1120  007760  005723            2$:     TST     (R3)+           ;INC EXPECTED DATA TABLE
1121  007762  020327  013416            CMP     R3,#EDCHP4      ;CHECK FOR END
1122  007766  001402                    BEQ     3$
1123  007770  005202                    INC     R2              ;SET UP TO CLEAR NEXT IRR BIT
1124  007772  000760                    BR      1$              ;CLEAR NEXT IRR BIT
1125  007774  005304            3$:     DEC     R4              ;DO GROUPS 1 AND 2
1126  007776  001405                    BEQ     TST37           ;;BR IF BOTH GROUPS TESTED
1127  010000  013700  001352            MOV     DRCSC,R0        ;CSRC ADDRESS
1128  010004  013701  001356            MOV     DRCSD,R1        ;CSRD ADDRESS
1129  010010  000741                    BR      111$            ;DO GROUP 2
1130
1131                                    ;;*************************************************************************
 (3)                                    ;*TEST 37        TEST GROUP 1,2  SET SINGLE IRR BIT INSTR.
 (3)                                    ;;*************************************************************************
 (2)  010012  000004            TST37:  SCOPE
1132  010014  004737  013124            JSR     PC,CLRCSR       ;CLEAR ALL CSRS
1133  010020  012704  000002            MOV     #2,R4           ;GROUP COUNTER
1134  010024  012737  010066  001110    MOV     #1$,$LPERR      ;SCOPE RETURN ADDRESS
1135  010032  013700  001342            MOV     DRCSA,R0        ;CSRA ADDRESS
```

```
  1136  010036  013701  001346                MOV     DRCSB,R1        ;CSPB ADDRESS
  1137  010042  004737  013174                JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
  1138  010046  010137  001122        111$:   MOV     R1,$BDADR       ;STORE ADDRESS
  1139  010052  012703  013334                MOV     #BGCHP3,R3      ;GOOD DATA PATTERN TABLE
  1140  010056  112710  000250                MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  1141  010062  012702  000130                MOV     #SSIRR,R2       ;SET SINGLE IRR BIT VALUE
  1142  010066  111337  001124        1$:     MOVB    (R3),$GDDAT     ;STORE EXPECTED
  1143  010072  112710  000100                MOVB    #CIRR,(R0)      ;CLEAR ALL IRR BITS
  1144  010076  110210                        MOVB    R2,(R0)         ;SET SINGLE IRR BIT
  1145  010100  111115                        MOVB    (R1),(R5)       ;READ DATA PORT
  1146  010102  023715  001124                CMP     $GDDAT,(R5)
  1147  010106  001401                        BEQ     2$
  1148  010110  104003                        ERROR   3               ;IRR REG ERROR
  1149  010112  005723                2$:     TST     (R3)+           ;INC EXPECTED DATA TABLE
  1150  010114  020327  013354                CMP     R3,#EDCHP3      ;CHECK FOR END
  1151  010120  001402                        BEQ     3$
  1152  010122  005202                        INC     R2              ;SET UP TO SET NEXT IRR BIT
  1153  010124  000760                        BR      1$              ;SET NEXT IRR BIT
  1154  010126  005304                3$:     DEC     R4              ;DO GROUPS 1 AND 2
  1155  010130  001405                        BEQ     TST40           ;;BR IF BOTH GROUPS TESTED
  1156  010132  013700  001352                MOV     DRCSC,R0        ;CSRC ADDRESS
  1157  010136  013701  001356                MOV     DRCSD,R1        ;CSRD ADDRESS
  1158  010142  000741                        BR      111$            ;DO GROUP 2
  1159
  1160                                ;;*******************************************************************
   (3)                                ;*TEST 40       TEST GROUP 1,2 CLEAR IRR+IMR INSTR.
   (3)                                ;;*******************************************************************
   (2)  010144  000004                TST40:  SCOPE
  1161  010146  004737  013124                JSR     PC,CLRCSR       ;CLEAR ALL CSRS
  1162  010152  012704  000002                MOV     #2,R4           ;COUNTER FOR TWO GROUPS
  1163  010156  013700  001342                MOV     DRCSA,R0        ;CSRA ADDRESS
  1164  010162  013701  001346                MOV     DRCSB,R1        ;CSRB ADDRESS
  1165  010166  004737  013174                JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
  1166  010172  010137  001122        111$:   MOV     R1,$BDADR       ;STORE ADDRESS
  1167  010176  005037  001124                CLR     $GDDAT          ;EXPECTED DATA
  1168  010202  112710  000250                MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
  1169  010206  112710  000120                MOVB    #SIRR,(R0)      ;SET IRR BITS
  1170  010212  112710  000060                MOVB    #SIMR,(R0)      ;SET IMR BITS
  1171  010216  112710  000020                MOVB    #CIRMR,(R0)     ;CLEAR IRR+IMR COMMAND
  1172  010222  111115                        MOVB    (R1),(R5)       ;READ DATA PORT FOR IRR
  1173  010224  023715  001124                CMP     $GDDAT,(R5)
  1174  010230  001401                        BEQ     1$
  1175  010232  104003                        ERROR   3               ;ERROR ,IRR SHOULD BE CLEARED
  1176  010234  112710  000244        1$:     MOVB    #MIMR,(R0)      ;LOAD MODE BITS TO READ IMR
  1177  010240  111115                        MOVB    (R1),(R5)       ;READ IMR REG
  1178  010242  023715  001124                CMP     $GDDAT,(R5)
  1179  010246  001401                        BEQ     2$
  1180  010250  104005                        ERROR   5               ;IRR+IMR COMMAND DID NOT CLEAR IMR
  1181  010252  005304                2$:     DEC     R4              ;FINISHED BOTH GROUPS?
  1182  010254  001405                        BEQ     TST41           ;;BR IF BOTH GROUPS TESTED
  1183  010256  013700  001352                MOV     DRCSC,R0        ;SETUP FOR GROUP 2
  1184  010262  013701  001356                MOV     DRCSD,R1
  1185  010266  000741                        BR      111$            ;DO IRR+IMR TEST WITH GROUP 2
  1186
  1187                                ;;*******************************************************************
   (3)                                ;*TEST 41       TEST GROUP 1,2  CLEAR SINGLE IRR+IMR BIT INSTR.
```

```
  (3)                                   ;;*****************************************************************
  (2)  010270  000004                   TST41:  SCOPE
1188   010272  004737  013124                   JSR     PC,CLRCSR       ;CLEAR ALL CSRS
1189   010276  012704  000002                   MOV     #2,R4           ;GROUP COUNTER
1190   010302  012737  010340  001110           MOV     #1$,$LPERR      ;SCOPE RETURN ADDRESS
1191   010310  013700  001342                   MOV     DRCSA,R0        ;CSRA ADDRESS
1192   010314  013701  001346                   MOV     DRCSB,R1        ;CSRB ADDRESS
1193   010320  004737  013174                   JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
1194   010324  010137  001122           111$:   MOV     R1,$BDADR       ;STORE ADDRESS
1195   010330  012703  013376                   MOV     #BGCHP4,R3      ;GOOD DATA PATTERN TABLE
1196   010334  012702  000030                   MOV     #CSIRMR,R2      ;CLEAR SINGLE IRR+IMR BIT VALUE
1197   010340  111337  001124           1$:     MOVB    (R3),$GDDAT     ;STORE EXPECTED
1198   010344  112710  000250                   MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
1199   010350  112710  000120                   MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
1200   010354  112710  000060                   MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
1201   010360  110210                           MOVB    R2,(R0)         ;CLEAR SINGLE IRR+IMR BIT
1202   010362  111115                           MOVB    (R1),(R5)       ;READ DATA PORT FOR IRR
1203   010364  023715  001124                   CMP     $GDDAT,(R5)
1204   010370  001401                           BEQ     2$
1205   010372  104003                           ERROR   3               ;IRR REG ERROR
1206   010374  112710  000244           2$:     MOVB    #MIMR,(R0)      ;SET UP TO READ IMR
1207   010400  111115                           MOVB    (R1),(R5)       ;READ IMR REGISTER
1208   010402  023715  001124                   CMP     $GDDAT,(R5)
1209   010406  001401                           BEQ     3$
1210   010410  104005                           ERROR   5               ;IMR REG ERROR
1211   010412  005723                   3$:     TST     (R3)+           ;INC EXPECTED DATA TABLE
1212   010414  020327  013416                   CMP     R3,#EDCHP4      ;CHECK FOR END
1213   010420  001402                           BEQ     4$
1214   010422  005202                           INC     R2              ;SET UP TO CLEAR NEXT IRR+IMR BIT
1215   010424  000745                           BR      1$              ;CLEAR NEXT IRR+IMR BIT
1216   010426  005304                   4$:     DEC     R4              ;DO GROUPS 1 AND 2
1217   010430  001405                           BEQ     TST42           ;;BR IF BOTH GROUPS TESTED
1218   010432  013700  001352                   MOV     DRCSC,R0        ;CSRC ADDRESS
1219   010436  013701  001356                   MOV     DRCSD,R1        ;CSRD ADDRESS
1220   010442  000730                           BR      111$            ;DO GROUP 2
1221
1222                                    ;;*****************************************************************
                                        ;*TEST 42        TEST GROUPS 1,2 FOR GROUP UNIQUENESS
                                        ;;*****************************************************************
  (3)                                   TST42:  SCOPE
  (3)
  (2)  010444  000004
1223   010446  004737  013124                   JSR     PC,CLRCSR       ;CLEAR ALL CSRS
1224                                                                     ;CSRA = R0
1225                                                                     ;CSRB = R1
1226                                                                     ;CSRC = R2
1227                                                                     ;CSRD = R3
1228   010452  012704  000002                   MOV     #2,R4           ;COUNTER FOR TESTING TWO GROUPS
1229   010456  004737  013174                   JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
1230   010462  010337  001122           111$:   MOV     R3,$BDADR       ;STORE ADDRESS
1231   010466  112710  000300                   MOVB    #PACR,(R0)      ;PRESELECT ACR FOR WRITING
1232   010472  112711  000377                   MOVB    #377,(R1)       ;WRITE INTO DATA PORT FOR ACR
1233   010476  112710  000120                   MOVB    #SIRR,(R0)      ;SET IRR TO ALL 1'S
1234   010502  112710  000040                   MOVB    #CIMR,(R0)      ;CLEAR IMR
1235   010506  112712  000254                   MOVB    #MACR,(R2)      ;LOAD MODE TO READ ACR
1236   010512  005037  001124                   CLR     $GDDAT          ;EXPECTED
1237   010516  111315                           MOVB    (R3),(R5)       ;STORE DATA FOR COMPARE
1238   010520  023715  001124                   CMP     $GDDAT,(R5)     ;CHECK ACR RESULTS
```

```
 1239  010524  001401                          BEQ     1$              ;
 1240  010526  104004                          ERROR   4               ;ERROR,OTHER GROUP ACR SHOULD BE CLEARED
 1241  010530  112712  000244          1$:     MOVB    #MIMR,(R2)      ;CHANGE TO IMR REGISTER
 1242  010534  012737  000377  001124          MOV     #377,$GDDAT     ;STORE EXPECTED
 1243  010542  111315                          MOVB    (R3),(R5)       ;READ IMR
 1244  010544  023715  001124                  CMP     $GDDAT,(R5)     ;SHOULD  BE ALL 1'S
 1245  010550  001401                          BEQ     2$
 1246  010552  104005                          ERROR   5               ;ERROR,OTHER GROUP IMR SHOULD BE SET
 1247  010554  112712  000240          2$:     MOVB    #MISR,(R2)      ;LOAD MODE BITS FOR ISR
 1248  010560  005037  001124                  CLR     $GDDAT          ;STORE EXPECTED
 1249  010564  111315                          MOVB    (R3),(R5)       ;READ ISR
 1250  010566  023715  001124                  CMP     $GDDAT,(R5)     ;ISR SHOULD BE CLEARED
 1251  010572  001401                          BEQ     3$
 1252  010574  104006                          ERROR   6               ;ISR REG ERROR
 1253  010576  112712  000250          3$:     MOVB    #MIRR,(R2)      ;LOAD MODE BITS FOR IRR
 1254  010602  111315                          MOVB    (R3),(R5)       ;READ IRR
 1255  010604  023715  001124                  CMP     $GDDAT,(R5)     ;IRR SHOULD BE CLEARED
 1256  010610  001401                          BEQ     4$
 1257  010612  104003                          ERROR   3               ;ERROR,OTHER GROUP IRR SHOULD BE CLEARED
 1258  010614  005304                  4$:     DEC     R4              ;TEST GROUPS 1 AND 2
 1259  010616  001415                          BEQ     TST43           ;;BR IF BOTH GROUPS TESTED
 1260  010620  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1261  010624  013700  001352                  MOV     DRCSC,R0        ;GROUP 2 CONTROL PORT
 1262  010630  013701  001356                  MOV     DRCSD,R1        ;GROUP 2 DATA PORT
 1263  010634  013702  001342                  MOV     DRCSA,R2        ;GROUP 1 CONTROL PORT
 1264  010640  013703  001346                  MOV     DRCSB,R3        ;GROUP 1 DATA PORT
 1265  010644  004737  013174                  JSR     PC,CLRIRR       ;CLEAR IRR REGS WITH CHIP RESET
 1266  010650  000704                          BR      111$            ;TEST GROUP 2 ACR UNIQUENESS
 1267
 1268                          ;;*****************************************************************
  (3)                          ;*TEST 43       TEST STATUS BITS GINT,S2,S1,S0,GP1,2
  (3)                          ;;*****************************************************************
  (2)  010652  000004          TST43:  SCOPE
 1269  010654  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1270  010660  013700  001342                  MOV     DRCSA,R0
 1271  010664  013701  001346                  MOV     DRCSB,R1
 1272  010670  012703  000002                  MOV     #2,R3           ;DO TWO GROUPS
 1273  010674  012704  000120          111$:   MOV     #120,R4         ;EXPECTED STATUS BITS
 1274  010700  005077  170436                  CLR     @DRCSA          ;INIT CSRA
 1275  010704  005077  170442                  CLR     @DRCSC          ;INIT CSRC
 1276  010710  012702  013234                  MOV     #BGPAT1,R2      ;EXPECTED IRR PATTERN
 1277  010714  112760  000001  000001          MOVB    #BIT0,1(R0)     ;CSR TO OUTPUT MODE
 1278  010722  112777  000204  170412          MOVB    #204,@DRCSA     ;POLLED MODE FOR CSRA,GROUP 1
 1279  010730  112777  000204  170414          MOVB    #204,@DRCSC     ;POLLED MODE FOR CSRC,GROUP 2
 1280  010736  112710  000020                  MOVB    #CIRMR,(R0)     ;CLEAR IMR + IRR
 1281  010742  012737  000070  001372          MOV     #SSIMR,IMRLOC   ;STORE CODE FOR SINGLE IMR
 1282  010750  012737  000130  001376          MOV     #SSIRR,IRRLOC   ;STORE CODE FOR SINGLE IRR
 1283  010756  010037  001122          112$:   MOV     R0,$BDADR       ;CSR CHIP COMMAND  ADDRESS
 1284  010762  010437  001124                  MOV     R4,$GDDAT       ;EXPECTED DATA
 1285  010766  113710  001376                  MOVB    IRRLOC,(R0)     ;SET SINGLE IRR BIT
 1286  010772  111015                          MOVB    (R0),(R5)       ;CHECK STATUS
 1287  010774  042715  000007                  BIC     #7,(R5)         ;; *** VDRCA1 ADDS THIS ***     ;;GPA
 1288  011000  023715  001124                  CMP     $GDDAT,(R5)
 1289  011004  001401                          BEQ     1$
 1290  011006  104007                          ERROR   7               ;CHIP STATUS ERROR
 1291  011010  005037  001124          1$:     CLR     $GDDAT
```

```
 1292  011014  010137  001122                  MOV     R1,$BDADR       ;CSR CHIP DATA ADDRESS
 1293  011020  111237  001124                  MOVB    (R2),$GDDAT
 1294  011024  112710  000250                  MOVB    #MIMR,(R0)      ;READ IRR
 1295  011030  111115                          MOVB    (R1),(R5)
 1296  011032  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK IRR
 1297  011036  001401                          BEQ     2$
 1298  011040  104003                          ERROR   3               ;IRR ERROR
 1299  011042  010037  001122          2$:     MOV     R0,$BDADR
 1300  011046  113710  001372                  MOVB    IMRLOC,(R0)     ;SET IMR BIT
 1301  011052  012737  000320  001124          MOV     #320,$GDDAT     ;CSR EXPECTED DATA
 1302  011060  111015                          MOVB    (R0),(R5)
 1303  011062  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
 1304  011066  023715  001124                  CMP     $GDDAT,(R5)
 1305  011072  001401                          BEQ     3$
 1306  011074  104007                          ERROR   7               ;CHIP STATUS ERROR
 1307  011076  010137  001122          3$:     MOV     R1,$BDADR
 1308  011102  011237  001124                  MOV     (R2),$GDDAT     ;EXPECTED DATA
 1309  011106  112710  000244                  MOVB    #MIMR,(R0)      ;READ IMR BITS
 1310  011112  111115                          MOVB    (R1),(R5)       ;SAVE IMR READ
 1311  011114  023715  001124                  CMP     $GDDAT,(R5)
 1312  011120  001401                          BEQ     4$
 1313  011122  104005                          ERROR   5               ;IMR ERROR
 1314  011124  005722                  4$:     TST     (R2)+           ;NEXT EXPECTED FOR IMR + IRR
 1315  011126  020227  013254                  CMP     R2,#EDCHP1      ;CHECK FOR END
 1316  011132  001407                          BEQ     5$
 1317  011134  005237  001376                  INC     IRRLOC          ;NEXT IRR BIT
 1318  011140  005237  001372                  INC     IMRLOC          ;NEXT IMR BIT
 1319                           ;;GPA          INC     P4              ;INDEX EXPECTED STATUS
 1320  011144  000240                          NOP                     ;; *** VDRCA1 DELETES INC R4 *** ;;GPA
 1321  011146  000137  010756                  JMP     112$            ;DO NEXT STATUS CHECK
 1322  011152  005303                  5$:     DEC     R3              ;FINISHED BOTH GROUPS?
 1323  011154  001406                          BEQ     TST44           ;;BR IF EQUAL
 1324  011156  013700  001352                  MOV     DRCSC,R0
 1325  011162  013701  001356                  MOV     DRCSD,R1
 132_  011166  000137  010674                  JMP     111$            ;DO NEXT GROUP
 1327
 1328
 1329                           ;;*****************************************************************
  (3)                           ;*TEST 44       TEST POLLED MODE;CSRS A,B=OUT C,D=IN,ACTIVE LOW
  (3)                           ;;*****************************************************************
  (2)  011172  000004           TST44:  SCOPE
 1330  011174  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1331                                                                  ;R0 = CSRA-GROUP 1 CONTROL
 1332                                                                  ;R1 = CSRB-GROUP 1 DATA
 1333                                                                  ;R2 = CSRC-GROUP 2 CONTROL
 1334                                                                  ;R3 = CSRD-GROUP 2 DATA
 1335  011200  012737  011242  001110          MOV     #1$,$LPERR      ;SET FOR SCOPE RETURN
 1336  011206  012704  013232                  MOV     #BEGPAT,R4      ;START OF PATTERN TABLE
 1337  011212  112760  000001  000001          MOVB    #BIT0,1(R0)     ;SET CSRA TO OUTPUT MODE
 1338  011220  112761  000001  000001          MOVB    #BIT0,1(R1)     ;SET CSRB TO OUTPUT MODE
 1339
 1340  011226  105010                          CLRB    (R0)            ;CHIP RESET GROUP 1 CSRA
 1341  011230  105012                          CLRB    (R2)            ;CHIP RESET GROUP 2 CSRC
 1342  011232  112710  000204                  MOVB    #204,(R0)       ;LOAD MODE BITS FOR POLLED MODE,GR 1
 1343  011236  112712  000204                  MOVB    #204,(R2)       ;LOAD MODE BITS FOR POLLED MODE,GR2
 1344  011242  011460  000002          1$:     MOV     (R4),2(R0)      ;SET PATTERN DBRA FROM H TO L
```

```
 1345  011246  112710  000020                        MOVB    #CIRMR,(R0)     ;CLEAR IMR+IRR GROUP 1
 1346  011252  112712  000020                        MOVB    #CIRMR,(R2)     ;CLEAR IMR+IRR GROUP 2
 1347  011256  005060  000002                        CLR     2(R0)           ;CLEAR DBRA,ACTIVE LOW WILL SET RPLY C
 1348  011262  010137  001122              2$:       MOV     R1,$BDADR       ; GROUP 1 DATA PORT
 1349  011266  112710  000250                        MOVB    #MIRR,(R0)      ;LOAD BITS TO READ IRR
 1350  011272  111437  001124                        MOVB    (R4),$GDDAT
 1351  011276  111115                                MOVB    (R1),(R5)       ;READ IRR,GROUP 1
 1352  011300  023715  001124                        CMP     $GDDAT,(R5)
 1353  011304  001401                                BEQ     3$
 1354  011306  104003                                ERROR   3               ;IRR ERROR,GROUP 1
 1355  011310  010337  001122              3$:       MOV     R3,$BDADR       ;GROUP 2 DATA PORT
 1356  011314  112712  000250                        MOVB    #MIRR,(R2)      ;LOAD MODE BITS TO READ IRR GROUP2
 1357  011320  116437  000001  001124                MOVB    1(R4),$GDDAT    ;BUILD EXPECTED DATA
 1358  011326  042737  000360  001124                BIC     #360,$GDDAT     ;SAVE BITS 0-3
 1359  011334  052737  000100  001124                BIS     #100,$GDDAT     ;EXPECTED 0-3,URPLY 6(C)
 1360  011342  111315                                MOVB    (R3),(R5)       ;READ IRR BITS,GROUP 2
 1361  011344  023715  001124                        CMP     $GDDAT,(R5)
 1362  011350  001401                                BEQ     4$
 1363  011352  104003                                ERROR   3               ;IRR ERROR,GROUP 2
 1364  011354  005762  000002              4$:       TST     2(R2)           ;READ DBRC FOR RPLY 4(A)
 1365  011360  052737  000020  001124                BIS     #20,$GDDAT      ;STORE EXPECTED
 1366  011366  111315                                MOVB    (R3),(R5)       ;READ IRR BITS,GROUP 2
 1367  011370  023715  001124                        CMP     $GDDAT,(R5)
 1368  011374  001401                                BEQ     5$
 1369  011376  104003                                ERROR   3               ;IRR ERROR,GROUP 2
 1370  011400  005061  000002              5$:       CLR     2(R1)           ;CLEAR DBRB FOR RPY 7(D)
 1371  011404  052737  000200  001124                BIS     #200,$GDDAT     ;EXPECTED
 1372  011412  111315                                MOVB    (R3),(R5)       ;READ IRR BITS GROUP 2
 1373  011414  023715  001124                        CMP     $GDDAT,(R5)
 1374  011420  001401                                BEQ     6$
 1375  011422  104003                                ERROR   3               ;IRR ERROR,GROUP 2
 1376  011424  005763  000002              6$:       TST     2(R3)           ;READ DBRD FOR RPLY 5(B)
 1377  011430  052737  000040  001124                BIS     #40,$GDDAT
 1378  011436  111315                                MOVB    (R3),(R5)       ;READ IRR BITS GROUP2
 1379  011440  023715  001124                        CMP     $GDDAT,(R5)
 1380  011444  001401                                BEQ     7$              ;GET NEXT PATTERN
 1381  011446  104003                                ERROR   3               ;IRR ERROR,GROUP 2
 1382  011450  005724                      7$:       TST     (R4)+           ;INDEX DATA TABLE
 1383  011452  020427  013532                        CMP     R4,#ENDDAT      ;CHECK FOR END
 1384  011456  001271                                BNE     1$              ;DO NEXT PATTERN
 1385
 1386
 1387                                   ;;*********************************************************************
                                        ;*TEST 45        TEST GROUPS 1,2 IN POLLED MODE,NO REPLY
  (3)                                   ;;*********************************************************************
  (3)
  (2)  011460  000004                   TST45:  SCOPE
 1388  011462  004737  013124                        JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1389                                                                          ;R0 = CSRA-GROUP 1 CONTROL
 1390                                                                          ;R1 = CSRB-GROUP 1 DATA
 1391                                                                          ;R2 = CSRC-GROUP 2 CONTROL
 1392                                                                          ;R3 = CSRD-GROUP 2 DATA
 1393  011466  012704  000002                        MOV     #2,R4           ;TWO PASSES
 1394                                                                          ;FIRST PASS CSRA,CSRB = OUTPUT
 1395                                                                          ;           CSRC,CSRD = INPUT
 1396                                                                          ;SEC PASS   CSRC,CSRD = OUTPUT
 1397                                                                          ;           CSRA,CSRB = INPUT
```

```
 1398  011472  112760  000001  000001  111$:   MOVB    #BIT0,1(R0)     ;SET CSR TO OUTPUT MODE
 1399  011500  112761  000001  000001          MOVB    #BIT0,1(R1)     ;SET CSR TO OUTPUT MODE
 1400  011506  012760  177777  000002          MOV     #-1,2(R0)       ;SET ALL ONES DBR FOR H TO L
 1401  011514  105010                          CLRB    (R0)            ;CHIP RESET GROUP  CSR
 1402  011516  105012                          CLRB    (R2)            ;CHIP RESET GROUP  CSR
 1403  011520  112710  000204                  MOVB    #204,(R0)       ;LOAD MODE BITS FOR POLLED MODE
 1404  011524  112712  000204                  MOVB    #204,(R2)       ;LOAD MODE BITS FOR POLLED MODE
 1405  011530  112710  000020                  MOVB    #CIRMR,(R0)     ;CLEAR IMR+IRR
 1406  011534  112712  000020                  MOVB    #CIRMR,(R2)     ;CLEAR IMR+IRR
 1407  011540  005760  000002                  TST     2(R0)           ;READ DBR IN OUTPUT MODE,NO REPLY
 1408  011544  012737  000320  001124          MOV     #320,$GDDAT     ;STORE EXPECTED
 1409  011552  010037  001122                  MOV     R0,$BDADR       ;STORE ADDRESS
 1410  011556  111015                          MOVB    (R0),(R5)       ;READ STATUS
 1411  011560  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
 1412  011564  023715  001124                  CMP     $GDDAT,(R5)     ;CHECK STATUS
 1413  011570  001401                          BEQ     1$
 1414  011572  104007                          ERROR   7               ;CHIP STATUS ERROR
 1415  011574  010237  001122          1$:     MOV     R2,$BDADR       ;STORE ADDRESS
 1416  011600  111215                          MOVB    (R2),(R5)       ;READ STATUS
 1417  011602  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
 1418  011606  023715  001124                  CMP     $GDDAT,(R5)
 1419  011612  001401                          BEQ     2$
 1420  011614  104007                          ERROR   7               ; CHIP STATUS ERROR
 1421  011616  012762  000000  000002  2$:     MOV     #0,2(R2)        ;WRITE ONLY,DBR INPUT MODE FOR NO REPLY
 1422  011624  005761  000002                  TST     2(R1)           ;READ DBR OUTPUT MODE,NO REPLY
 1423  011630  012763  000000  000002          MOV     #0,2(R3)        ;WRITE ONLY, DBR INPUT MODE,NO REPLY
 1424  011636  010137  001122                  MOV     R1,$BDADR       ; CHIP DATA PORT
 1425  011642  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD BITS TO READ IRR
 1426  011646  005037  001124                  CLR     $GDDAT          ;IRR SHOULD BE ZERO
 1427  011652  111115                          MOVB    (R1),(R5)       ;READ IRR
 1428  011654  023715  001124                  CMP     $GDDAT,(R5)
 1429  011660  001412                          BEQ     3$
 1430  011662  005737  017306                  TST     KXTFLAG         ; ON KXT11, READ-BEFORE-WRITE...;;GPA
 1431  011666  001406                          BEQ     100$            ;...AT 2$ UPSETS EXPECTED IRR.  ;;GPA
 1432  011670  012737  000300  001124          MOV     #300,$GDDAT     ; TRY THE ALTERNATE VALUE.      ;;GPA
 1433  011676  023715  001124                  CMP     $GDDAT,(R5)                                     ;;GPA
 1434  011702  001401                          BEQ     3$                                              ;;GPA
 1435  011704  104003                  100$:   ERROR   3               ;IRR ERROR                      ;;GPA
 1436  011706  005037  001124          3$:     CLR     $GDDAT                                          ;;GPA
 1437  011712  010337  001122                  MOV     R3,$BDADR       ;CHIP DATA PORT
 1438  011716  112712  000250                  MOVB    #MIRR,(R2)      ;LOAD MODE BITS TO READ IRR
 1439  011722  111315                          MOVB    (R3),(R5)       ;READ IRR BITS
 1440  011724  023715  001124                  CMP     $GDDAT,(R5)
 1441  011730  001412                          BEQ     4$
 1442  011732  005737  017306                  TST     KXTFLAG         ; ON KXT11, DITTO.              ;;GPA
 1443  011736  001406                          BEQ     101$                                            ;;GPA
 1444  011740  012737  000060  001124          MOV     #60,$GDDAT      ; TRY THE ALTERNATE VALUE.      ;;GPA
 1445  011746  023715  001124                  CMP     $GDDAT,(R5)                                     ;;GPA
 1446  011752  001401                          BEQ     4$                                              ;;GPA
 1447  011754  104003                  101$:   ERROR   3               ;IRR ERROR                      ;;GPA
 1448  011756  005304                  4$:     DEC     R4              ;FINISHED TWO PASSES
 1449  011760  001413                          BEQ     TST46           ;;BR IF EQUAL
 1450  011762  004737  013124                  JSR     PC,CLRCSR       ;CLEAR ALL CSRS
 1451  011766  013700  001352                  MOV     DRCSC,R0        ;SET UP FOR NEXT PASS
 1452  011772  013701  001356                  MOV     DRCSD,R1
 1453  011776  013702  001342                  MOV     DRCSA,R2
```

```
                                MOV     DRCSB,R3
                                BR      111$            ;DO NEXT PASS

                        ;;*****************************************************************
                        ;*TEST 46       TEST POLLED MODE;CSRS C,D=OUT A,B=IN,ACTIVE LOW
                        ;;*****************************************************************
                        TST46:  SCOPE
                                JSR     PC,CLRCSR       ;CLEAR ALL CSRS
                                                        ;R0 = CSRA-GROUP 1 CONTROL
                                                        ;R1 = CSRB-GROUP 1 DATA
                                                        ;R2 = CSRC-GROUP 2 CONTROL
                                                        ;R3 = CSRD-GROUP 2 DATA
                                MOV     #1$,$LPERR      ;SCOPE LOOP RETURN
                                MOV     #BEGPAT,R4      ;PATTERN TABLE
                                MOVB    #BIT0,1(R2)     ;SET CSRC TO OUTPUT MODE
                                MOVB    #BIT0,1(R3)     ;SET CSRD TO OUTPUT MODE
                                CLRB    (R0)            ;CHIP RESET GROUP 1 CSRA
                                CLRB    (R2)            ;CHIP RESET GROUP 2 CSRC
                                MOVB    #204,(R0)       ;LOAD MODE BITS FOR POLLED MODE,GR 1
                                MOVB    #204,(R2)       ;LOAD MODE BITS FOR POLLED MODE,GR2
                        1$:     MOV     (R4),2(R2)      ;SET PATTERN DBRC FOR H TO L
                                MOVB    #CIRMR,(R0)     ;CLEAR IMR+IRR GROUP 1
                                MOVB    #CIRMR,(R2)     ;CLEAR IMR+IRR GROUP 2
                                CLR     2(R2)           ;CLEAR DBRC,ACTIVE LOW WILL SET RPLY A
                        2$:     MOV     R1,$BDADR       ; GROUP 1 DATA PORT
                                MOVB    #MIRR,(R0)      ;LOAD BITS TO READ IRR
                                MOVB    (R4),$GDDAT
                                MOVB    (R1),(R5)       ;READ IRR,GROUP 1
                                CMP     $GDDAT,(R5)
                                BEQ     3$
                                ERROR   3               ;IRR ERROR,GROUP 1
                        3$:     MOV     R3,$BDADR       ;GROUP 2 DATA PORT
                                MOVB    #MIRR,(R2)      ;LOAD MODE BITS TO READ IRR GROUP2
                                MOVB    1(R4),$GDDAT
                                BIC     #360,$GDDAT     ;SAVE BITS 0-3
                                BIS     #20,$GDDAT      ;EXPECTED 0-3,URPLY 4(A)
                                MOVB    (R3),(R5)       ;READ IRR BITS,GROUP 2
                                CMP     $GDDAT,(R5)
                                BEQ     4$
                                ERROR   3               ;IRR ERROR,GROUP 2
                        4$:     TST     2(R0)           ;READ DBRA FOR RPLY 6(C)
                                BIS     #100,$GDDAT     ;STORE EXPECTED
                                MOVB    (R3),(R5)       ;READ IRR BITS,GROUP 2
                                CMP     $GDDAT,(R5)
                                BEQ     5$
                                ERROR   3               ;IRR ERROR,GROUP 2
                        5$:     CLR     2(R3)           ;CLEAR DBRD FOR RPY 5(B)
                                BIS     #40,$GDDAT      ;EXPECTED
                                MOVB    (R3),(R5)       ;READ IRR BITS GROUP 2
                                CMP     $GDDAT,(R5)
                                BEQ     6$
                                ERROR   3               ;IRR ERROR,GROUP 2
                        6$:     TST     2(R1)           ;READ DBRB FOR RPLY 7(D)
                                BIS     #200,$GDDAT
                                MOVB    (R3),(R5)       ;READ IRR BITS GROUP2
                                CMP     $GDDAT,(R5)
```

```
  1507  012262  001401                          BEQ     7$
  1508  012264  104003                          ERROR   3                       ;IRR ERROR,GROUP 2
  1509  012266  005724                  7$:     TST     (R4)+                   ;INDEX DATA TABLE
  1510  012270  020427  013532                  CMP     R4,#ENDDAT              ;CHECK FOR END
  1511  012274  001271                          BNE     1$                      ;DO NEXT PATTERN
  1512
  1513
  1514
  1515                                  ;;**********************************************************************
   (3)                                  ;*TEST 47        TEST IRR BITS WITH DATA PAT.,POLLED MODE,ACT. HIGH
   (3)                                  ;;**********************************************************************
   (2)  012276  000004                  TST47:  SCOPE
  1516  012300  004737  013124                  JSR     PC,CLRCSR               ;CLEAR ALL CSRS
  1517  012304  012703  013232                  MOV     #BEGPAT,R3              ;GET DATA PATTERN TABLE
  1518  012310  012737  000001  001110          MOV     #1,$LPERR               ;SET UP SCOPE ADDRESS
  1519  012316  013700  001342                  MOV     DRCSA,R0
  1520  012322  013701  001346                  MOV     DRCSB,R1
  1521  012326  013702  001356                  MOV     DRCSD,R2
  1522  012332  112760  000001  000001          MOVB    #BIT0,1(R0)             ;CSRA OUTPUT MODE
  1523  012340  112761  000001  000001          MOVB    #BIT0,1(R1)             ;CSRB OUTPUT MODE
  1524  012346  112710  000224                  MOVB    #224,(R0)               ;LOAD MODE BITS FOR POLLED MODE,GP1
  1525  012352  112760  000224  000010          MOVB    #224,10(R0)             ;LOAD MODE BITS FOR POLLED MODE,GP2
  1526  012360  005060  000002          1$:     CLR     2(R0)                   ;CLEAR DBRA
  1527  012364  112710  000020                  MOVB    #CIRMR,(R0)             ;CLEAR IMR +IRR GROUP 1
  1528  012370  112760  000020  000010          MOVB    #CIRMR,10(R0)           ;CLEAR IMR + IRR GROUP 2
  1529  012376  011360  000002                  MOV     (R3),2(R0)              ;STORE PATTERN INTO DBRA
  1530  012402  010137  001122                  MOV     R1,$BDADR               ;CSRB ADDRESS
  1531  012406  112710  000250                  MOVB    #MIRR,(R0)              ;LOAD BITS TO READ IRR1
  1532  012412  111337  001124                  MOVB    (R3),$GDDAT
  1533  012416  111115                          MOVB    (R1),(R5)               ;READ IRR,GP1
  1534  012420  023715  001124                  CMP     $GDDAT,(R5)
  1535  012424  001401                          BEQ     2$
  1536  012426  104003                          ERROR   3                       ;IRR ERROR,GP1
  1537  012430  010237  001122          2$:     MOV     R2,$BDADR               ;CSRD ADDRESS
  1538  012434  112760  000250  000010          MOVB    #MIRR,10(R0)            ;SET MODE TO READ IRR BITS GROUP 2
  1539  012442  116337  000001  001124          MOVB    1(R3),$GDDAT            ;BUILD EXPECTED DATA
  1540  012450  042737  000360  001124          BIC     #360,$GDDAT             ;BITS 0-3 EXPECTED
  1541  012456  052737  000100  001124          BIS     #100,$GDDAT             ;"A" REPLY EXPECTED
  1542  012464  111215                          MOVB    (R2),(R5)               ;READ IRR2
  1543  012466  023715  001124                  CMP     $GDDAT,(R5)
  1544  012472  001401                          BEQ     3$
  1545  012474  104003                          ERROR   3                       ;IRR GROUP 2
  1546  012476  005723                  3$:     TST     (R3)+                   ;INDEX DATA TABLE
  1547  012500  020327  013532                  CMP     R3,#ENDDAT              ;CHECK FOR PATTERN END
  1548  012504  001325                          BNE     1$                      ;DO NEXT PATTERN
  1549
  1550
  1551                                  ;;**********************************************************************
   (3)                                  ;*TEST 50        TEST CSRA AND CSRB WITH RESET
   (3)                                  ;;**********************************************************************
   (2)  012506  000004                  TST50:  SCOPE
   (1)  012510  012737  000001  001160          MOV     #1,$TIMES               ;;DO 1 ITERATION
  1552  012516  004737  013124                  JSR     PC,CLRCSR               ;CLEAR ALL CSRS
  1553  012522  013700  001342                  MOV     DRCSA,R0
  1554  012526  013701  001346                  MOV     DRCSB,R1
  1555  012532  112760  001001  000001          MOVB    #IE!BIT0,1(R0)          ;CSRA TO OUTPUT MODE
```

```
 1556  012540  112761  000001  000001          MOVB    #BIT0,1(R1)     ;SET CSRB TO OUTPUT MODE
 1557  012546  010037  001122                  MOV     R0,$BDADR       ;STORE CSRA ADDRESS
 1558  012552  012737  100300  001124          MOV     #100300,$GDDAT  ;RDY + CHIP STATUS EXP'D
 1559  012560  000005                          RESET                   ;INITIALIZE
 1560  012562  011015                          MOV     (R0),(R5)       ;STORE REC'D
 1561  012564  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
 1562  012570  023715  001124                  CMP     $GDDAT,(R5)     ;MAKE COMPARE
 1563  012574  001401                          BEQ     1$
 1564  012576  104002                          ERROR   2               ;CSRA ERROR ON RESET
 1565  012600  012737  100000  001124  1$:     MOV     #100000,$GDDAT  ;STORE EXPECTED
 1566  012606  010137  001122                  MOV     R1,$BDADR       ;CHECK CSRB
 1567  012612  011115                          MOV     (R1),(R5)       ;STORE IN $BDDAT
 1568  012614  023715  001124                  CMP     $GDDAT,(R5)     ;MAKE COMPARE
 1569  012620  001401                          BEQ     TST51           ;;BR IF EQUAL
 1570  012622  104002                          ERROR   2               ;CSRB ERROR WITH RESET
 1571
 1572
 1573                                  ;;*******************************************************************
   (3)                                 ;*TEST 51       TEST CSRC AND CSRD WITH RESET
   (3)                                 ;;*******************************************************************
   (2)  012624  000004                  TST51:  SCOPE
   (1)  012626  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
 1574  012634  004737  013124                  JSR     PC,CLRCSR
 1575  012640  013702  001352                  MOV     DRCSC,R2
 1576  012644  013703  001356                  MOV     DRCSD,R3
 1577  012650  112762  000001  000001          MOVB    #BIT0,1(R2)     ;SET CSRC TO OUTPUT MODE
 1578  012656  112763  000001  000001          MOVB    #BIT0,1(R3)     ;SET CSRD TO OUTPUT MODE
 1579  012664  010237  001122                  MOV     R2,$BDADR       ;STORE ADDRESS
 1580  012670  012737  100200  001124          MOV     #100200,$GDDAT  ;RDY + CHIP STATUS
 1581  012676  000005                          RESET                   ;INITIALIZE
 1582  012700  011215                          MOV     (R2),(R5)       ;STORE CSRC
 1583  012702  042715  000007                  BIC     #7,(R5)         ;CLEAR UNDEFINED BITS
 1584  012706  023715  001124                  CMP     $GDDAT,(R5)     ;MAKE COMPARE
 1585  012712  001401                          BEQ     1$
 1586  012714  104002                          ERROR   2               ;CSRC ERROR WITH RESET
 1587  012716  010337  001122          1$:     MOV     R3,$BDADR       ;CHECK CSRD
 1588  012722  012737  100000  001124          MOV     #100000,$GDDAT  ;STORE EXPECTED
 1589  012730  011315                          MOV     (R3),(R5)       ;READ CSRD
 1590  012732  023715  001124                  CMP     $GDDAT,(R5)     ;MAKE COMPARE
 1591  012736  001401                          BEQ     2$
 1592  012740  104002                          ERROR   2               ;CSRD ERROR ON RESET
 1593  012742                          2$:
 1594
```

```
                                        ;DON'T REPORT "EOP" UNTIL ALL SELECTED DRV11-J'S HAVE BEEN TESTED.
013742  013701  001342                NXDEV:  MOV     DRCSA,R1        ;INIT TO SETUP NEXT DRV11J ADDRESSES
012746  000241                        NXDEV1: CLC                     ;CLEAR CARRY FOR DEVICE MAP
012750  006037  001364                        ROR     DMAP            ;LOOK FOR NEXT DRV11J
012754  001412                        BEQ     $EOP            ;BR IF ALL TESTED
012756  162701  000020                SUB     #20,R1          ;NEXT DRV11-J STARTS -20
012762  005237  001202                INC     $UNIT           ;UPDATE UNIT NUMBER
012766  032737  000001  001364        BIT     #1,DMAP         ;IS UNIT SELECTED?
012774  001764                        BEQ     NXDEV1          ;NEXT,IF NOT SELECTED
012776  000137  002100                JMP     NEXPAS          ;TEST NEXT DRV11-J
                                .SBTTL  END OF PASS ROUTINE
                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO START1
                                
013002                          $EOP:
013002  000240                          NOP
013004  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
013010  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
013014  005237  001176                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
013020  042737  100000  001176          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
013026  005327                          DEC     (PC)+           ;;LOOP?
013030  000001                  $EOPCT. .WORD   1
013032  003022                          BGT     $DOAGN          ;;YES
013034  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
013036  000001                  $ENDCT: .WORD   1
013040  013030                          $EOPCT
013042  104401  013107                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
013046  013746  001176                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
013052  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
013054  104401  013104                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
013060  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
013064  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
013066  000005                          RESET                   ;;CLEAR THE WORLD
013070  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
013072  000240                          NOP                     ;;SAVE ROOM
013074  000240                          NOP                     ;;FOR
013076  000240                          NOP                     ;;ACT11
013100                          $DOAGN:
013100  000137                          JMP     @(PC)+          ;;RETURN
013102  002034                  $RTNAD: .WORD   START1
013104     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
013107     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
```

```
                                        .SBTTL  PROGRAM SUBROUTINES


                                        ;;*****************************************************************
                                        ;CLEAR ALL CONTROL/STATUS REGISTERS
                                        ;INIT REGISTERS R0-R4 WITH CSRA-CSRD
                                        ;AND STORE $BDDAT INTO R5.
                                        ;;*****************************************************************
013124  012705  001126          CLRCSR: MOV     #$BDDAT,R5      ;BAD DATA STORAGE IN R5
013130  013700  001342                  MOV     DRCSA,R0        ;CSRA ADDRESS TO R0
013134  013701  001346                  MOV     DRCSB,R1        ;CSRB ADDRESS TO R1
013140  013702  001352                  MOV     DRCSC,R2        ;CSRC ADDRESS TO R2
013144  013703  001356                  MOV     DRCSD,R3        ;CSRD ADDRESS TO R3
013150  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
013154  005015                          CLR     (R5)            ;CLEAR REC'D
013156  005010                          CLR     (R0)            ;CLEAR CSRA;CHIP RESET GROUP 1
013160  105061  000001                  CLRB    1(R1)           ;CLEAR HIGH BYTE CSRB
013164  005012                          CLR     (R2)            ;CLEAR CSRC;CHIP RESET GROUP 2
013166  105063  000001                  CLRB    1(R3)           ;CLEAR HIGH BYTE CSRD
013172  000207                          RTS     PC


                                        ;CLEAR IRR REGISTERS,GROUP 1,GROUP 2 WITH CHIP RESET
013174  010046                  CLRIRR: MOV     R0,-(SP)
013176  013700  001342                  MOV     DRCSA,R0        ;START OF CSR ADDRESS
013202  105060  000011                  CLRB    11(R0)          ;CSRC TO INPUT MODE
013206  112760  000001  000001          MOVB    #BIT0,1(R0)     ;CSRA TO OUTPUT MODE
013214  005060  000002                  CLR     2(R0)           ;CLEAR DBRA
013220  105010                          CLRB    (R0)            ;CHIP RESET OF GROUP1
013222  105060  000010                  CLRB    10(R0)          ;CHIP RESET OF GROUP 2
013226  012600                          MOV     (SP)+,R0        ;RESTORE REGISTER
013230  000207                          RTS     PC              ;EXIT
```

```
                                .SBTTL  PATTERNS FOR REGISTER R/W
                                ;
                                ;PATTERNS USED FOR LOADING/READING REGISTERS

013232  000000                  BEGPAT: 0                       ;GROWING 1
013234  000001                  BGPAT1: 1
013236  000003                          3
013240  000007                          7
013242  000017                          17
013244  000037                          37
013246  000077                          77
013250  000177                          177
013252  000377                          377
013254  000777                  EDCHP1: 777
013256  001777                          1777
013260  003777                          3777
013262  007777                          7777
013264  017777                          17777
013266  037777                          37777
013270  077777                          77777
013272  177777                          177777
013274  177776                  BGCHP2: 177776                  ;GROWING 0
013276  177774                          177774
013300  177770                          177770
013302  177760                          177760
013304  177740                          177740
013306  177700                          177700
013310  177600                          177600
013312  177400                  EDCHP2: 177400
013314  177000                          177000
013316  176000                          176000
013320  174000                          174000
013322  170000                          170000
013324  160000                          160000
013326  140000                          140000
013330  100000                          100000

013332  000000                          000000
013334  000001                  BGCHP3: 1                       ;WALKING 1
013336  000002                          2
013340  000004                          4
013342  000010                          10
013344  000020                          20
013346  000040                          40
013350  000100                          100
013352  000200                          200
013354  000400                  EDCHP3: 400
013356  001000                          1000
013360  002000                          2000
013362  004000                          4000
013364  010000                          10000
013366  020000                          20000
013370  040000                          40000
013372  100000                          100000
013374  177777                          177777                  ;WALKING 0
```

```
 1701  013376  177776                BGCHP4: 177776
 1702  013400  177775                        177775
 1703  013402  177773                        177773
 1704  013404  177767                        177767
 1705  013406  177757                        177757
 1706  013410  177737                        177737
 1707  013412  177677                        177677
 1708  013414  177577                        177577
 1709  013416  177377                EDCHP4: 177377
 1710  013420  176777                        176777
 1711  013422  175777                        175777
 1712  013424  173777                        173777
 1713  013426  167777                        167777
 1714  013430  157777                        157777
 1715  013432  137777                        137777
 1716  013434  077777                        077777
 1717  013436  177777                        177777
 1718  013440  000000                ENDPAT: 000000
 1719
 1720                                ;DATA  PATTERNS
 1721  013442  155555                PATDAT: 155555
 1722  013444  133333                        133333
 1723  013446  066666                        066666
 1724  013450  125252                        125252
 1725  013452  052525                        052525
 1726  013454  177777                        177777
 1727  013456  000000                        000000
 1728  013460  107070                        107070
 1729  013462  070707                        070707
 173C  013464  144444                        144444
 1731  013466  033333                        033333
 1732  013470  011111                        011111
 1733  013472  022222                        022222
 1734  013474  044444                        044444
 1735  013476  111111                        111111
 1736  013500  166666                        166666
 1737  013502  010421                        010421
 1738  013504  021042                        021042
 1739  013506  031463                        031463
 1740  013510  042104                        042104
 1741  013512  063146                        063146
 1742  013514  073567                        073567
 1743  013516  104210                        104210
 1744  013520  114631                        114631
 1745  013522  135673                        135673
 1746  013524  146314                        146314
 1747  013526  156735                        156735
 1748  013530  167356                        167356
 1749  013532  000000                ENDDAT: 000000
 1750
```

```
                                        .SBTTL  SYSMAC ROUTINES

                                .SBTTL  TYPE ROUTINE

                                ;;******************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

(1)  013534  105737  001157     $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  013540  100002             BPL     1$              ;;BR IF YES
(1)  013542  000000             HALT                    ;;HALT HERE IF NO TERMINAL
(1)  013544  000430             BR      3$              ;;LEAVE
(1)  013546  010046             1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  013550  017600  000002             MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  013554  122737  000001  001210     CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  013562  001011             BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  013564  132737  000100  001211     BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  013572  001405             BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  013574  010037  013604             MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  013600  004737  014076             JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  013604  000000             61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  013606  132737  000040  001211 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  013614  001003             BNE     60$             ;;YES,SKIP TYPE OUT
(1)  013616  112046             2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  013620  001005             BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  013622  005726             TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  013624  012600             60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  013626  062716  000002     3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  013632  000002             RTI                     ;;RETURN
(1)  013634  122716  000011     4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  013640  001430             BEQ     8$
(1)  013642  122716  000200             CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  013646  001006             BNE     5$
(1)  013650  005726             TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  013652  104401             TYPE                    ;;TYPE A CR AND LF
(1)  013654  001165             $CRLF
(1)  013656  105037  014064             CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  013662  000755             BR      2$              ;;GET NEXT CHARACTER
(1)  013664  004737  013746     5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  013670  123726  001156     6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  013674  001350             BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  013676  013746  001154             MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                     ;;AND THE NULL CHAR.
(1)  013702  105366  000001     7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  013706  002770             BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
    (1)  013710  004737  013746                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
    (1)  013714  105337  014064                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
    (1)  013720  000770                         BR      7$              ;;LOOP
    (1)
    (1)                                 ;HORIZONTAL TAB PROCESSOR
    (1)
    (1)  013722  112716  000040          8$:    MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
    (1)  013726  004737  013746          9$:    JSR     PC,$TYPEC       ;;TYPE A SPACE
    (1)  013732  132737  000007  014064        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
    (1)  013740  001372                         BNE     9$              ;;TAB STOP
    (1)  013742  005726                         TST     (SP)+           ;;POP SPACE OFF STACK
    (1)  013744  000724                         BR      2$              ;;GET NEXT CHARACTER
    (1)  013746                         $TYPEC:
    (1)  013746  105777  165172                 TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
    (1)  013752  100022                         BPL     10$             ;;BR IF NOT                     ;MJD001
    (1)  013754  017746  165166                 MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
    (1)  013760  042716  177600                 BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
    (1)  013764  122716  000023                 CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
    (1)  013770  001012                         BNE     102$            ;;BR IF NOT                     ;MJD001
    (1)  013772                         101$:                                                           ;MJD001
    (1)  013772  105777  165146                 TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
    (1)  013776  100375                         BPL     101$                                            ;MJD001
    (1)  014000  117716  165142                 MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
    (1)  014004  042716  177600                 BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
    (1)  014010  122716  000021                 CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
    (1)  014014  001366                         BNE     101$            ;;BR IF NOT                     ;MJD001
    (1)  014016                         102$:                                                           ;MJD001
    (1)  014016  005726                         TST     (SP)+           ;;FIX STACK                     ;MJD001
    (1)  014020                         10$:                                                            ;MJD001
    (1)  014020  105777  165124                 TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
    (1)  014024  100375                         BPL     10$                                             ;MJD001
    (1)  014026  116677  000002  165116        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
    (1)  014034  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
    (1)  014042  001003                         BNE     1$              ;;BRANCH IF NO
    (1)  014044  105037  014064                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
    (1)  014050  000406                         BR      $TYPEX          ;;EXIT
    (1)  014052  122766  000012  000002  1$:    CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
    (1)  014060  001402                         BEQ     $TYPEX          ;;BRANCH IF YES
    (1)  014062  105227                         INCB    (PC)+           ;;COUNT THE CHARACTER
    (1)  014064  000000          $CHARCNT:.WORD 0                       ;;CHARACTER COUNT STORAGE
    (1)  014066  000207          $TYPEX: RTS    PC
    (1)
1755                                .SBTTL  APT COMMUNICATIONS ROUTINE
    (1)
    (2)                                 ;;**********************************************************************
    (1)  014070  112737  000001  014334 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
    (1)  014076  112737  000001  014332 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
    (1)  014104  000403                         BR      $ATYC
    (1)  014106  112737  000001  014334 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
    (1)  014114                         $ATYC:
    (3)  014114  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
    (3)  014116  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
    (1)  014120  105737  014332                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
    (1)  014124  001450                         BEQ     5$              ;;IF NOT:  BR
    (1)  014126  122737  000001  001210        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
    (1)  014134  001031                         BNE     3$              ;;IF NOT:  BR
```

```
   (1)  014136  132737  000100  001211          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
   (1)  014144  001425                          BEQ     3$              ;;IF NOT:  BR
   (1)  014146  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
   (1)  014152  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
   (1)  014160  005737  001170          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
   (1)  014164  001375                          BNE     1$              ;;IF NOT:  WAIT
   (1)  014166  010037  001204                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
   (1)  014172  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
   (1)  014174  001376                          BNE     2$
   (1)  014176  163700  001204                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
   (1)  014202  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
   (1)  014204  010037  001206                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
   (1)  014210  012737  000004  001170          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
   (1)  014216  000413                          BR      5$
   (1)  014220  017637  000004  014244  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
   (1)  014226  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
   (3)  014234  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
   (1)  014240  004737  013534                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
   (1)  014244  000000                  4$:     .WORD   0
   (1)  014246                          5$:
   (1)  014246  105737  014334          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
   (1)  014252  001416                          BEQ     12$             ;;IF NOT:  BR
   (1)  014254  005737  001210                  TST     $ENV            ;;RUNNING UNDER APT?
   (1)  014260  001413                          BEQ     12$             ;;IF NOT:  BR
   (1)  014262  005737  001170          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
   (1)  014266  001375                          BNE     11$             ;;IF NOT:  WAIT
   (1)  014270  017637  000004  001172          MOV     @4(SP),$FATAL   ;;GET ERROR #
   (1)  014276  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
   (1)  014304  005237  001170                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
   (1)  014310  105037  014334          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
   (1)  014314  105037  014333                  CLRB    $LFLG           ;;CLEAR LOG FLAG
   (1)  014320  105037  014332                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
   (3)  014324  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  014326  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  014330  000207                          RTS     PC              ;;RETURN
   (1)  014332     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
   (1)  014333     000          $LFLG:  .BYTE   0               ;;LOG FLAG
   (1)  014334     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
   (1)          014336                  .EVEN
   (1)          000200          APTSIZE=200
   (1)          000001          APTENV=001
   (1)          000100          APTSPOOL=100
   (1)          000040          APTCSUP=040
1756                            .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
   (1)
   (2)                          ;;******************************************************************************
   (1)                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
   (1)                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
   (1)                          ;*$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)  014336  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  014342  116637  000001  014561          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  014350  112637  014563                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  014354  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  014360  000406                          BR      $TYPON
(1)  014362  112737  000001  014561  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  014370  112737  000006  014563          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  014376  112737  000005  014560  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  014404  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  014406  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  014410  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  014412  113704  014563                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  014416  005404                          NEG     R4
(1)  014420  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  014424  110437  014562                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  014430  113704  014561                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  014434  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  014440  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  014442  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  014444  000404                          BR      3$              ;;GO DO MSB
(1)  014446  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  014450  006105                          ROL     R5
(1)  014452  006105                          ROL     R5
(1)  014454  010503                          MOV     R5,R3
(1)  014456  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  014460  105337  014562                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  014464  100016                          BPL     7$              ;;BR IF NO
(1)  014466  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  014472  001002                          BNE     4$              ;;TEST FOR 0
(1)  014474  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  014476  001403                          BEQ     5$              ;;BR IF YES
(1)  014500  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  014502  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  014506  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  014512  110337  014556                  MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  014516  104401  014556                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  014522  105337  014560          7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  014526  003347                          BGT     2$              ;;BR IF MORE TO DO
(1)  014530  002402                          BLT     6$              ;;BR IF DONE
(1)  014532  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  014534  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(1)  014536  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  014540  012604                          MOV     (SP)+,R4        ;;RESTORE R4
(1)  014542  012603                          MOV     (SP)+,R3        ;;RESTORE R3
```

```
   (1)  014544  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
   (1)  014552  012616                          MOV     (SP)+,(SP)
   (1)  014554  000002                          RTI                     ;;RETURN
   (1)  014556     000                8$:       .BYTE   0               ;;STORAGE FOR ASCII DIGIT
   (1)  014557     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
   (1)  014560     000                $OCNT:    .BYTE   0               ;;OCTAL DIGIT COUNTER
   (1)  014561     000                $OFILL:   .BYTE   0               ;;ZERO FILL SWITCH
   (1)  014562  000000                $OMODE:   .WORD   0               ;;NUMBER OF DIGITS TO TYPE
  1757                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
   (1)
   (2)                                ;;**********************************************************************
   (1)                                ;*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)  014564                        $TYPDS:
   (3)  014564  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  014566  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  014570  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (3)  014572  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (3)  014574  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
   (1)  014576  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
   (1)  014602  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
   (1)  014606  100004                          BPL     1$              ;;BR IF INPUT IS POS.
   (1)  014610  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
   (1)  014612  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
   (1)  014620  005000                1$:       CLR     R0              ;;ZERO THE CONSTANTS INDEX
   (1)  014622  012703  015000                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
   (1)  014626  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
   (1)  014632  005002                2$:       CLR     R2              ;;CLEAR THE BCD NUMBER
   (1)  014634  016001  014770                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
   (1)  014640  160105                3$:       SUB     R1,R5           ;;FORM THIS BCD DIGIT
   (1)  014642  002402                          BLT     4$              ;;BR IF DONE
   (1)  014644  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
   (1)  014646  000774                          BR      3$
   (1)  014650  060105                4$:       ADD     R1,R5           ;;ADD BACK THE CONSTANT
   (1)  014652  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
   (1)  014654  001002                          BNE     5$              ;;FALL THROUGH IF 0
   (1)  014656  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
   (1)  014660  100407                          BMI     7$              ;;BR IF YES
   (1)  014662  106316                5$:       ASLB    (SP)            ;;MSD?
   (1)  014664  103003                          BCC     6$              ;;BR IF NO
   (1)  014666  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
   (1)  014674  052702  000060                6$:       BIS     #'0,R2        ;;MAKE THE BCD DIGIT ASCII
   (1)  014700  052702  000040                7$:       BIS     #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
   (1)  014704  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
   (1)  014706  005720                          TST     (R0)+           ;;JUST INCREMENTING
   (1)  014710  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
   (1)  014714  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
   (1)  014716  003002                          BGT     8$              ;;GO TO EXIT
   (1)  014720  010502                          MOV     R5,R2           ;;GET THE LSD
```

```
(1)  014722  000764                        BR      6$                ;;GO CHANGE TO ASCII
(1)  014724  105726                8$:     TSTB    (SP)+             ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  014726  100003                        BPL     9$                ;;BR IF NO
(1)  014730  116663  177777  177776       MOVB    -1(SP),-2(R3)     ;;YES--SET THE SIGN FOR TYPING
(1)  014736  105013                9$:     CLRB    (R3)              ;;SET THE TERMINATOR
(3)  014740  012605                        MOV     (SP)+,R5          ;;POP STACK INTO R5
(3)  014742  012603                        MOV     (SP)+,R3          ;;POP STACK INTO R3
(3)  014744  012602                        MOV     (SP)+,R2          ;;POP STACK INTO R2
(3)  014746  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
(3)  014750  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
(1)  014752  104401  015000                TYPE    ,$DBLK            ;;NOW TYPE THE NUMBER
(1)  014756  016666  000002  000004        MOV     2(SP),4(SP)       ;;ADJUST THE STACK
(1)  014764  012616                        MOV     (SP)+,(SP)
(1)  014766  000002                        RTI                       ;;RETURN TO USER
(1)  014770  023420                $DTBL:  10000.
(1)  014772  001750                        1000.
(1)  014774  000144                        100.
(1)  014776  000012                        10.
(1)  015000  000004                $DBLK:  .BLKW   4
1758                               .SBTTL  ERROR HANDLER ROUTINE
 (1)
 (2)                               ;;**********************************************************************
 (1)                               ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 (1)                               ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 (1)                               ;*AND GO TO SWRCK ON ERROR
 (1)                               ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 (1)                               ;*SW15=1          HALT ON ERROR
 (1)                               ;*SW13=1          INHIBIT ERROR TYPEOUTS
 (1)                               ;*SW09=1          LOOP ON ERROR
 (1)                               ;*CALL
 (1)                               ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 (1)
(1)  015010                        $ERROR:
(1)  015010  104407                        CKSWR                     ;;TEST FOR CHANGE IN SOFT-SWR
(2)  015012  104407                        CKSWR           ;GO LOOK FOR SWR CHANGE
(1)  015014  105237  001103        7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
(1)  015020  001775                        BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
(1)  015022  013777  001102  164112        MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  015030  005237  001112                INC     $ERTTL            ;;INC THE ERROR COUNT
(1)  015034  011637  001116                MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  015040  162737  000002  001116        SUB     #2,$ERRPC
(1)  015046  117737  164044  001114        MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  015054  032777  020000  164056        BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
(1)  015062  001004                        BNE     20$               ;;SKIP TYPEOUTS
(1)  015064  004737  015164                JSR     PC,SWRCK          ;;GO TO USER ERROR ROUTINE
(1)  015070  104401  001165                TYPE    ,$CRLF
(1)  015074                        20$:
(1)  015074  122737  000001  001210        CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
(1)  015102  001007                        BNE     2$                ;;NO,SKIP APT ERROR REPORT
(1)  015104  113737  001114  015116        MOVB    $ITEMB,21$        ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  015112  004737  014106                JSR     PC,$ATY4          ;;REPORT FATAL ERROR TO APT
(1)  015116     000                21$:    .BYTE   0
(1)  015117     000                        .BYTE   0
(1)  015120  000777                22$:    BR      22$               ;;APT ERROR LOOP
(1)  015122  005777  164012        2$:     TST     @SWR              ;;HALT ON ERROR
(1)  015126  100002                        BPL     3$                ;;SKIP IF CONTINUE
```

```
  (1)  015130  000000                          HALT                    ;;HALT ON ERROR!
  (1)  015132  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
  (1)  015134  032777  001000  163776  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
  (1)  015142  001402                          BEQ     4$              ;;BR IF NO
  (1)  015144  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
  (1)  015150  005737  001162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
  (1)  015154  001402                          BEQ     5$              ;;BR IF NONE
  (1)  015156  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
  (1)  015162                          5$:
  (1)  015162  000002                          RTI                     ;;RETURN
 1759                                 ;;*********************************************************************
 1760                                 ;GO TYPE ERROR
 1761                                 ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
 1762                                 ;;*********************************************************************
 1763  015164  113737  001102  001362 $SWRCK:  MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ER
 1764  015172  004737  015202                  JSR     PC,$ERRTYP      ;GO TYPE ERROR
 1765  015176  104407                          CKSWR                   ;GO LOOK FOR SWR CHANGE
 1766  015200  000207                          RTS     PC              ;RETURN TO ERROR HANDLER
 1767                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
  (1)
  (2)                                 ;;*********************************************************************
  (1)                                 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
  (1)                                 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
  (1)                                 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
  (1)
  (1)  015202                          $ERRTYP:
  (1)  015202  104401  001165                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
  (1)  015206  010046                          MOV     R0,-(SP)        ;;SAVE R0
  (1)  015210  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
  (1)  015212  153700  001114                  BISB    @#$ITEMB,R0
  (1)  015216  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
  (1)                                                                  ;;TYPE THE PC OF THE ERROR
  (2)  015220  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
  (2)                                                                  ;;ERROR ADDRESS
  (2)  015224  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (1)  015226  000426                          BR      6$              ;;GET OUT
  (1)  015230  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
  (1)  015232  006300                          ASL     R0              ;;       WORK FOR THE ERROR TABLE
  (1)  015234  006300                          ASL     R0
  (1)  015236  006300                          ASL     R0
  (1)  015240  062700  001252                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
  (1)  015244  012037  015254                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
  (1)  015250  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
  (1)  015252  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
  (1)  015254  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
  (1)  015256  104401  001165                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
  (1)  015262  012037  015272          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
  (1)  015266  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
  (1)  015270  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
  (1)  015272  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
  (1)  015274  104401  001165                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
  (1)  015300  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
  (1)  015302  001004                          BNE     7$              ;;GO TYPE THE DATA
  (1)  015304  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
  (1)  015306  104401  001165                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
  (1)  015312  000207                          RTS     PC              ;;RETURN
```

```
   (1)  015314                          7$:
   (2)  015314  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
   (2)  015316  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   (1)  015320  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
   (1)  015322  001770                          BEQ     6$              ;;BR IF NO
   (1)  015324  104401  015332                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
   (1)  015330  000771                          BR      7$              ;;LOOP
   (1)  015332  020040     000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
   (1)          015336                          .EVEN
  1768                                  .SBTTL  SCOPE HANDLER ROUTINE
   (1)
   (2)                                  ;;**********************************************************************
   (1)                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
   (1)                                  ;*AND LOAD THE TEST NUMBER($TSTNM) 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)
   (1)  015336                          $SCOPE:
   (1)  015336  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
   (1)  015340  032777  040000  163572  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
   (1)  015346  001114                          BNE     $OVER           ;;YES IF SW14=1
   (1)                                  ;#####START OF CODE FOR THE XOR TESTER#####
   (1)  015350  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
   (1)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
   (1)  015352  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
   (1)  015356  012737  015376  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
   (1)  015364  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
   (1)  015370  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  015374  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
   (1)  015376  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
   (1)  015400  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  015404  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
   (1)  015406                          6$:;#####END OF CODE FOR THE XOR TESTER#####
   (1)  015406  032777  000400  163524          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
   (1)  015414  001404                          BEQ     2$              ;;BR IF NO
   (1)  015416  127737  163516  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
   (1)  015424  001465                          BEQ     $OVER           ;;BR IF YES
   (1)  015426  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
   (1)  015432  001421                          BEQ     3$              ;;BR IF NO
   (1)  015434  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
   (1)  015442  101015                          BHI     3$              ;;BR IF NO
   (1)  015444  032777  001000  163466          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
   (1)  015452  001404                          BEQ     4$              ;;BR IF NO
   (1)  015454  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
   (1)  015462  000446                          BR      $OVER
   (1)  015464  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
   (1)  015470  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
   (1)  015474  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
   (1)  015476  032777  004000  163434  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
   (1)  015504  001011                          BNE     1$              ;;BR IF YES
```

```
   (1)  015506  005737  001176                 TST     $PASS           ;;IF FIRST PASS OF PROGRAM
   (1)  015512  001406                         BEQ     1$              ;;       INHIBIT ITERATIONS
   (1)  015514  005237  001104                 INC     $ICNT           ;;INCREMENT ITERATION COUNT
   (1)  015520  023737  001160  001104         CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
   (1)  015526  002024                         BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
   (1)  015530  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
   (1)  015536  013737  015614  001160         MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
   (1)  015544  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
   (1)  015550  113737  001102  001174         MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
   (1)  015556  011637  001106                 MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
   (1)  015562  011637  001110                 MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
   (1)  015566  005037  001162                 CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
   (1)  015572  112737  000001  001115         MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
   (1)  015600  013777  001102  163334  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
   (1)  015606  013716  001106                 MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
   (1)  015612  000002                         RTI                     ;;FIXES PS
   (1)  015614  000062                 $MXCNT: 50.                     ;;MAX. NUMBER OF ITERATIONS
  1769                                 .SBTTL  TTY INPUT ROUTINE
   (1)
   (2)                                 ;;**************************************************************************
   (1)                                 .ENABL  LSB
   (1)
   (2)                                 ;;**************************************************************************
   (1)                                 ;*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)  015616  022737  000176  001140 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
   (1)  015624  001074                         BNE     15$             ;;BRANCH IF NO
   (1)  015626  105777  163312                 TSTB    @$TKS           ;;CHAR THERE?
   (1)  015632  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
   (1)  015634  117746  163306                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
   (1)  015640  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
   (1)  015644  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
   (1)  015650  001062                         BNE     15$             ;;NO, RETURN TO USER
   (1)  015652  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
   (1)  015660  001456                         BEQ     15$             ;;BRANCH IF YES
   (1)
   (1)  015662  104401  016353                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
   (1)  015666  104401  016360          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
   (2)  015672  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
   (2)  015676  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   (1)  015700  104401  016371                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
   (1)  015704  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
   (1)  015706  005046                          CLR     -(SP)           ;;THE NEW SWR
   (1)  015710  105777  163230          7$:     TSTB    @$TKS           ;;CHAR THERE?
   (1)  015714  100375                          BPL     7$              ;;IF NOT TRY AGAIN
   (1)
   (1)  015716  117746  163224                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
   (1)  015722  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
   (1)
   (1)
   (1)
   (1)  015726  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
   (1)  015732  001005                          BNE     10$             ;;BRANCH IF NOT
   (1)  015734  104401  016346                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
```

```
(1)  015740  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  015744  000757                          BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  015746  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  015752  001022                          BNE     16$             ;;BRANCH IF NO
(1)  015754  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  015760  001403                          BEQ     11$             ;;BRANCH IF YES
(1)  015762  016677  000002  163150          MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  015770  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
(1)  015774  104401  001165          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  016000  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  016006  001003                          BNE     15$             ;;BRANCH IF NOT
(1)  016010  012777  000100  163126          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  016016  000002          15$:    RTI                             ;;RETURN
(1)  016020  004737  013746          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  016024  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
(1)  016030  002420                          BLT     18$             ;;BRANCH IF YES
(1)  016032  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
(1)  016036  003015                          BGT     18$             ;;BRANCH IF YES
(1)  016040  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  016044  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  016050  001403                          BEQ     17$             ;;BRANCH IF YES
(1)  016052  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  016054  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  016056  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  016060  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  016064  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  016070  000707                          BR      7$              ;;GET THE NEXT ONE
(1)  016072  104401  001164          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  016076  000720                          BR      20$             ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;*****************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  016100  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  016102  016666  000004  000002  MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  016110  105777  163030          1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  016114  100375                          BPL     1$              ;;A CHARACTER
(1)  016116  117766  163024  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  016124  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  016132  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  016140  001013                          BNE     3$              ;;BRANCH IF NO
(1)  016142  105777  162776          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  016146  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  016150  117746  162772                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  016154  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  016160  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  016164  001366                          BNE     2$              ;;IF NOT DISCARD IT
```

```
    (1)  016166  000750                          BR      1$                ;;YES, RESUME
    (1)  016170  026627  000004  000021  3$:     CMP     4(SP),#$XON       ;;IS IT A RANDOM XON?                    ;RAN001
    (1)  016176  001744                          BEQ     1$                ;;BRANCH IF YES                          ;RAN001
    (1)  016200  026627  000004  000140          CMP     4(SP),#140        ;;IS IT UPPER CASE?
    (1)  016206  002407                          BLT     4$                ;;BRANCH IF YES
    (1)  016210  026627  000004  000175          CMP     4(SP),#175        ;;IS IT A SPECIAL CHAR?
    (1)  016216  003003                          BGT     4$                ;;BRANCH IF YES
    (1)  016220  042766  000040  000004          BIC     #40,4(SP)         ;;MAKE IT UPPER CASE
    (1)  016226  000002                  4$:     RTI                       ;;GO BACK TO USER
    (2)                                  ;;*****************************************************************
    (1)                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
    (1)                                  ;*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)
    (1)  016230  010346                  $RDLIN: MOV     R3,-(SP)          ;;SAVE R3
    (1)  016232  012703  016336          1$:     MOV     #$TTYIN,R3        ;;GET ADDRESS
    (1)  016236  022703  016346          2$:     CMP     #$TTYIN+8.,R3     ;;BUFFER FULL?
    (1)  016242  101405                          BLOS    4$                ;;BR IF YES
    (1)  016244  104410                          RDCHR                     ;;GO READ ONE CHARACTER FROM THE TTY
    (1)  016246  112613                          MOVB    (SP)+,(R3)        ;;GET CHARACTER
    (1)  016250  122713  000177          10$:    CMPB    #177,(R3)         ;;IS IT A RUBOUT
    (1)  016254  001003                          BNE     3$                ;;SKIP IF NOT
    (1)  016256  104401  001164          4$:     TYPE    ,$QUES            ;;TYPE A '?'
    (1)  016262  000763                          BR      1$                ;;CLEAR THE BUFFER AND LOOP
    (1)  016264  111337  016334          3$:     MOVB    (R3),9$           ;;ECHO THE CHARACTER
    (1)  016270  104401  016334                  TYPE    ,9$
    (1)  016274  122723  000015                  CMPB    #15,(R3)+         ;;CHECK FOR RETURN
    (1)  016300  001356                          BNE     2$                ;;LOOP IF NOT RETURN
    (1)  016302  105063  177777                  CLRB    -1(R3)            ;;CLEAR RETURN (THE 15)
    (1)  016306  104401  001166                  TYPE    ,$LF              ;;TYPE A LINE FEED
    (1)  016312  012603                          MOV     (SP)+,R3          ;;RESTORE R3
    (1)  016314  011646                          MOV     (SP),-(SP)        ;;ADJUST THE STACK AND PUT ADDRESS OF THE
    (1)  016316  016666  000004  000002          MOV     4(SP),2(SP)       ;;      FIRST ASCII CHARACTER ON IT
    (1)  016324  012766  016336  000004          MOV     #$TTYIN,4(SP)
    (1)  016332  000002                          RTI                       ;;RETURN
    (1)  016334     000                  9$:     .BYTE   0                 ;;STORAGE FOR ASCII CHAR. TO TYPE
    (1)  016335     000                          .BYTE   0                 ;;TERMINATOR
    (1)  016336  000010                  $TTYIN: .BLKB   8.                ;;RESERVE 8 BYTES FOR TTY INPUT
    (1)  016346  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>      ;;CONTROL "U"
    (1)  016353     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>      ;;CONTROL "G"
    (1)  016360  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
    (1)  016371     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
1770
    (1)                                  .SBTTL  POWER DOWN AND UP ROUTINES
    (2)                                  ;;*****************************************************************
    (1)                                  ;POWER DOWN ROUTINE
    (1)  016402  012737  016546  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST UP
    (1)  016410  012737  000340  000026          MOV     #340,@#PWRVEC+2   ;;PRIO:7
    (3)  016416  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
    (3)  016420  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
    (3)  016422  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
    (3)  016424  010346                          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
    (3)  016426  010446                          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
    (3)  016430  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
```

```
   (3)  016432  017746  162502                    MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
   (1)  016436  010637  016552                    MOV     SP,$SAVR6       ;;SAVE SP
   (1)  016442  012737  016454  000024            MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
   (1)  016450  000000                            HALT
   (1)  016452  000776                            BR      .-2             ;;HANG UP
   (1)
   (2)                                  ;;*****************************************************************
   (1)                                  ;POWER UP ROUTINE
   (1)  016454  012737  016546  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
   (1)  016462  013706  016552                    MOV     $SAVR6,SP       ;;GET SP
   (1;  016466  005037  016552                    CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
   (1)  016472  005237  016552            1$:     INC     $SAVR6          ;;WAIT FOR THE INC
   (1)  016476  001375                            BNE     1$              ;;OF  WORD
   (3)  016500  012677  162434                    MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
   (3)  016504  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
   (3)  016506  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
   (3)  016510  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
   (3)  016512  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
   (3)  016514  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  016516  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  016520  012737  016402  000024            MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
   (1)  016526  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
   (1)  016534  104401                            TYPE                    ;;REPORT THE POWER FAILURE
   (1)  016536  016554                    $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
   (1)  016540  012716                            MOV     (PC)+,(SP)      ;;RESTART AT START1
   (1)  016542  002034                    $PWRAD: .WORD   START1          ;;RESTART ADDRESS
   (1)  016544  000002                            RTI
   (1)  016546  000000                    $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
   (1)  016550  000776                            BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
   (1)  016552  000000                    $SAVR6: 0                       ;;PUT THE SP HERE
  1771  016554  005015  042522  052123    PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
  1772                                    .EVEN
  1773                                    .SBTTL  TRAP DECODER
   (1)
   (2)                                  ;;*****************************************************************
   (1)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
   (1)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
   (1)                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
   (1)                                  ;*GO TO THAT ROUTINE.
   (1)
   (1)  016606  010046                    $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
   (1)  016610  016600  000002                    MOV     2(SP),R0        ;;GET TRAP ADDRESS
   (1)  016614  005740                            TST     -(R0)           ;;BACKUP BY 2
   (1)  016616  111000                            MOVB    (R0),R0         ;;GET RIGHT BYTE C. TRAP
   (1)  016620  006300                            ASL     R0              ;;POSITION FOR INDEXING
   (1)  016622  016000  016642                    MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
   (1)  016626  000200                            RTS     R0              ;;GO TO ROUTINE
   (1)
   (1)
   (1)                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
   (1)
   (1)  016630  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
   (1)  016632  016666  000004  000002            MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
   (1)  016640  000002                            RTI                     ;;RESTORE THE PSW
   (1)
   (3)                                  .SBTTL  TRAP TABLE
```

```
   (3)
   (3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
   (3)                                  ;*BY THE "TRAP" INSTRUCTION.
   (3)
   (3)                                  ;       ROUTINE
   (3)                                  ;       -------
   (3)  016642  016630                  $TRPAD: .WORD   $TRAP2
   (3)  016644  013534                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
   (3)  016646  014362                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  016650  014336                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  016652  014376                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   (3)  016654  014564                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
   (1)
   (3)  016656  015666                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
   (1)
   (3)  016660  015616                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
   (3)  016662  016100                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
   (3)  016664  016230                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
  1774
  1775                                  .SBTTL  ASCII MESSAGES
  1776                                  ;;GPA TITLED:   .ASCIZ  <15><12>/CVDRCA  DRV11J DIAG TEST PART 1  /<15><12>
  1777  016666  005015  053103  051104  TITLED: .ASCIZ  <15><12>/CVDRCB  DRV11J DIAG TEST PART 1  /<15><12> ;;GPA
  1778  016734  005015  051104  030526  TLCABL: .ASCIZ  <15><12>/DRV11J CABLE REQ'D/<15><12>
  1779
  1780  016763     122  043505  052040  EM1:    .ASCIZ  /REG TIMEOUT ER/
  1781  017002  042522  020107  042522  EM2:    .ASCIZ  'REG READ/WRITE ER'
  1782  017024  051111  020122  042522  EM3:    .ASCIZ  /IRR REG ER/
  1783  017037     101  051103  051040  EM4:    .ASCIZ  /ACR REG ER/
  1784  017052  046511  020122  042522  EM5:    .ASCIZ  /IMR REG ER/
  1785  017065     111  051123  051040  EM6:    .ASCIZ  /ISR REG ER/
  1786  017100  044103  050111  05.440  EM7:    .ASCIZ  /CHIP STAT ER/
  1787  017115     105  051122  041520  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
  1788  017162  051105  050122  020103  DH2:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  ADRS    EXPCT   RCVD/
  1789
  1790          017240                          .EVEN
  1791  017240  001116  001362  001122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
  1792  017254  001116  001362  001122  DT2:    $ERRPC,TSTNUM,$BDADR,$GDADR,$GDDAT,$BDDAT,0
```

```
                                        .SBTTL  FALCON (KXT-11) UPGRADE ROUTINES.                       ;:GPA
                                        ;
                                        ; THE FOLLOWING ROUTINES HAVE BEEN ADDED TO ALLOW DIAGNOSTIC(S)
                                        ; TO RUN ON A FALCON (KXT-11) BASED SYSTEM.
                                        ; TO DETERMINE WHETHER WE'RE A FALCON OR NOT, WE'LL SIZE THE 1ST 3/4 OF
                                        ; THE I/O PAGE (28K TO 31K).  FALCON HAS 2KW LOCAL RAM AT 28K(+4) TO 30K
                                        ; AND A MACRO-ODT AT 30K TO 31K.  CONSEQUENTLY, ALL I/O DEVICES MUST
                                        ; BE PLACED BETWEEN 174000 AND 177776.  ADDITIONALLY, WE'LL STRAP THE
                                        ; EMT AND TRAP SERVICE LEVEL TO PRI6, AND SET THE HALT VECTOR SO THAT
                                        ; WE CAN STOP THE SUCKER !!
                                        ;
                                        ; TO MINIMIZE THE IMPACT OF THESE CHANGES ON FINAL PROGRAM SIZE, THE
                                        ; BULK OF THIS CODE IS PLACED IN THE FLOATING VECTOR SPACE (400-776).
                                        ; IF THE CPU AT HAND IS A FALCON (KXT11), IT STAYS THERE (NO HARM DONE).
                                        ; OTHERWISE, THE AREA IS RESTORED TO ITS ORIGINAL "TRAP-CATCHER" STATE.
                                        ;
017272  005227  177777          FALCON: INC     #-1             ; ONCE-ONLY !!!                 ;:GPA
017276  001002                          BNE     1$                                              ;:GPA
017300  004737  000400                  CALL    KXTCHK          ; EXECUTE FALCON CHECK          ;:GPA
017304  005727                  1$:     TST     (PC)+           ; TEST FALCON FLAG...           ;:GPA
017306  000000                  KXTFLAG: 0                      ;...NZ = FALCON...              ;:GPA
017310  000207                          RETURN                  ;...AND RETURN TO CALLER...     ;:GPA
                                        
        017312                          $SVPC= .                                                ;:GPA
        000400                          .= 400                  ; RESTORE FROM 374:376 AT END   ;:GPA
000400  005037  017306          KXTCHK: CLR     KXTFLAG         ; ASSUME NOT FALCON.            ;:GPA
000404  013746  000004                  MOV     @#4,-(SP)       ; SAVE ERROR VECTOR.            ;:GPA
000410  012737  000504  000004          MOV     #2$,@#4         ; SET A TRAP CATCHER.           ;:GPA
000416  012700  160010                  MOV     #160010,R0      ; FALCON RAM STARTS AT 28K+4.   ;:GPA
000422  005720                  1$:     TST     (R0)+           ;                               ;:GPA
000424  000240                          240                                                     ;:GPA
000426  020027  174000                  CMP     R0,#174000      ; SIZE TO 31K.                  ;:GPA
000432  103773                          BLO     1$                                              ;:GPA
000434  010037  017306                  MOV     R0,KXTFLAG      ; MUST BE FALCON, SET THE FLAG  ;:GPA
000440  012700  000040                  MOV     #40,R0          ; GET PRI1 BIT...               ;:GPA
000444  040037  000006                  BIC     R0,@#6          ;...AND LOWER BUS-ERROR...      ;:GPA
000450  040037  000016                  BIC     R0,@#16         ;...BPT...                      ;:GPA
000454  040037  000022                  BIC     R0,@#22         ;...IOT...                      ;:GPA
000460  040037  000032                  BIC     R0,@#32         ;...EMT...                      ;:GPA
000464  040037  000036                  BIC     R0,@#36         ;...AND TRAP SERVICE TO PRI6    ;:GPA
000470  012737  170000  000140          MOV     #170000,@#140   ; ENABLE "BREAK" HALT.          ;:GPA
000476  012637  000004                  MOV     (SP)+,@#4       ; RESTORE ERROR VECTOR...       ;:GPA
000502  000207                          RETURN                  ;...AND RETURN.                 ;:GPA
                                        
000504  012716  000512          2$:     MOV     #3$,(SP)        ; TRAP -- NOT A FALCON...       ;:GPA
000510  000002                          RTI                     ;...CONTINUE.                   ;:GPA
000512  012637  000004          3$:     MOV     (SP)+,@#4       ; RESET ERROR VECTOR            ;:GPA
000516  012700  000402                  MOV     #402,R0         ; SET-UP TO RESTORE FLOATING... ;:GPA
000522  013701  000376                  MOV     @#376,R1        ;...VECTORS (400 - 776).        ;:GPA
000526  010602                          MOV     SP,R2           ; SAVE STACK POINTER IN R2      ;:GPA
000530  012704  000570                  MOV     #6$,R4
000534  014446                  4$:     MOV     -(R4),-(SP)     ; PUSH THE RESTORE CODE...      ;:GPA
000536  020427  000546                  CMP     R4,#5$          ;...ONTO THE STACK.             ;:GPA
000542  101374                          BHI     4$                                              ;:GPA
000544  010607                          MOV     SP,PC           ; AND EXECUTE IT.               ;:GPA
```

```
                                ;
                                ; THIS CODE IS RELOCATED TO AND EXECUTED IN THE STACK AREA.
                                ;
000546  010060  177776        5$:     MOV     R0,-2(R0)       ; RESTORE .+2...                ;;GPA
000552  010110                        MOV     R1,(R0)         ;...HALT (OR IOT).              ;;GPA
000554  022020                        CMP     (R0)+,(R0)+                                     ;;GPA
000556  020027  000776                CMP     R0,#776                                         ;;GPA
000562  101771                        BLOS    5$              ; LOOP 'TIL DONE                ;;GPA
000564  010206                        MOV     R2,SP           ; THEN RESTORE SP...            ;;GPA
000566  000207                        RETURN                  ;...AND RETURN TO CALLER        ;;GPA
000570                          6$:                                                           ;;GPA
                                ;
                                ; IF FALCON, THIS AREA IS FREE FOR ANY PROGRAM UNIQUE
                                ; CHANGES OR DATA STRUCTURES.
                                ; IF USED, YOU'D BETTER PROTECT IT !!!
                                ;
000104                          $FREE=  <1000-.>/2              ; FREE WORDS LEFT.          ;;GPA
                                
017312                                  .=$SVPC                                             ;;GPA
017312                          LASTAD= .                                                   ;;GPA
000001                                  .END
```

```
ABASE = 164160
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACRLOC  001400
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000001
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000

BEGPAT  013232
BGCHP2  013274
BGCHP3  013334
BGCHP4  013376
BGPAT1  013234
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BPTVEC= 000014
CHPISR= 000140
CIMR  = 000040
CIRMR = 000020
CIRR  = 000100
CISR  = 000160
CKSWR = 104407
CLRCSR  013124
CLRIRR  013174
CR    = 000015
CRLF  = 000200
CSIMR = 000050
CSIRMR= 000030
CSIRR = 000110
CSISR = 000170
DDISP = 177570
DH1     017115
DH2     017162
DIR   = 000400
DISPLA  001142
DISPRE  000174
DMAP    001364

DRCSA   001342
DRCSB   001346
DRCSC   001352
DRCSD   001356
DRDBA   001344
DRDBB   001350
DRDBC   001354
DRDBD   001360
DSWR  = 177570
DT1     017240
DT2     017254
EDCHP1  013254
EDCHP2  013312
EDCHP3  013354
EDCHP4  013416
EMTVEC= 000030
EM1     016763
EM2     017002
EM3     017024
EM4     017037
EM5     017052
EM6     017065
EM7     017100
ENDDAT  013532
ENDPAT  013440
ERRVEC= 000004
FALCON  017272
GTSWR = 104406
HT    = 000011
IE    = 001000
IMRLOC  001372
INTFLG  001366
IOTVEC= 000020
IRRLOC  001376
ISRLOC  001374
KXTCHK  000400
KXTFLA  017306
LASTAD= 017312
LF    = 000012
LMD04 = 000200
LMD57 = 000240
MACR  = 000254
MIMR  = 000244
MIRR  = 000250
MISR  = 000240
NEXPAS  002100
NEXPA1  002104
NXDEV   012742
NXDEV1  012746
PACR  = 000300
PATDAT  013442
PIMR  = 000260
PIRQ  = 177772

PIRQVE= 000240
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PVMA  = 000340
PWRMSG  016554
PWRVEC= 000024
RDCHR = 104410
RDLIN = 104411
RDY   = 100000
RESVEC= 000010
R6    =%000006
R7    =%000007
SIMR  = 000060
SIRR  = 000120
SSIMR = 000070
SSIRR = 000130
STACK = 001100
START   001402
START1  002034
STKLMT= 177774
SWR     001140
SWRCK   015164
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
TBITVE= 000014
TITLED  016666
TKVEC = 000060
TLCABL  016734
TPVEC = 000064
TRAPVE= 000034
TRTVEC= 000014
TSTNUM  001362
TST1    002142
TST10   003326
TST11   003422
TST12   003516
TST13   003612
TST14   003706
TST15   004056
TST16   004224
TST17   004524
TST2    002220
TST20   005012
TST21   005312
TST22   005600
TST23   005734
TST24   006144
TST25   006350
TST26   006554
TST27   006672
TST3    002412
TST30   007010
TST31   007102
TST32   007202
TST33   007330
TST34   007456
TST35   007556
TST36   007660
TST37   010012
TST4    002522
TST40   010144
TST41   010270
TST42   010444
TST43   010652
TST44   011172
TST45   011460
TST46   012010
TST47   012276
TST5    002620
TST50   012506
TST51   012624
TST6    002766
TST7    003160
```

```
TYPDS = 104405          $ENDCT  013036          $INTAG  001135          $OMODE  014562          $TRAP   016606
TYPE  = 104401          $ENDMG  013107          $ITEMB  001114          $OVER   015600          $TRAP2  016630
TYPOC = 104402          $ENULL  013104          $LF     001166          $PASS   001176          $TRP  = 000012
TYPON = 104404          $ENV    001210          $LFLG   014333          $PASTM  001006          $TRPAD  016642
TYPOS = 104403          $ENVM   001211          $LPADR  001106          $PWRAD  016542          $TSTM   001004
XXDP    001370          $EOP    013002          $LPERR  001110          $PWRDN  016402          $TSTNM  001102
$APTHD  001000          $EOPCT  013030          $MADR1  001222          $PWRMG  016536          $TTYIN  016336
$ATYC   014114          $ERFLG  001103          $MADR2  001226          $PWRUP  016454          $TYPDS  014564
$ATY1   014070          $ERMAX  001115          $MADR3  001232          $QUES   001164          $TYPE   013534
$ATY3   014076          $ERROR  015010          $MADR4  001236          $RDCHR  016100          $TYPEC  013746
$ATY4   014106          $ERRPC  001116          $MAIL   001170          $RDLIN  016230          $TYPEX  014066
$AUTOB  001134          $ERRTB  001252          $MAMS1  001220          $RDSZ = 000010          $TYPOC  014362
$BASE   001244          $ERRTY  015202          $MAMS2  001224          $RTNAD  013102          $TYPON  014376
$BDADR  001122          $ERTTL  001112          $MAMS3  001230          $SAVR6  016552          $TYPOS  014336
$BDDAT  001126          $ESCAP  001162          $MAMS4  001234          $SCOPE  015336          $UNIT   001202
$CDW1   001250          $ETABL  001210          $MBADR  001002          $SETUP= 000117          $UNITM  001010
$CHARC  014064          $ETEND  001252          $MFLG   014332          $STUP = 177777          $USWR   001214
$CKSWR  015616          $FATAL  001172          $MNEW   016371          $SVLAD  015544          $VECT1  001240
$CMTAG  001100          $FFLG   014334          $MSGAD  001204          $SVPC = 017312          $VECT2  001242
$CM3  = 000000          $FILLC  001156          $MSGLG  001206          $SWR  = 165400          $XOFF = 000023
$CNTLG  016353          $FILLS  001155          $MSGTY  001170          $SWREG  001212          $XON  = 000021
$CNTLU  016346          $FREE = 000104          $MSWR   016360          $SWRMK= 000000          $XTSTR  015350
$CPUOP  001216          $GDADR  001120          $MTYP1  001221          $TESTN  001174          $$GET4= 000000
$CRLF   001165          $GDDAT  001124          $MTYP2  001225          $TIMES  001160          $OFILL  014561
$DBLK   015000          $GET42  013060          $MTYP3  001231          $TKB    001146          .     = 017312
$DEVCT  001200          $GTSWR  015666          $MTYP4  001235          $TKS    001144          .$X   = 001000
$DEVM   001246          $HD   = 000003          $MXCNT  015614          $TN   = 000052
$DOAGN  013100          $4IBTS  001000          $NULL   001154          $TPB    001152
$DTBL   014770          $ICNT   001104          $NWTST= 000001          $TPFLG  001157
$ENDAD  013070          $ILLUP  016546          $OCNT   014560          $TPS    001150

. ABS.  017312      000         CON   RW    ABS   GBL   D

 ERRORS DETECTED:  0

 CVDRCB,CVDRCB=CVDRCB
 RUN-TIME: 22 10 .3 SECONDS
 RUN-TIME RATIO: 93/33=2.8
 CORE USED:  25K  (49 PAGES)
```