# DRV11J

DRV11J DIAG TST PRT 2

CVDRDA0

AH-F761A-MC



FICHE 1 OF 1

JAN 1980

digital

MADE IN USA

## IDENTIFICATION

PRODUCT CODE: AC-F759A-MC

PRODUCT NAME: CVDRDA0 DRV11J DIAG TST PRT2

DATE CREATED: OCTOBER 1979

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 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 DRV11-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 DRV11-J CABLE(BC05W-02) MUST BE INSTALLED WITH 1/2 TWIST BETWEEN THE 50 PIN CONNECTORS.

## 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 PROGRAM 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 CONNECT 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 CVDRDA 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.

## 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 |
| SW12=1 | 010000 | INHIBIT ILLEGAL VECTOR RETURN ROUTINES |
| SW11=1 | 004000 | INHIBIT ITERATIONS |
| SW09=1 | 001000 | LOOP ON ERROR |
| SW08=1 | 0004XX | LOOP ON TEST IN SWR <7-0> |

* SW12 EXPLANATION

SW12 = 1 WILL STORE THE REGULAR TRAP (PC+2,HALT) IN LOCATIONS 204-1774 FOR THE VECTOR TESTING. ON ILLEGAL VECTOR ERRORS,THIS WILL AID THE USER TO DETERMINE THE LOCATION OF THE ILLEGAL VECTOR.
THE LOOPING CAPABILITIES THAT ARE NORMALLY PROVIDED BY ILLEGAL VECTOR SERVICE ROUTINES WHEN SW12=0 WILL NO LONGER BE AVAILABLE WHEN SW12 = 1.
IF USING SW12=1 TO TRACK DOWN THE ILLEGAL VECTOR,THE USER SHOULD RESTART AT ADDRESS 200 WITH SW12=0 TO OBTAIN LOOPING CAPABILITIES AFTER ERROR OCCURS.
VECTOR AREA 0-200 WILL ALWAYS HAVE ILLEGAL VECTOR ROUTINES WHEN VECTOR TESTING (0-200) IN ORDER TO PRESERVE PROGRAM.

5.2 CONTROL

* WARNING:

PLEASE USE ''CTRL AND G'' KEYS AND WAIT FOR THE
''CTRL G'' TO BE ECHOED BEFORE ATTEMPTING TO HALT THE
CPU OR TO STOP TESTING OR THE PROGRAM MAY NOT BE
RESTARTABLE.
PROGRAM RELOCATION FOR LOCATIONS 0-200 TAKES PLACE
DURING THE VECTOR TESTING OF 0-200. THE NORMAL
CONTENTS OF 0-200 MAY NOT GET RESTORED IF THE CPU
IS HALTED AND RELOAD OF THE PROGRAM MAY BE NECESSARY.

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.

### 6.2 ERROR DATA

#### 6.2.1 ERROR TITLE TYPE A

1. REG READ/WRITE ER
2. IRR REG ER
3. ACR REG ER
4. IMR REG ER

5. ISR REG ER

6. CHIP STAT ER

7. MULTIPLE INTERRUPTS RECEIVED

8. INTERRUPT TEST ERROR

| ERRPC | TSTNUM | BUSADR | VAM | ADDR | EXPCT | RCVD |
|---|---|---|---|---|---|---|
| XXXXXX | XXXXXX | 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 |
| VAM | VALUE STORED INTO VECTOR ADDRESS MEMORY INDICATING BYTE COUNT AND LEVEL. |
| ADDR | VECTOR ADDRESS |
| EXPCT | DATA THAT WAS EXPECTED |
| RCVD | DATA THAT WAS RECEIVED |

6.2.2 ERROR TITLE TYPE B

1. ILLEGAL INTERRUPT RECEIVED

| ERRPC | TSTNUM | BUSADR | VAM | ADDR |
|---|---|---|---|---|
| XXXXXX | XXXXXX | XXXXXX | XXXXXX | XXXXXX |

| | |
|---|---|
| ERRPC | ADDRESS WHERE THE ERROR WAS DETECTED |
| TSTNUM | TEST NUMBER WHERE THE ERROR OCCURRED |
| BUSADR | DRV11J BUS ADDRESS |
| VAM | VALUE OF VECTOR ADDRESS MEMORY AT TIME OF ILLEGAL INTERRUPT. |
| ADDR | VECTOR ADDRESS |

6.2.3 ERROR TITLE TYPE C

1. ILLEGAL VECTOR MEM ADDR ER

| ERRPC | TSTNUM | TESTPC | BUSADR | VAM | ADDR |
|---|---|---|---|---|---|
| XXXXXX | XXXXXX | XXXXXX | XXXXXX | XXXXXX | XXXXXX |

| | |
|---|---|
| ERRPC | ADDRESS WHERE ERROR WAS DETECTED |
| TSTNUM | TEST NUMBER WHERE THE ERROR OCCURED |
| TESTPC | ADDRESS OF TEST THAT WAS RUNNING WHEN DRV11J VECTORED TO THE WRONG ADDRESS. |
| BUSADR | BUS ADDRESS OF CONCERNED OPERATION |
| VAM | VALUE OF VECTOR ADDRESS MEMORY BYTE COUNT AND LEVEL. |
| ADDR | CORRECT VECTOR ADDRESS |

6.2.4 ERROR TYPE D

THIS ERROR IS ONLY FOUND WHEN RUNNING THE VECTOR ADDRESS UNIQUENESS TESTS. THE VECTOR MEMORY IS PRESET TO PERFORM 64 INTERRUPTS. INTERRUPT SERVICE ROUTINES ARE STORED FROM VECTORS 300 TO 674. THE ROUTINES WILL STORE THE VALUE OF THE VECTOR ADDRESS WHERE THEY CAME FROM IN A BUFFER TO COMPARE AFTER ALL INTERRUPTS TAKE PLACE. THE BUFFER SHOULD THEN HAVE ADDRESS VALUES IN CONSECUTIVE ORDER STARTING WITH VECTOR VALUE 300 AND ENDING WITH

VECTOR VALUE 674.

1. VECT ADDR MEM ER
   ERRPC  TSTNUM  BUSADR  EXPCT  RCVD
   XXXXXX XXXXXX  XXXXXX  XXXXXX XXXXXX

   ERRPC  ADDRESS WHERE ERROR WAS DETECTED
   TSTNUM  TEST NUMBER WHERE ERROR OCCURRED
   BUSADR  STARTUP BUS ADDRESS BEFORE VECTOR UNIQUENESS
   EXPCT  THIS IS THE EXPECTED VECTOR ADDRESS
   RCVD  VECTOR ADDRESS VALUE STORED IN BUFFER

## 7.0 MISCELLANEOUS

### 7.1 DRV11-J BUS ADDRESS MODIFICATION

MODIFY LOCATION '$BASE'(ADDR:2244) 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:2246) WITH A BIT MAP INDICATING WHAT INTERFACES ARE TO BE TESTED.
I.E. BIT0= 1 SAYS TEST 1ST DRV11J,
BIT1= 1 SAYS TEST 1ST DRV11J
BIT2= 1 SAYS TEST 2ND DRV11J
BIT3= 1 SAYS TEST 3RD 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

ALWAYS TYPE ''CTRL G'' AND WAIT FOR ECHO OF CHARACTERS BEFORE HALTING THE CPU OTHERWISE PORTIONS OF THE PROGRAM MAY NOT GET RESTORED. FOR MORE INFORMATION REFER TO SECTION 5.2.

## 8.0 EXECUTION TIME

EXECUTION TIME WILL BE LESS THAN 30 SECONDS 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.

REFERENCES ARE MADE IN THE LISTING TO THE TWO INTERRUPT
CONTROL CHIPS USED ON THE DRV11J AS GROUP1 AND GROUP2.

| | |
|---|---|
| GROUP1 CHIP | CSRA = CONTROL FOR GROUP 1 |
| . | CSRB = DATA FOR GROUP1 |
| GROUP2 CHIP | CSRC = CONTROL FOR GROUP2 |
| | CSRD = DATA FOR GROUP2 |

### 9.1 TEST 1 - TEST CHIP INTERRUPT,GROUP1,GROUP2

THIS TEST WILL CHECK THE ABILITY OF EACH INTERRUPT
CONTROL CHIP TO INTERRUPT TO COMMON VECTOR 300.
THIS TEST WILL ALSO CLEAR THE HIGHEST PRIORITY
ISR BITS WITH TEST VARIATIONS OF IRR AND
IMR BITS.

### 9.2 TEST 2 - TEST ACR/ISR INTERRUPT,GROUP 1,GROUP 2

THIS TEST WILL TEST THE ABILITY OF BOTH GROUPS
TO INTERRUPT TO COMMON VECTOR 300 AND THE ABILITY
OF THE ACR(AUTOCLEAR REGISTER TO CLEAR THE ISR
AFTER BEING SET AND MASKED BY COMBINATIONS OF IRR BITS
AND IMR BITS.

### 9.3 TEST 3 - TEST VECTOR ADDR MEM,LEV 0-7,BY0-3,(204-1774),GRPS 1,2

THIS TEST WILL CHECK BOTH GROUPS,ONE AT A TIME,TO VECTOR
FROM ADDRESS 204 TO ADDRESS 1774 FOR LEVELS 0-7 AND
FOR BYTE COUNTS 1-4.THE BYTE COUNT NUMBER WILL INDICATE
THE NUMBER OF INTERRUPTS AT A GIVEN ADDRESS.
THE IRR,IMR AND ACR REGISTERS WILL ALL BE EXERCISED
AND CHECKED TO INSURE PROPER VECTORING.

### 9.4 TEST 4 - TEST VECTOR ADDR MEM,LEV 0-7,BY0-3,(0-200),GRPS 1,2

THIS TEST WILL CHECK BOTH GROUPS,ONE AT A TIME, TO VECTOR
FROM ADDRESS 0-200 FOR LEVELS 0-7 AND FOR BYTE COUNTS 1-4.
TESTS ARE DONE TO INSURE THAT INTERRUPTS ONLY TAKE PLACE
WHEN EVERYTHING IS READY. THE CHIP BEING ARMED,INTERRUPT
ENABLE IS SET AND THE PROPER IMR AND IRR BITS ARE SET.
AFTER THE INTERRUPTS CHECK TO INSURE THAT THE INTERRUPTS
GENERATED MATCH THE BYTE COUNT AND THAT THE PROPER

ISR BITS WERE SET AND THE PROPER IRR BITS WERE CLEARED.

9.5 TEST 5 - TEST VECTOR ADDR UNIQUENESS IN FIXED MODE FOR GROUPS 1,2

THIS TEST WILL CHECK THE FIXED PRIORITY OF THE TWO
INTERRUPT CONTROL CHIPS AS WELL AS THE ADDRESSING
CAPABILITY OF THE VECTOR ADDRESS MEMORY.
THE AUTOCLEAR REGISTER(ACR) IS USED TO CLEAR OUT THE
ISR BITS AFTER 4 INTERRUPTS PER LEVEL HAS TAKEN PLACE.
CLEARING THE ISR REGISTER WILL THEN ALLOW THE NEXT LEVEL
OF PRIORITY TO INITIATE 4 INTERRUPTS.
THE VECTOR MEMORY IS PRESET TO PERFORM 64 UNIQUE INTERRUPTS.
INTERRUPT SERVICE ROUTINES ARE STORED FROM VECTORS
300-674. EACH INTERRUPT SERVICE ROUTINE UPON INTERRUPT
ENTRY WILL STORE THE ADDRESS OF ITS VECTOR
IN A BUFFER FOR A COMPARE AFTER 64 INTERRUPTS TAKE PLACE.
THE BUFFER SHOULD THEN HAVE ADDRESS VALUES IN CONSECUTIVE
ORDER STARTING WITH VECTOR VALUE 300 AND ENDING WITH
VECTOR VALUE 674.

9.6 TEST 6 - TEST VECTOR ADDRESS UNIQUENESS,ROTATING PRIORITY
FOR BOTH GROUPS 1,2

THIS TEST WILL CHECK THE ROTATING PRIORITY OF EACH
OF THE TWO INTERRUPT CONTROL CHIPS AS WELL AS THE ADDRESSING
CAPABILITY OF THE VECTOR ADDRESS MEMORY.
THE VECTOR ADDRESS MEMORY IS PRESET FOR 64 UNIQUE INTERRUPTS.
INTERRUPT SERVICE ROUTINES ARE STORED FROM VECTORS 300-674.
ALL IRR BITS ARE SET AND ALL IMR BITS ARE CLEARED.
AFTER THE FIRST FOUR INTERRUPTS FROM IRR0 TAKES PLACE,
THE ISR0 BIT WILL SET AND IRR0 BIT WILL CLEAR.
TO PROVE ROTATION OF PRIORITY,THE ISR0 BIT IS CLEARED
AND THE IRR0 BIT IS SET AGAIN IN ORDER TO PROVE THAT
IRR1 WILL NOW BE CONSIDERED THE HIGHEST PRIORITY AND
IRR0 THE LOWEST PRIORITY. THIS FLOW IS REPEATED FOR
ALL LEVELS OF GROUP 1(0-7). THEN GROUP 2 IS TESTED
IN THE SAME MANNER FOR LEVELS 0-7. EACH INTERRUPT
SERVICE ROUTINE UPON ACCESS FROM AN INTERRUPT WILL STORE
THE ADDRESS OF THE VECTOR WHERE THE ROUTINE RESIDES IN
A BUFFER. AFTER ALL 64 INTERRUPTS TAKE PLACE(4 AT A TIME
FOR EACH LEVEL)THE BUFFER WILL BE CHECKED FOR CONSECUTIVE
VECTOR VALUES OF 300-674. THIS WILL INSURE THAT PROPER
VECTOR ADDDRESSING AND ROTATION OF PRIORITIES WAS
COMPLETED.

10.0 LISTING
-------

```
                        .TITLE  CVDRDA DRV11J DIAG TST PRT2
                        ;*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-C3), JAN 19, 1977.
                        ;*
000001                  $TN=1
160000                  $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
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
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
```

```
                                PR0=    0               ;;PRIORITY LEVEL 0
                                PR1=    40              ;;PRIORITY LEVEL 1
                                PR2=    100             ;;PRIORITY LEVEL 2
                                PR3=    140             ;;PRIORITY LEVEL 3
                                PR4=    200             ;;PRIORITY LEVEL 4
                                PR5=    240             ;;PRIORITY LEVEL 5
                                PR6=    300             ;;PRIORITY LEVEL 6
                                PR7=    340             ;;PRIORITY LEVEL 7

                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
                                SW15=   100000
                                SW14=   40000
                                SW13=   20000
                                SW12=   10000
                                SW11=   4000
                                SW10=   2000
                                SW09=   1000
                                SW08=   400
                                SW07=   200
                                SW06=   100
                                SW05=   40
                                SW04=   20
                                SW03=   10
                                SW02=   4
                                SW01=   2
                                SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                                BIT15=  100000
                                BIT14=  40000
                                BIT13=  20000
                                BIT12=  10000
                                BIT11=  4000
                                BIT10=  2000
                                BIT09=  1000
                                BIT08=  400
                                BIT07=  200
                                BIT06=  100
                                BIT05=  40
                                BIT04=  20
                                BIT03=  10
                                BIT02=  4
                                BIT01=  2
                                BIT00=  1
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
```

```
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;''T'' BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;''TRAP'' TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
164160                  ABASE=  164160          ;BASE ADDRESS
000001                  ADEVM=  1               ;DEFAULT TO ONE DRV11J
100000                  RDY=    BIT15
000400                  DIR=    BIT8
001000                  IE=     BIT9

                        ;CHIP COMMAND SUMMARY
000020                  CIRMR=  20              ;CLEAR IRR AND IMR
000030                  CSIRMR= 30              ;CLEAR SINGLE IRR AND IMR BIT

000040                  CIMR=   40              ;CLEAR IMR
000050                  CSIMR=  50              ;CLEAR SINGLE IMR BIT
000060                  SIMR=   60              ;SET ALL IMR BITS
000070                  SSIMR=  70              ;SET SINGLE IMR BITS

000100                  CIRR=   100             ;CLEAR IRR
000110                  CSIRR=  110             ;CLEAR SINGLE IRR BITS
000120                  SIRR=   120             ;SET ALL IRR BITS
000130                  SSIRR=  130             ;SET SINGLE IRR BITS

000140                  CHPISR= 140             ;CLEAR HIGHEST PRIORITY ISR BIT
000160                  CISR=   160             ;CLEAR ISR
000170                  CSISR=  170             ;CLEAR SINGLE ISR BIT

000200                  LMD04=  200             ;LOAD MODE BITS M0-M4
000240                  LMD57=  240             ;LOAD MODE BITS M5-M7

                        ;CHIP MODE BIT PRESELECTION
000240                  MISR=   240
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

                        000000                          .=0
                                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                  .=174
000174  000000                                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  002476                                  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  010614                  $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
002000                          .=2000

                                                                ;LONGEST TEST TIME
                                                                ;1ST PASS RUN TIME
                                                                ;ADDITIONAL RUN TIME
                        .SBTTL  APT PARAMETER BLOCK

                        ;;**********************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                        ;;**********************************************************************
002000                          .$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  002000                  $APTHDR ;;POINT TO APT HEADER BLOCK
002000                          .=.$X   ;;RESET LOCATION COUNTER
                        ;;**********************************************************************
                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                        ;INTERFACE SPEC.

002000                  $APTHD:
002000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
002002  002170          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
002004  000014          $TSTM:  .WORD   12.     ;;RUN TIM OF LONGEST TEST
002006  000036          $PASTM: .WORD   30.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
002010  000036          $UNITM: .WORD   30.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
002012  000031                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                        .SBTTL  COMMON TAGS

                        ;;*****************************************************************
                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                        ;*USED IN THE PROGRAM.

          002100                .=2100
002100                  $CMTAG:                         ;;START OF COMMON TAGS
002100    000000                .WORD   0
002102      000         $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
002103      000         $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
002104    000000        $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
002106    000000        $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
002110    000000        $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
002112    000000        $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
002114      000         $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
002115      001         $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
002116    000000        $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
002120    000000        $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
002122    000000        $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
002124    000000        $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
002126    000000        $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
002130    000000                .WORD   0               ;;RESERVED--NOT TO BE USED
002132    000000                .WORD   0
002134      000         $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
002135      000         $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
002136    000000                .WORD   0
002140    177570        SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
002142    177570        DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
002144    177560        $TKS:   177560                  ;;TTY KBD STATUS
002146    177562        $TKB:   177562                  ;;TTY KBD BUFFER
002150    177564        $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
002152    177566        $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
002154      000         $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
002155      002         $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
002156      012         $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
002157      000         $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
002160    000000        $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
002162    000000        $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
002164      077         $QUES:  .ASCII  /?/             ;;QUESTION MARK
002165      015         $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
002166    000012        $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;*****************************************************************
                        .SBTTL  APT MAILBOX-ETABLE

                        ;;*****************************************************************
                        .EVEN
002170                  $MAIL:                          ;;APT MAILBOX
002170    000000        $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
002172    000000        $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
002174    000000        $TESTN: .WORD   ATESTN          ;;TEST NUMBER
002176    000000        $PASS:  .WORD   APASS           ;;PASS COUNT
002200    000000        $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
002202    000000        $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
002204    000000        $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
002206    000000        $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
```

```
279  002210                  $ETABLE:                ;;APT ENVIRONMENT TABLE
280  002210  000             $ENV:    .BYTE  AENV    ;;ENVIRONMENT BYTE
281  002211  000             $ENVM:   .BYTE  AENVM   ;;ENVIRONMENT MODE BITS
282  002212  000000          $SWREG:  .WORD  ASWREG  ;;APT SWITCH REGISTER
283  002214  000000          $USWR:   .WORD  AUSWR   ;;USER SWITCHES
284  002216  000000          $CPUOP:  .WORD  ACPUOP  ;;CPU TYPE,OPTIONS
285                          ;*                      BITS 15-11=CPU TYPE
286                          ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
287                          ;*                              11/70=06,PDQ=07,Q=10
288                          ;*                      BIT 10=REAL TIME CLOCK
289                          ;*                      BIT  9=FLOATING POINT PROCESSOR
290                          ;*                      BIT  8=MEMORY MANAGEMENT
291  002220  000             $MAMS1:  .BYTE  AMAMS1  ;;HIGH ADDRESS,M.S.#BYTE
292  002221  000             $MTYP1:  .BYTE  AMTYP1  ;;MEM. TYPE,BLK#1
293                          ;*                      MEM.TYPE BYTE    --  (HIGH BYTE)
294                          ;*                              900 NSEC CORE=001
295                          ;*                              300 NSEC BIPOLAR=002
296                          ;*                              500 NSEC MOS=003
297  002222  000000          $MADR1:  .WORD  AMADR1  ;;HIGH ADDRESS,BLK#1
298                          ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
299  002224  000             $MAMS2:  .BYTE  AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
300  002225  000             $MTYP2:  .BYTE  AMTYP2  ;;MEM.TYPE,BLK#2
301  002226  000000          $MADR2:  .WORD  AMADR2  ;;MEM.LAST ADDRESS,BLK#2
302  002230  000             $MAMS3:  .BYTE  AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
303  002231  000             $MTYP3:  .BYTE  AMTYP3  ;;MEM.TYPE,BLK#3
304  002232  000000          $MADR3:  .WORD  AMADR3  ;;MEM.LAST ADDRESS,BLK#3
305  002234  000             $MAMS4:  .BYTE  AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
306  002235  000             $MTYP4:  .BYTE  AMTYP4  ;;MEM.TYPE,BLK#4
307  002236  000000          $MADR4:  .WORD  AMADR4  ;;MEM.LAST ADDRESS,BLK#4
308  002240  000000          $VECT1:  .WORD  AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
309  002242  000000          $VECT2:  .WORD  AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
310  002244  164160          $BASE:   .WORD  ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
311  002246  000001          $DEVM:   .WORD  ADEVM   ;;DEVICE MAP
312  002250  000000          $CDW1:   .WORD  ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
313  002252                  $ETEND:
314                          .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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

002252                          $ERRTB:
                                ;ERROR  1
002252  016031                          EM1             ;REG TIMEOUT ER
002254  016354                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
002256  016624                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
002260  000000                          0

                                ;ERROR  2
002262  016050                          EM2     ;REG READ/WRITE ER
002264  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002266  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002270  000000                          0

                                ;ERROR  3
002272  016072                          EM3     ;IRR REG ER
002274  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002276  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002300  000000                          0

                                ;ERROR  4
002302  016105                          EM4     ;ACR REG ER
002304  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002306  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002310  000000                          0

                                ;ERROR  5
002312  016120                          EM5     ;IMR REG ERROR
002314  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002316  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002320  000000                          0

                                ;ERROR  6
002322  016133                          EM6     ;ISR REG ERROR
002324  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002326  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002330  000000                          0

                                ;ERROR  7
002332  016333                          EM14    ;VECTOR ADDR MEM ER
002334  016354                          DH1     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
002336  016624                          DT1     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
002340  000000                          0
```

```
                                ;ERROR  10
002342  016201                          EM10    ;CHIP STAT ER
002344  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002346  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002350  000000                          0

                                ;ERROR  11
002352  016216                          EM11    ;ILLEGAL INTERRUPT  RECEIVED
002354  016506                          DH3     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR
002356  016660                          DT3     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC
002360  000000                          0

                                ;ERROR  12
002362  016251                          EM12    ;INTERRUPT TEST ERROR
002364  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002366  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002370  000000                          0

                                ;ERROR  13      ;MULTIPLE INTERRUPTS RECEIVED
002372  016276                          EM13    ;MULTIPLE INTERRUPTS RECEIVED
002374  016421                          DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002376  016640                          DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002400  000000                          0

                                ;ERROR  14
002402  016146                          EM7     ;ILLEGAL VECTOR ADDR MEM ERROR
002404  016553                          DH4     ;ERRPC  TSTNUM  TESTPC  BUSADR  VAM     ADDR
002406  016674                          DT4     ;$ERRPC TSTNUM  $GDADR  $BDADR  VECVAL  VECLOC
002410  000000                          0


                                ; BUS REGISTER ADDRESS POINTERS

002412  164160                  DRCSA:  ABASE
002414  164162                  DRDBA:  ABASE+2
002416  164164                  DRCSB:  ABASE+4
002420  164166                  DRDBB:  ABASE+6
002422  164170                  DRCSC:  ABASE+10
002424  164172                  DRDBC:  ABASE+12
002426  164174                  DRCSD:  ABASE+14
002430  164176                  DRDBD:  ABASE+16


                                ;COMMON PROGRAM LOCATION(S)

002432  000000                  TSTNUM: 0               ;CONTAINS TEST NUMBER ON ERROR
002434  000001                  DMAP:   1
002436  000000                  INTFLG: .WORD   0
002440  000000                  XXDP:   .WORD   0
002442  000000                  IMRLOC: .WORD   0
002444  000000                  ISRLOC: .WORD   0
002446  000000                  IRRLOC: .WORD   0
002450  000000                  ACRLOC: .WORD   0
002452  000000                  VECLOC: .WORD   0
```

ERROR POINTER TABLE

```
  427  002454  000000                VECPAT: .WORD   0
  428  002456  000000                VECVAL: .WORD   0
  429  002460  000000                SWRSAV: .WORD   0
  430  002462  000000                GRPCNT: .WORD   0
  431  002464  000000                BDSAV:  .WORD   0
  432  002466  000000                LVLCNT: .WORD   0
  433  002470  000000                BYCNT:  .WORD   0
  434  002472  000000                BYNUM:  .WORD   0
  435  002474  000000                LVLSAV: .WORD   0
```

```
                                        .SBTTL  PROGRAM START
002476                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002476  012706  002100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002502  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002504  022706  002140                  CMP     #SWR,R6 ;;DONE?
002510  001374                          BNE     .-6             ;;LOOP BACK IF NO
002512  012706  002100                  MOV     #2100,SP                ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002516  012737  014416  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002524  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002532  012737  014070  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002540  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002546  012737  015656  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002554  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002562  012737  015452  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002570  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002576  005037  002160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002602  005037  002162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002606  112737  000001  002115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002614  012737  002614  002106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002622  012737  002622  002110          MOV     #.,$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.
002630  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002634  012737  002670  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002642  012737  177570  002140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002650  012737  177570  002142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002656  022777  177777  177254          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002664  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002666  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002670  012716  002676          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002674  000002                          RTI
002676  012737  000176  002140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002704  012737  000174  002142          MOV     #DISPREG,DISPLAY
002712  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002716  005037  002176                  CLR     $PASS           ;;CLEAR PASS COUNT
002722  132737  000200  002211          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002730  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002732  012737  002212  002140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002740                          67$:
002740  005037  002172                  CLR     $FATAL          ;CLEAR ERROR NUMBER
002744  005037  002170                  CLR     $MSGTYP         ;CLEAR MESSAGE TYPE
002750  005037  002174                  CLR     $TESTN          ;CLEAR TEST NUMBER
                                ;CHECK OPERATING ENVIRONMENT
002754  005737  000042                  TST     @#42            ;ARE WE IN ACT/XXDP AUTO MODE?
002760  001410                          BEQ     1$              ;BRANCH IF NO
002762  023737  000042  000046          CMP     @#42,@#46       ;IS IT ACT AUTO MODE?
002770  001410                          BEQ     2$              ;BRANCH IF YES
002772  012737  177777  002440          MOV     #-1,XXDP        ;SET XXDP CHAIN MODE INDICATOR
003000  000404                          BR      2$
003002  123727  002210  000001  1$:     CMPB    $ENV,#1         ;ARE WE IN APT AUTO MODE?
003010  001003                          BNE     3$              ;BRANCH IF NO
```

```
  492  003012  112737  000001  002134  2$:     MOVB    #1,$AUTOB       ;SET AUTO MODE INDICATOR
  493
  494                                         ;PRINT TITLE IF NOT IN ACT OR APT AUTO MODE
  495  003020  005227  177777          3$:     INC     #-1             ;FIRST TIME?
  496  003024  001014                          BNE     5$              ;SKIP TITLE IF NO
  497  003026  004537  011164                  JSR     R5,TOBUF        ;STORE 0-202 INTO BUFFER AREA
  498  003032  005737  002134                  TST     $AUTOB          ;ARE WE IN AUTO MODE?
  499  003036  001403                          BEQ     4$              ;BRANCH TO TITLE TYPEOUT IF NOT
  500  003040  005737  002440                  TST     XXDP            ;IS THE AUTO MODE UNDER XXDP?
  501  003044  001404                          BEQ     5$              ;SKIP TITLE IF NOT
  502  003046  104401  015736          4$:     TYPE    ,TITLED         ;PRINT OUT THE TITLE
  503  003052  104401  016002                  TYPE    ,TLCABL         ;PRINT DRV11J CABLE REQ'D
  504
  505                                         ;GET THE VALUE IN THE SOFTWARE SWITCH REGISTER
  506  003056  005737  002134          5$:     TST     $AUTOB          ;ARE WE IN AUTOMATIC MODE?
  507  003062  001001                          BNE     START1          ;BRANCH IF YES
  508  003064  104406                          GTSWR                   ;ASK FOR SWR INPUT FROM CONSOLE
  509  003066  005037  002202          START1: CLR     $UNIT           ;CLEAR UNIT NUMBER
  510  003072  013737  002246  002434          MOV     $DEVM,DMAP      ;UP TO 4 DRV11J'S
  511  003100  042737  177760  002434          BIC     #177760,DMAP    ;POSITION OF DRV11J'S
  512  003106  013701  002244                  MOV     $BASE,R1        ;GET BASE ADDRESS
  513  003112  010137  002412                  MOV     R1,DRCSA        ;MAKE BUS REG.POINTER = BASE
  514  003116  032737  000001  002434          BIT     #1,DMAP         ;IS FIRST DRV11-J SELECTED
  515  003124  001002                          BNE     NEXPAS          ;YES
  516  003126  000137  010472                  JMP     NXDEV1          ;NO,ADVANCE BASE DRV11J ADDRESS
  517  003132  012700  002412          NEXPAS: MOV     #DRCSA,R0       ;SET UP REGISTER ADDRESS POINTERS
  518  003136  010120                  NEXPA1: MOV     R1,(R0)+        ;LOAD EM
  519  003140  062701  000002                  ADD     #2,R1           ;
  520  003144  022700  002432                  CMP     #DRDBD+2,R0     ;ALL DONE?
  521  003150  001372                          BNE     NEXPA1          ;BR IF NOT
  522  003152  004537  011334                  JSR     R5,TRPCAT       ;EXTEND TRAP CATCHER 204-1774
  523  003156  012706  002100                  MOV     #2100,SP        ;ALWAYS RESET STACK
  524  003162  013737  002202  002200          MOV     $UNIT,$DEVCT    ;LOAD APT COUNTER WITH UNIT #
  525  003170  123727  002210  000001          CMPB    $ENV,#1         ;ARE WE IN APT MODE?
  526  003176  001401                          BEQ     NEXPA2          ;BR IF YES,SKIP RESET
  527  003200  000005                          RESET                   ;INITIALIZE
  528  003202  106427  000340          NEXPA2: MTPS    #PR7
  529
```

```
                                ;;*****************************************************************
                                ;*TEST 1          TEST CHIP INTERRUPT ,GROUP 1,GROUP 2
                                ;;*****************************************************************
003206  000004                  TST1:   SCOPE
003210  013700  002412                  MOV     DRCSA,R0        ;START WITH GROUP 1
003214  013701  002416                  MOV     DRCSB,R1
003220  013702  002412                  MOV     DRCSA,R2        ;REGISTER IS COMMON FOR BOTH GROUPS
003224  012737  003276  002110          MOV     #111$,$LPERR    ;SET UP LOOP RETURN
003232  012737  000002  002462          MOV     #2,GRPCNT       ;COUNTER FOR BOTH GROUPS
003240  004537  010650          120$:   JSR     R5,CLRCSR       ;CLEAR ALL CSRS
003244  012703  012644                  MOV     #BGCHP3,R3      ;EXPECTED GDDAT FOR ISR,IMR
003250  012737  000050  002442          MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR CODE
003256  004537  010730                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
003262  012737  011454  000300          MOV     #INTSR1,300     ;STORE VECTOR ROUTINE
003270  012737  000340  000302          MOV     #340,302        ;STORE PSW
003276  005037  002436          111$:   CLR     INTFLG          ;CLEAR INT. FLAG
003302  012706  002100                  MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
003306  112710  000340                  MOVB    #PVMA,(R0)      ;PRESELECT VECTOR MEM ADDR
003312  112711  000060                  MOVB    #60,(R1)        ;WRITE VECTOR 300,BY0,LV0
003316  010037  002122                  MOV     R0,$BDADR       ;SAVE BUS ADDRESS
003322  112710  000060                  MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
003326  113710  002442                  MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
003332  112710  000202                  MOVB    #202,(R0)       ;LMD04 - COMMON VECTOR
003336  106427  000000                  MTPS    #PR0
003342  112712  000241                  MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
                                                                ;ARM GROUP 1 TO PASS
                                                                ;ENABLE TO GROUP 2.
003346  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
003352  005737  002436                  TST     INTFLG          ;CHECK FOR INTERRUPTS
003356  001401                          BEQ     1$
003360  104011                          ERROR   11              ;ERROR,ILLEGAL INTERRUPT
003362  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
003366  112710  000120                  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
003372  106427  000340                  MTPS    #PR7
003376  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
003402  001401                          BEQ     2$
003404  104011                          ERROR   11              ;ILLEGAL INTERRUPTS
003406  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
003412  106427  000000                  MTPS    #PR0
003416  152762  000002  000001          BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
003424  012737  000001  002124          MOV     #1,$GDDAT       ;EXPECT ONE INTERRUPT
003432  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE ONE INTERRUPT
003440  023737  002124  002126          CMP     $GDDAT,$BDDAT
003446  001401                          BEQ     3$
003450  104012                          ERROR   12              ;INTERRUPT TEST ERROR
003452  106427  000340          3$:     MTPS    #PR7
003456  012737  101710  002124          MOV     #101710,$GDDAT  ;RDY,DIR,I/E, CHIP - 31X
003464  010237  002122                  MOV     R2,$BDADR       ;CSRA ADDRESS
003470  011237  002126                  MOV     (R2),$BDDAT
003474  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
003502  023737  002124  002126          CMP     $GDDAT,$BDDAT
003510  001401                          BEQ     4$
003512  104002                          ERROR   2               ;CSRA REG ERROR
003514  010137  002122          4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
003520  005037  002126                  CLR     $BDDAT          ;SET UP FOR BYTE READS
```

```
  586  003524  005037  002124                      CLR     $GDDAT                  ;SET UP FOR BYTE EXPECTED
  587  003530  111337  002124                      MOVB    (R3),$GDDAT             ;EXPECTED DATA
  588  003534  112710  000240                      MOVB    #MISR,(R0)              ;LOAD MODE TO READ ISR
  589  003540  111137  002126                      MOVB    (R1),$BDDAT
  590  003544  023737  002124  002126              CMP     $GDDAT,$BDDAT
  591  003552  001401                              BEQ     5$
  592  003554  104006                              ERROR   6                       ;ISR ERROR
  593  003556  116337  000001  002124  5$:         MOVB    1(R3),$GDDAT            ;STORE EXPECTED
  594  003564  112710  000244                      MOVB    #MIMR,(R0)              ;LOAD MODE FOR IMR
  595  003570  111137  002126                      MOVB    (R1),$BDDAT
  596  003574  023737  002124  002126              CMP     $GDDAT,$BDDAT
  597  003602  001401                              BEQ     6$
  598  003604  104005                              ERROR   5                       ;IMR ERROR
  599  003606  116337  000001  002124  6$:         MOVB    1(R3),$GDDAT            ;EXPECTED IRR
  600  003614  112710  000250                      MOVB    #MIRR,(R0)              ;LOAD MODE BITS FOR IRR
  601  003620  111137  002126                      MOVB    (R1),$BDDAT
  602  003624  023737  002124  002126              CMP     $GDDAT,$BDDAT
  603  003632  001401                              BEQ     7$
  604  003634  104003                              ERROR   3                       ;IRR ERROR
  605  003636  005037  002124            7$:       CLR     $GDDAT
  606  003642  112710  000254                      MOVB    #MACR,(R0)
  607  003646  111137  002126                      MOVB    (R1),$BDDAT
  608  003652  023737  002124  002126              CMP     $GDDAT,$BDDAT
  609  003660  001401                              BEQ     10$
  610  003662  104004                              ERROR   4                       ;ACR ERROR
  611  003664  005037  002124           10$:       CLR     $GDDAT
  612  003670  112710  000140                      MOVB    #CHPISR,(R0)            ;CLEAR HIGHEST PRIOR ISR
  613  003674  112710  000240                      MOVB    #MISR,(R0)              ;LOAD MODE TO READ ISR
  614  003700  111137  002126                      MOVB    (R1),$BDDAT
  615  003704  023737  002124  002126              CMP     $GDDAT,$BDDAT
  616  003712  001401                              BEQ     11$
  617  003714  104006                              ERROR   6                       ;ISR REG ERROR
  618  003716  012737  101710  002124  11$:        MOV     #101710,$GDDAT          ;RDY,DIR,I/E, CHIP - 31X
  619  003724  010237  002122                      MOV     R2,$BDADR               ;CSRA ADDRESS
  620  003730  011237  002126                      MOV     (R2),$BDDAT
  621  003734  042737  000007  002126              BIC     #7,$BDDAT               ;CLEAR UNDEFINED BITS
  622  003742  023737  002124  002126              CMP     $GDDAT,$BDDAT
  623  003750  001401                              BEQ     12$
  624  003752  104002                              ERROR   2                       ;CSRA REG ERROR
  625  003754  012737  100710  002124  12$:        MOV     #100710,$GDDAT          ;RDY,DIR, CHIP - 31X
  626  003762  042712  001000                      BIC     #BIT9,(R2)              ;CLEAR I/E
  627  003766  011237  002126                      MOV     (R2),$BDDAT             ;READ CSRA
  628  003772  042737  000007  002126              BIC     #7,$BDDAT               ;CLEAR UNDEFINED BITS
  629  004000  023737  002124  002126              CMP     $GDDAT,$BDDAT
  630  004006  001401                              BEQ     13$
  631  004010  104002                              ERROR   2                       ;CSRA REG ERROR
  632  004012  005723                    13$:      TST     (R3)+                   ;GET NEXT ISR,IMR EXPECTED RESULTS
  633  004014  020327  012664                      CMP     R3,#EDCHP3              ;CHECK FOR END
  634  004020  001404                              BEQ     14$
  635  004022  005237  002442                      INC     IMRLOC                  ;INCREMENT MASK BIT TO CLEAR
  636  004026  000137  003276                      JMP     111$                    ;TEST NEXT ISR BIT
  637  004032  005337  002462           14$:       DEC     GRPCNT                  ;TESTED BOTH GROUPS?
  638  004036  001406                              BEQ     TST2                    ;;BR IF DONE
  639  004040  013700  002422                      MOV     DRCSC,R0                ;SETUP FOR GROUP 2
  640  004044  013701  002426                      MOV     DRCSD,R1                ;CSRC = CONTROL GROUP 2
  641                                                                              ;CSRD = DATA  GROUP 2
```

```
                                JMP     120$            ;DO GROUP 2
                        ;;*****************************************************************
                        ;*TEST 2         TEST ACR/ISR INTERRUPT ,GROUP 1,GROUP 2.
                        ;;*****************************************************************
                        TST2:   SCOPE
                                MOV     DRCSA,R0
                                MOV     DRCSB,R1
                                MOV     DRCSA,R2
                                MOV     #111$,$LPERR    ;SET UP LOOP RETURN.
                                MOV     #2,GRPCNT       ;DO TWO GROUPS
                        120$:   JSR     R5,CLRCSR       ;CLEAR ALL CSRS
                                MOV     #BGCHP3,R3      ;EXPECTED GDDAT FOR ISR
                                MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR CODE
                                JSR     R5,CLRIRR       ;CLEAR IRR REGS
                                MOV     #INTSR1,300     ;STORE VECTOR ROUTINE
                                MOV     #340,302        ;STORE PSW
                        111$:   CLR     INTFLG          ;CLEAR INT. FLAG
                                MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
                                BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
                                MOVB    #242,(R2)       ;CLEAR MASTER MASK GP1
                                MOVB    #242,(R0)       ;CLEAR MASTER MASK GP1,GP2
                                MOVB    #PACR,(R0)      ;PRESELECT ACR FOR WRITING
                                MOVB    (R3),(R1)       ;NEXT BIT INTO ACR
                                MOVB    #PVMA,(R0)      ;PRESELECT VECTOR MEM ADDR
                                MOVB    #60,(R1)        ;WRITE VECTOR 300,BY0,LV0
                                MOV     R0,$BDADR       ;STORE BUS ADDRESS
                                MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
                                MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
                                MOVB    #202,(R0)       ;LMD04 - COMMON VECTOR
                                MTPS    #PR0
                                BISB    #BIT1,1(R2)     ;SET I/E,NO INTERRUPT
                                TST     INTFLG          ;CHECK FOR INTERRUPTS
                                BEQ     1$
                                ERROR   11              ;ERROR,ILLEGAL INTERRUPT
                        1$:     CLR     INTFLG          ;RESET INT. COUNTER
                                MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
                                MTPS    #PR7
                                TST     INTFLG          ;CHECK FOR NO INTERRUPTS
                                BEQ     2$
                                ERROR   11              ;ILLEGAL INTERRUPTS
                        2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
                                MTPS    #PR0
                                MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
                                                        ;ARM GROUP 1 CHIP TO PASS
                                                        ;ENABLE TO GROUP 2 CHIP.
                                MOVB    #241,(R0)       ;ARM THE CHIP
                                MOV     #1,$GDDAT       ;EXPECT ONE INTERRUPT
                                MOV     INTFLG,$BDDAT   ;SHOULD HAVE ONE INTERRUPT
                                CMP     $GDDAT,$BDDAT
                                BEQ     3$
                                ERROR   12              ;INTERRUPT TEST ERROR
                        3$:     MTPS    #PR7
                        4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
                                CLR     $BDDAT          ;SET UP FOR BYTE READS
```

```
  698  004356  005037  002124                    CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
  699  004362  112710  000240                    MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
  700  004366  111137  002126                    MOVB    (R1),$BDDAT
  701  004372  023737  002124  002126            CMP     $GDDAT,$BDDAT   ;ACR SHOULD CLEAR ISR
  702  004400  001401                            BEQ     5$
  703  004402  104006                            ERROR   6               ;ISR ERROR
  704  004404  116337  000001  002124  5$:       MOVB    1(R3),$GDDAT    ;STORE EXPECTED
  705  004412  112710  000244                    MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
  706  004416  111137  002126                    MOVB    (R1),$BDDAT
  707  004422  023737  002124  002126            CMP     $GDDAT,$BDDAT
  708  004430  001401                            BEQ     6$
  709  004432  104005                            ERROR   5               ;IMR ERROR
  710  004434  116337  000001  002124  6$:       MOVB    1(R3),$GDDAT    ;EXPECTED IRR
  711  004442  112710  000250                    MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  712  004446  111137  002126                    MOVB    (R1),$BDDAT
  713  004452  023737  002124  002126            CMP     $GDDAT,$BDDAT
  714  004460  001401                            BEQ     7$
  715  004462  104003                            ERROR   3               ;IRR ERROR
  716  004464  111337  002124          7$:       MOVB    (R3),$GDDAT     ;EXPECTED ACR
  717  004470  112710  000254                    MOVB    #MACR,(R0)
  718  004474  111137  002126                    MOVB    (R1),$BDDAT
  719  004500  023737  002124  002126            CMP     $GDDAT,$BDDAT
  720  004506  001401                            BEQ     10$
  721  004510  104004                            ERROR   4               ;ACR ERROR
  722  004512  005723                  10$:       TST     (R3)+           ;GET NEXT ISR,IMR,IRR EXPECTED RESULTS
  723  004514  020327  012664                    CMP     R3,#EDCHP3      ;CHECK FOR END
  724  004520  001404                            BEQ     11$             ;DONE WITH GROUP TEST
  725  004522  005237  002442                    INC     IMRLOC          ;INCREMENT MASK BIT TO CLEAR
  726  004526  000137  004144                    JMP     111$            ;TEST NEXT ISR BIT
  727  004532  005337  002462          11$:      DEC     GRPCNT          ;TESTED BOTH GROUPS?
  728  004536  001406                            BEQ     TST3            ;;BR IF DONE
  729  004540  013700  002422                    MOV     DRCSC,R0        ;SETUP FOR GROUP 2
  730  004544  013701  002426                    MOV     DRCSD,R1        ;CSRC = CONTROL GROUP 2
  731                                                                    ;CSRD = DATA    GROUP 2
  732  004550  000137  004106                    JMP     120$            ;DO GROUP 2
  733
  734
  735
  736                                  ;;*****************************************************************
  737                                  ;*TEST 3        TEST VECTOR ADDR MEM,LEV 0-7,BY0-3,(204-1774),GRPS 1,2
  738                                  ;;*****************************************************************
  739  004554  000004                  TST3:   SCOPE
  740  004556  013700  002412                    MOV     DRCSA,R0
  741  004562  013701  002416                    MOV     DRCSB,R1
  742  004566  013702  002412                    MOV     DRCSA,R2
  743  004572  012737  000002  002462            MOV     #2,GRPCNT       ;DO TWO GROUPS
  744  004600  012737  000001  002472  120$:     MOV     #1,BYNUM        ;INIT FOR BYTE COUNT 0
  745  004606  004537  010650                    JSR     R5,CLRCSR       ;CLEAR ALL CSRS
  746  004612  012737  000340  002456            MOV     #PVMA,VECVAL    ;START VECTOR LEVEL 0,BYTE COUNT 0
  747  004620  012704  012644          121$:     MOV     #BGCHP3,R4      ;EXPECTED ISR,IMR PATTERN
  748  004624  012737  000010  002466            MOV     #8.,LVLCNT      ;COUNTER FOR 0-7 LEVELS
  749  004632  012737  000050  002442            MOV     #CSIMR,IMRLOC   ;STORE SINGLE IMR CODE
  750  004640  012737  000130  002446            MOV     #SSIRR,IRRLOC   ;STORE SINGLE IRR CODE
  751  004646  012737  000170  002444            MOV     #CSISR,ISRLOC   ;STORE CLEAR SINGLE ISR CODE
  752  004654  004537  011334                    JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
  753  004660  012737  004674  002110            MOV     #112$,$LPERR    ;STORE SCOPE LOOP
```

```
  754  004666  012737  000204  002452  111$:   MOV     #204,VECLOC     ;START AT VECTOR 204
  755  004674  106427  000340          112$:   MTPS    #340            ;
  756  004700  012706  002100                  MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
  757  004704  042712  001000                  BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
  758  004710  004537  010730                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
  759  004714  112712  000241                  MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
  760                                                                  ;ARM GROUP 1 CHIP TO PASS
  761                                                                  ;ENABLE TO GROUP2
  762  004720  013703  002452                  MOV     VECLOC,R3       ;GET VECTOR
  763  004724  010337  002454                  MOV     R3,VECPAT
  764  004730  006037  002454                  ROR     VECPAT          ;ROTATE VECTOR ADDR TWO RIGHT
  765  004734  006037  002454                  ROR     VECPAT          ;TO WRITE VECTOR MEM ADDR.
  766  004740  012713  011510                  MOV     #INTBY4,(R3)    ;STORE VECTOR ROUTINE
  767  004744  012763  000340  000002          MOV     #340,2(R3)      ;STORE PSW
  768  004752  005037  002436                  CLR     INTFLG          ;CLEAR INT. FLAG
  769  004756  013737  002472  002470          MOV     BYNUM,BYCNT     ;BYTE COUNTS OF 0-3
  770  004764  113710  002456                  MOVB    VECVAL,(R0)     ;PRESELECT VECTOR MEM ADDR
  771  004770  113711  002454          113$:   MOVB    VECPAT,(R1)     ;WRITE VECTOR
  772  004774  005337  002470                  DEC     BYCNT           ;DONE WITH BYTE COUNT VALUE
  773  005000  001373                          BNE     113$            ;NO LOAD VECTOR FOR NEXT BYTE COUNT
  774  005002  010037  002122                  MOV     R0,$BDADR       ;SAVE BUS ADDRESS
  775  005006  112710  000060                  MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
  776  005012  113710  002442                  MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
  777  005016  106427  000000                  MTPS    #PR0
  778  005022  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
  779  005026  005737  002436                  TST     INTFLG          ;CHECK FOR INTERRUPTS
  780  005032  001401                          BEQ     1$
  781  005034  104011                          ERROR   11              ;ERROR,ILLEGAL INTERRUPT
  782  005036  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
  783  005042  113710  002446                  MOVB    IRRLOC,(R0)     ;SET SINGLE IRR BIT
  784  005046  106427  000340                  MTPS    #PR7
  785  005052  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
  786  005056  001401                          BEQ     2$
  787  005060  104011                          ERROR   11              ;ILLEGAL INTERRUPTS
  788  005062  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
  789  005066  106427  000000                  MTPS    #PR0
  790  005072  152762  000002  000001          BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
  791  005100  013737  002472  002124          MOV     BYNUM,$GDDAT    ;EXPECT 1 TO 4 INTERRUPTS
  792                                                                  ;BASED ON BYTE COUNT VALUE
  793  005106  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE 1 TO 4 INTERRUPTS
  794                                                                  ;BASED ON BYTE COUNT VALUE
  795  005114  023737  002124  002126          CMP     $GDDAT,$BDDAT
  796  005122  001401                          BEQ     3$
  797  005124  104012                          ERROR   12              ;INTERRUPT TEST ERROR
  798  005126  106427  000340          3$:     MTPS    #PR7
  799  005132  010137  002122          4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
  800  005136  005037  002126                  CLR     $BDDAT          ;SET UP FOR BYTE READS
  801  005142  005037  002124                  CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
  802  005146  111437  002124                  MOVB    (R4),$GDDAT     ;EXPECTED DATA
  803  005152  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
  804  005156  111137  002126                  MOVB    (R1),$BDDAT
  805  005162  023737  002124  002126          CMP     $GDDAT,$BDDAT
  806  005170  001401                          BEQ     5$
  807  005172  104006                          ERROR   6               ;ISR ERROR
  808  005174  116437  000001  002124  5$:     MOVB    1(R4),$GDDAT    ;STORE EXPECTED
  809  005202  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
```

```
  810  005206  111137  002126                MOVB    (R1),$BDDAT
  811  005212  023737  002124  002126        CMP     $GDDAT,$BDDAT
  812  005220  001401                        BEQ     6$
  813  005222  104005                        ERROR   5               ;IMR ERROR
  814  005224  005037  002124          6$:   CLR     $GDDAT
  815  005230  112710  000250                MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
  816  005234  111137  002126                MOVB    (R1),$BDDAT
  817  005240  023737  002124  002126        CMP     $GDDAT,$BDDAT
  818  005246  001401                        BEQ     7$
  819  005250  104003                        ERROR   3               ;IRR ERROR
  820  005252  005037  002124          7$:   CLR     $GDDAT
  821  005256  112710  000254                MOVB    #MACR,(R0)
  822  005262  111137  002126                MOVB    (R1),$BDDAT
  823  005266  023737  002124  002126        CMP     $GDDAT,$BDDAT
  824  005274  001401                        BEQ     10$
  825  005276  104004                        ERROR   4               ;ACR ERROR
  826  005300  005037  002124          10$:  CLR     $GDDAT
  827  005304  113710  002444                MOVB    ISRLOC,(R0)     ;CLEAR SINGLE ISR BIT
  828  005310  112710  000240                MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
  829  005314  111137  002126                MOVB    (R1),$BDDAT
  830  005320  023737  002124  002126        CMP     $GDDAT,$BDDAT
  831  005326  001401                        BEQ     11$
  832  005330  104006                        ERROR   6               ;ISR REG ERROR
  833  005332  004537  011242          11$:  JSR     R5,RESTRP       ;RESTORE VECTOR JUST TESTED
  834  005336  022737  002000  002452        CMP     #2000,VECLOC    ;FINISHED?
  835  005344  001402                        BEQ     12$             ;FINISHED VECTORS 204-1774?
  836  005346  000137  004674                JMP     112$            ;TEST NEXT VECTOR ADDR
  837  005352  005724                  12$:  TST     (R4)+           ;INDEX EXPECTED ISR AND IMR PATTERN
  838  005354  005237  002442                INC     IMRLOC          ;STORE CLEAR FOR NEXT MASK BIT
  839  005360  005237  002446                INC     IRRLOC          ;STORE SET FOR NEXT IRR BIT
  840  005364  005237  002444                INC     ISRLOC          ;STORE CLEAR FOR NEXT ISR BIT
  841  005370  005237  002456                INC     VECVAL          ;SETUP TO TEST NEXT VECTOR LEVEL
  842  005374  005337  002466                DEC     LVLCNT          ;FINISHED LEVELS 0-7
  843  005400  001402                        BEQ     13$             ;YES,CHECK BYTE COUNT END
  844  005402  000137  004666                JMP     111$            ;NO,DO NEXT LEVEL
  845  005406  005237  002472          13$:  INC     BYNUM           ;INDEX BYTE COUNT
  846  005412  104407                        CKSWR                   ;LOOK FOR SWITCH CHANGE REQUEST
  847  005414  022737  000400  002456        CMP     #400,VECVAL     ;FINISHED  ALL BYTE COUNTS 0-3
  848  005422  001402                        BEQ     14$             ;FINISHED TWO GROUPS?
  849  005424  000137  004620                JMP     121$            ;DO NEXT BYTE COUNT FOR
  850                                                                ;LEVELS 0-7
  851  005430  005337  002462          14$:  DEC     GRPCNT          ;FINISHED TWO GROUPS?
  852  005434  001406                        BEQ     TST4            ;;BR IF DONE
  853  005436  013700  002422                MOV     DRCSC,R0        ;CSRC = CONTROL GROUP 2
  854  005442  013701  002426                MOV     DRCSD,R1        ;CSRD = DATA    GROUP 2
  855  005446  000137  004600                JMP     120$            ;DO GROUP 2 FOR BYTE COUNTS 0-3
  856                                                                ;AND LEVELS 0-7 FOR EACH BYTE COUNT.
  857
  858
  859                                ;;*****************************************************************
  860                                ;*TEST 4         TEST VECTOR ADDR MEM,LVLS 0-7,BY0-3,(0-200)GRPS 1,2
  861                                ;;*****************************************************************
  862  005452  000004                TST4:   SCOPE
  863  005454  013700  002412                MOV     DRCSA,R0
  864  005460  013701  002416                MOV     DRCSB,R1
  865  005464  013702  002412                MOV     DRCSA,R2
```

}

```
866  005470  012737  000002  002462          MOV     #2,GRPCNT       ;DO TWO GROUPS
867  005476  012737  000001  002472  120$:   MOV     #1,BYNUM        ;INIT FOR FIRST BYTE COUNT
868  005504  004537  010650                  JSR     R5,CLRCSR       ;CLEAR ALL CSRS
869  005510  012737  000340  002456          MOV     #PVMA_VECVAL    ;START VECTOR LEVEL 0,BYTE COUNT 0
870  005516  012704  012644          121$:   MOV     #BGCHP3,R4      ;ISR,IMR PATTERN
871  005522  012737  000010  002466          MOV     #8.,LVLCNT      ;START LEVEL COUNT 0-7
872  005530  012737  000050  002442          MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR BIT CODE
873  005536  012737  000130  002446          MOV     #SSIRR,IRRLOC   ;STORE SET SINGLE IRR BIT CODE
874  005544  012737  005556  002110          MOV     #112$,$LPERR    ;LOOP RETURN
875  005552  005037  002452          111$:   CLR     VECLOC          ;START AT VECTOR 0
876  005556  106427  000340          112$:   MTPS    #340            ;
877  005562  012706  002100                  MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
878  005566  042712  001000                  BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
879  005572  004537  010730                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
880  005576  112712  000241                  MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
881                                                                  ;ARM GROUP 1 CHIP TO PASS
882                                                                  ;ENABLE TO GROUP 2.
883  005602  013703  002452                  MOV     VECLOC,R3       ;GET VECTOR
884  005606  004537  011634                  JSR     R5,CAT200       ;STORE TRAP CATCHER FOR 0-200
885  005612  010337  002454                  MOV     R3,VECPAT
886  005616  006037  002454                  ROR     VECPAT          ;ROTATE VECTOR ADDR TWO RIGHT
887  005622  006037  002454                  ROR     VECPAT          ;TO WRITE VECTOR MEM ADDR.
888  005626  012713  011734                  MOV     #INTSR3,(R3)    ;STORE VECTOR ROUTINE
889  005632  012763  000340  000002          MOV     #340,2(R3)      ;STORE PSW
890  005640  005037  002436                  CLR     INTFLG          ;CLEAR INT. FLAG
891  005644  013737  002472  002470          MOV     BYNUM,BYCNT     ;BYTE COUNT OF 0 TO 3
892                                                                  ;FOR 1 TO 4 VECTORS.
893  005652  113710  002456                  MOVB    VECVAL,(R0)     ;PRESELECT VECTOR MEM ADDR
894  005656  113711  002454          113$:   MOVB    VECPAT,(R1)     ;WRITE VECTOR
895  005662  005337  002470                  DEC     BYCNT           ;DO 1 TO 4 VECTORS
896  005666  001373                          BNE     113$            ;WRITE VECTORS BASED ON BYTE COUNT
897  005670  010037  002122                  MOV     R0,$BDADR       ;SAVE BUS ADDRESS
898  005674  112710  000060                  MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
899  005700  113710  002442                  MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
900  005704  106427  000000                  MTPS    #PR0
901  005710  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
902  005714  005737  002436                  TST     INTFLG          ;CHECK FOR INTERRUPTS
903  005720  001405                          BEQ     1$
904  005722  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
905  005726  104011                          ERROR   11              ;ERROR,ILLEGAL INTERRUPT
906  005730  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
907  005734  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
908  005740  113710  002446                  MOVB    IRRLOC,(R0)     ;SET SINGLE IRR BIT
909  005744  106427  000340                  MTPS    #PR7
910  005750  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
911  005754  001405                          BEQ     2$
912  005756  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
913  005762  104011                          ERROR   11              ;ILLEGAL INTERRUPTS
914  005764  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
915  005770  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
916  005774  106427  000000                  MTPS    #PR0
917  006000  152762  000002  000001          BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
918  006006  013737  002472  002124          MOV     BYNUM,$GDDAT    ;EXPECT NUMBER OF INTERRUPTS
919                                                                  ;EQUAL TO BYTE COUNT.
920  006014  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE NUMBER OF INTERRUPTS
921                                                                  ;EQUAL TO BYTE COUNT.
```

```
 922  006022  023737  002124  002126          CMP     $GDDAT,$BDDAT
 923  006030  001405                          BEQ     3$
 924  006032  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
 925  006036  104012                          ERROR   12              ;INTERRUPT TEST ERROR
 926  006040  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER SUB
 927  006044  106427  000340          3$:     MTPS    #PR7
 928  006050  010137  002122          4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
 929  006054  005037  002126                  CLR     $BDDAT          ;SET UP FOR BYTE READS
 930  006060  005037  002124                  CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
 931  006064  111437  002124                  MOVB    (R4),$GDDAT     ;EXPECTED DATA
 932  006070  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
 933  006074  111137  002126                  MOVB    (R1),$BDDAT
 934  006100  023737  002124  002126          CMP     $GDDAT,$BDDAT
 935  006106  001405                          BEQ     5$
 936  006110  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-200
 937  006114  104006                          ERROR   6               ;ISR ERROR
 938  006116  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
 939  006122  116437  000001  002124  5$:     MOVB    1(R4),$GDDAT    ;STORE EXPECTED
 940  006130  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
 941  006134  111137  002126                  MOVB    (R1),$BDDAT
 942  006140  023737  002124  002126          CMP     $GDDAT,$BDDAT
 943  006146  001405                          BEQ     6$
 944  006150  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
 945  006154  104005                          ERROR   5               ;IMR ERROR
 946  006156  004537  011634                  JSR     R5,CAT200       ;RESTORE CATCHER
 947  006162  005037  002124          6$:     CLR     $GDDAT
 948  006166  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
 949  006172  111137  002126                  MOVB    (R1),$BDDAT
 950  006176  023737  002124  002126          CMP     $GDDAT,$BDDAT
 951  006204  001405                          BEQ     7$
 952  006206  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
 953  006212  104003                          ERROR   3               ;IRR ERROR
 954  006214  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
 955  006220  005037  002124          7$:     CLR     $GDDAT
 956  006224  112710  000254                  MOVB    #MACR,(R0)
 957  006230  111137  002126                  MOVB    (R1),$BDDAT
 958  006234  023737  002124  002126          CMP     $GDDAT,$BDDAT
 959  006242  001405                          BEQ     10$
 960  006244  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
 961  006250  104004                          ERROR   4               ;ACR ERROR
 962  006252  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
 963  006256  005037  002124          10$:    CLR     $GDDAT
 964  006262  112710  000160                  MOVB    #CISR,(R0)      ;CLEAR ISR
 965  006266  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
 966  006272  111137  002126                  MOVB    (R1),$BDDAT
 967  006276  023737  002124  002126          CMP     $GDDAT,$BDDAT
 968  006304  001405                          BEQ     11$
 969  006306  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-202
 970  006312  104006                          ERROR   6               ;ISR REG ERROR
 971  006314  004537  011634                  JSR     R5,CAT200       ;RESTORE TRAP CATCHER
 972  006320  004537  011572          11$:    JSR     R5,RES200       ;RESTORE VECTOR JUST TESTED
 973  006324  004537  011204                  JSR     R5,FRMBUF       ;RESTORE 0-200 TO SYSMAC CONTROL
 974  006330  022737  000204  002452          CMP     #204,VECLOC     ;FINISHED?
 975  006336  001402                          BEQ     12$
 976  006340  000137  005556                  JMP     112$            ;TEST NEXT VECTOR ADDR
 977  006344  005724                  12$:    TST     (R4)+           ;INDEX EXPECTED ISR AND IMR PATTERN
```

```
 978  006346  005237  002442                 INC     IMRLOC          ;STORE CLEAR NEXT IMR BIT
 979  006352  005237  002446                 INC     IRRLOC          ;STORE SET NEXT IRR BIT
 980  006356  005237  002456                 INC     VECVAL          ;SETUP TO TEST NEXT VECTOR LEVEL
 981  006362  005337  002466                 DEC     LVLCNT          ;FINISHED LEVELS 0-7?
 982  006366  001402                         BEQ     13$             ;YES,CHECK BYTE COUNT END
 983  006370  000137  005552                 JMP     111$            ;NO,DO NEXT LEVEL,SAME BYTE COUNT
 984  006374  005237  002472          13$:   INC     BYNUM           ;INDEX BYTE COUNT
 985  006400  104407                         CKSWR                   ;LOOK FOR SWITCH CHANGE
 986  006402  022737  000400  002456         CMP     #400,VECVAL     ;FINISHED ALL FOUR BYTE COUNTS
 987  006410  001402                         BEQ     14$             ;YES,FINISHED GROUP
 988  006412  000137  005516                 JMP     121$            ;NO,DO NEXT BYTE COUNT WITH
 989                                                                 ;LEVELS 0-7.
 990  006416  005337  002462          14$:   DEC     GRPCNT          ;FINISHED BOTH GROUP 1 AND GROUP 2?
 991  006422  001406                         BEQ     15$             ;YES,BOTH GROUPS TESTED
 992  006424  013700  002422                 MOV     DRCSC,R0        ;NO,TEST GROUP 2
 993                                                                 ;CSRC = CONTROL GROUP 2
 994  006430  013701  002426                 MOV     DRCSD,R1        ;CSRD = DATA    GROUP 2
 995  006434  000137  005476                 JMP     120$            ;RETURN AND TEST GROUP2
 996                                                                 ;FOR BYTE COUNTS 0-3 AND
 997                                                                 ;LEVELS 0-7 FOR EACH BYTE COUNT.
 998  006440  004537  011204          15$:   JSR     R5,FRMBUF       ;RESTORE 0-202,EXIT TEST
 999
1000
1001                                 ;;*****************************************************************
1002                                 ;*TEST 5        TEST VECTOR ADDR UNIQUENESS IN FIXED MODE FOR GRPS 1,2
1003                                 ;;*****************************************************************
1004  006444  000004                 TST5:   SCOPE
1005                                                                 ;BYTE COUNT = 4
1006                                                                 ;LEVELS = 0-7
1007  006446  004537  010650                 JSR     R5,CLRCSR       ;CLEAR ALL CSRS
1008  006452  004537  011144                 JSR     R5,CLRBF        ;CLEAR VECTOR BUFFER AREA
1009  006456  004537  011334                 JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
1010  006462  004537  011110                 JSR     R5,STRVEC       ;STORE VECTOR AREA 300-674
1011                                                                 ;WITH INT. SERV ROUTINES
1012  006466  012737  000377  002456         MOV     #377,VECVAL     ;VECTOR MEM  FINAL VALUE
1013  006474  004537  010730                 JSR     R5,CLRIRR       ;CLEAR IRR REGS
1014  006500  012704  017116                 MOV     #VBUF,R4        ;VECTOR ADDRESS BUFFER
1015  006504  004537  010766                 JSR     R5,VECFIL       ;FILL VECTOR MEMORY FOR GROUPS 1,2
1016                                                                 ;WITH VECTORS 300-674
1017  006510  013700  002412                 MOV     DRCSA,R0        ;GROUP 1 CSR
1018  006514  013701  002422                 MOV     DRCSC,R1        ;GROUP 2 CSR
1019  006520  013702  002416                 MOV     DRCSB,R2
1020  006524  013703  002426                 MOV     DRCSD,R3
1021  006530  010037  002122                 MOV     R0,$BDADR       ;STORE BUS ADDRESS
1022  006534  112710  000300                 MOVB    #PACR,(R0)      ;PRESELECT GP1 ACR FOR WRITING
1023  006540  112712  000377                 MOVB    #377,(R2)       ;SET ALL ACR BITS GP1
1024  006544  112710  000040                 MOVB    #CIMR,(R0)      ;CLEAR ALL IMR BITS GP1
1025  006550  005037  002436                 CLR     INTFLG          ;CLEAR INT. FLAG
1026  006554  106427  000000                 MTPS    #PR0
1027  006560  112710  000120                 MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
1028  006564  112710  000241                 MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
1029  006570  106427  000340                 MTPS    #340
1030  006574  005737  002436                 TST     INTFLG          ;CHECK FOR NO INTERRUPTS
1031  006600  001401                         BEQ     1$
1032  006602  104011                         ERROR   11              ;ILLEGAL INTERRUPT
1033  006604  005037  002436          1$:    CLR     INTFLG
```

```
1034  006610  112711  000300                  MOVB    #PACR,(R1)      ;PRESELECT GROUP2 FOR WRITING
1035  006614  112713  000377                  MOVB    #377,(R3)       ;ALL ACR BITS SET GP2
1036  006620  112711  000040                  MOVB    #CIMR,(R1)      ;CLEAR ALL IMR BITS GP2
1037  006624  106427  000000                  MTPS    #PR0
1038  006630  112711  000120                  MOVB    #SIRR,(R1)      ;SET ALL IRR BITS
1039  006634  112711  000241                  MOVB    #241,(R1)       ;LMD57 - ARM THE CHIP
1040  006640  106427  000340                  MTPS    #PR7
1041  006644  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
1042  006650  001401                          BEQ     2$
1043  006652  104011                          ERROR   11              ;ILLEGAL INTERRUPT
1044  006654  005037  002436          2$:     CLR     INTFLG
1045  006660  106427  000000                  MTPS    #PR0
1046  006664  152760  000002  000001          BISB    #BIT1,1(R0)     ;SET INT. ENABLE,EXPECT INTERRUPTS
1047  006672  012737  000100  002124          MOV     #64.,$GDDAT     ;EXPECT 64 INTERRUPTS
1048  006700  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE 64 INTERRUPTS
1049  006706  106427  000340                  MTPS    #PR7
1050  006712  023737  002124  002126          CMP     $GDDAT,$BDDAT
1051  006720  001401                          BEQ     3$
1052  006722  104012                          ERROR   12              ;INTERRUPT TEST ERROR
1053  006724  012737  101710  002124  3$:     MOV     #101710,$GDDAT  ;RDY,DIR,I/E,CHIP - 31X
1054  006732  010037  002122                  MOV     R0,$BDADR       ;CSRA
1055  006736  011037  002126                  MOV     (R0),$BDDAT
1056  006742  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
1057  006750  023737  002124  002126          CMP     $GDDAT,$BDDAT
1058  006756  001401                          BEQ     4$
1059  006760  104002                          ERROR   2               ;CSRA REG ERROR
1060  006762  012737  000310  002124  4$:     MOV     #310,$GDDAT     ;STORE EXPECTED
1061  006770  010137  002122                  MOV     R1,$BDADR       ;CSRC
1062  006774  011137  002126                  MOV     (R1),$BDDAT     ;
1063  007000  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
1064  007006  023737  002124  002126          CMP     $GDDAT,$BDDAT
1065  007014  001401                          BEQ     5$
1066  007016  104002                          ERROR   2               ;CSRC ERROR
1067  007020  010237  002122          5$:     MOV     R2,$BDADR       ;CSRB ADDRESS
1068  007024  005037  002126                  CLR     $BDDAT
1069  007030  005037  002124                  CLR     $GDDAT
1070  007034  112710  000240                  MOVB    #MISR,(R0)      ;READ ISR
1071  007040  111237  002126                  MOVB    (R2),$BDDAT
1072  007044  023737  002124  002126          CMP     $GDDAT,$BDDAT
1073  007052  001401                          BEQ     6$
1074  007054  104006                          ERROR   6               ;ISR ERROR,GP1
1075  007056  112710  000244          6$:     MOVB    #MIMR,(R0)      ;MODE BITS FOR IMR
1076  007062  111237  002126                  MOVB    (R2),$BDDAT
1077  007066  023737  002124  002126          CMP     $GDDAT,$BDDAT
1078  007074  001401                          BEQ     7$
1079  007076  104005                          ERROR   5               ;IMR ERROR,GP1
1080  007100  112710  000250          7$:     MOVB    #MIRR,(R0)      ;MODE BITS FOR IRR
1081  007104  111237  002126                  MOVB    (R2),$BDDAT
1082  007110  023737  002124  002126          CMP     $GDDAT,$BDDAT
1083  007116  001401                          BEQ     10$
1084  007120  104003                          ERROR   3               ;IRR ERROR
1085  007122  012737  000377  002124  10$:     MOV     #377,$GDDAT     ;EXPECTED ACR
1086  007130  112710  000254                  MOVB    #MACR,(R0)      ;READ ACR
1087  007134  111237  002126                  MOVB    (R2),$BDDAT
1088  007140  023737  002124  002126          CMP     $GDDAT,$BDDAT
1089  007146  001401                          BEQ     11$
```

```
1090  007150  104004                    ERROR   4               ;ACR ERROR
1091  007152  005037  002124    11$:     CLR     $GDDAT
1092  007156  005037  002126             CLR     $BDDAT
1093  007162  010337  002122             MOV     R3,$BDADR       ;CSRD ADDRESS
1094  007166  112711  000240             MOVB    #MISR,(R1)      ;READ ISR ,GP2
1095  007172  111337  002126             MOVB    (R3),$BDDAT
1096  007176  023737  002124  002126     CMP     $GDDAT,$BDDAT
1097  007204  001401                    BEQ     12$
1098  007206  104006                    ERROR   6               ;ISR ERROR,GP1
1099  007210  112711  000244    12$:     MOVB    #MIMR,(R1)      ;READ IMR GP2
1100  007214  111337  002126             MOVB    (R3),$BDDAT
1101  007220  023737  002124  002126     CMP     $GDDAT,$BDDAT
1102  007226  001401                    BEQ     13$
1103  007230  104005                    ERROR   5               ;IMR ERROR,GP2
1104
1105  007232  112711  000250    13$:     MOVB    #MIRR,(R1)      ;READ IRR
1106  007236  111337  002126             MOVB    (R3),$BDDAT
1107  007242  023737  002124  002126     CMP     $GDDAT,$BDDAT
1108  007250  001401                    BEQ     14$
1109  007252  104003                    ERROR   3               ;IRR ERROR,GP2
1110  007254  112711  000254    14$:     MOVB    #MACR,(R1)      ;READ ACR
1111  007260  012737  000377  002124     MOV     #377,$GDDAT
1112  007266  111337  002126             MOVB    (R3),$BDDAT
1113  007272  023737  002124  002126     CMP     $GDDAT,$BDDAT
1114  007300  001401                    BEQ     15$
1115  007302  104004                    ERROR   4               ;ACR ERROR,GP2
1116  007304  012702  000100    15$:     MOV     #64.,R2
1117  007310  012704  017116             MOV     #VBUF,R4        ;VECTOR BUFFER
1118  007314  012737  000300  002124     MOV     #300,$GDDAT     ;START WITH FIRST VECTOR 300
1119  007322  010437  002122    16$:     MOV     R4,$BDADR       ;VECTOR BUFFER ADDRESS
1120  007326  011437  002126             MOV     (R4),$BDDAT
1121  007332  023737  002124  002126     CMP     $GDDAT,$BDDAT
1122  007340  001401                    BEQ     17$             ;VECTORED PROPERLY
1123  007342  104007                    ERROR   7               ;VECTOR ADDR MEM ERROR
1124  007344  062737  000004  002124  17$:     ADD     #4,$GDDAT
1125  007352  062704  000002             ADD     #2,R4           ;NEXT BUFFER ADDRESS
1126  007356  005302                    DEC     R2              ;COMPLETED 64 BUFFER CHECKS?
1127  007360  001360                    BNE     16$             ;CHECK NEXT VECTOR BUFFER LOCATION
1128                                                            ;THAT WAS STORED BY THE INTERRUPT
1129                                                            ;SERVICE ROUTINES.
1130
1131                                    ;;*******************************************************************
1132                                    ;*TEST 6        TEST VECTOR ADDR UNIQUENESS IN ROTATING MODE FOR GRPS 1,2
1133                                    ;;*******************************************************************
1134  007362  000004            TST6:   SCOPE
1135                                                            ;BYTE COUNT = 4
1136                                                            ;LEVELS = 0-7
1137  007364  004537  010650             JSR     R5,CLRCSR       ;CLEAR ALL CSRS
1138  007370  004537  011144             JSR     R5,CLRBF        ;CLEAR VECTOR BUFFER AREA
1139  007374  004537  011334             JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
1140  007400  004537  011110             JSR     R5,STRVEC       ;STORE VECTOR AREA 300-674
1141                                                            ;WITH INT. SERV ROUTINES
1142  007404  012737  000377  002456     MOV     #377,VECVAL     ;VECTOR MEM  FINAL VALUE
1143  007412  004537  010730             JSR     R5,CLRIRR       ;CLEAR IRR REGS
1144  007416  012704  017116             MOV     #VBUF,R4        ;VECTOR ADDRESS BUFFER
1145  007422  004537  010766             JSR     R5,VECFIL       ;FILL VECTOR MEMORY FOR GROUPS 1,2
```

```
1146
1147  007426  013700  002412                  MOV     DRCSA,R0        ;WITH VECTORS 300-674
1148  007432  013701  002422                  MOV     DRCSC,R1        ;GROUP 1 CSR
1149  007436  013702  002416                  MOV     DRCSB,R2        ;GROUP 2 CSR
1150  007442  013703  002426                  MOV     DRCSD,R3
1151  007446  010037  002122                  MOV     R0,$BDADR       ;STORE BUS ADDRESS
1152  007452  012737  000010  002474          MOV     #8.,LVLSAV      ;COUNTER FOR 7 LEVELS
1153  007460  112710  000201                  MOVB    #201,(R0)       ;ROTATING PRIORITY GROUP 1
1154  007464  112710  000040                  MOVB    #CIMR,(R0)      ;CLEAR ALL IMR BITS GP1
1155  007470  005037  002436                  CLR     INTFLG          ;CLEAR INT. FLAG
1156  007474  106427  000000                  MTPS    #PR0
1157  007500  112710  000120                  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
1158  007504  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
1159  007510  106427  000340                  MTPS    #340
1160  007514  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
1161  007520  001401                          BEQ     1$
1162  007522  104011                          ERROR   11              ;ILLEGAL INTERRUPT
1163  007524  005037  002436          1$:     CLR     INTFLG
1164  007530  112711  000201                  MOVB    #201,(R1)       ;ROTATING PRIORITY FOR GROUP 2
1165  007534  112711  000040                  MOVB    #CIMR,(R1)      ;CLEAR ALL IMR BITS GP2
1166  007540  106427  000000                  MTPS    #PR0
1167  007544  112711  000120                  MOVB    #SIRR,(R1)      ;SET ALL IRR BITS
1168  007550  112711  000241                  MOVB    #241,(R1)       ;LMD57 - ARM THE CHIP
1169  007554  106427  000340                  MTPS    #PR7
1170  007560  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
1171  007564  001401                          BEQ     2$
1172  007566  104011                          ERROR   11              ;ILLEGAL INTERRUPT
1173  007570  005037  002436          2$:     CLR     INTFLG
1174  007574  106427  000000                  MTPS    #PR0
1175  007600  152760  000002  000001          BISB    #BIT1,1(R0)     ;SET INT. ENABLE,EXPECT INTERRUPTS
1176  007606  012737  000004  002124          MOV     #4,$GDDAT       ;EXPECT 4 INTERRUPTS
1177  007614  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE 4 INTERRUPTS
1178  007622  106427  000340                  MTPS    #PR7
1179  007626  023737  002124  002126          CMP     $GDDAT,$BDDAT
1180  007634  001401                          BEQ     100$
1181  007636  104012                          ERROR   12              ;INTERRUPT TEST ERROR
1182  007640  112710  000140          100$:   MOVB    #CHPISR,(R0)    ;CLEAR HIGHEST PRIORITY ISR
1183  007644  112710  000120                  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
1184  007650  005337  002474                  DEC     LVLSAV          ;DONE 8 LEVELS,GROUP 1
1185  007654  001345                          BNE     2$              ;DO NEXT LEVEL IN ROTATION
1186  007656  112710  000100                  MOVB    #CIRR,(R0)      ;CLEAR IRR BITS GROUP 1
1187  007662  012737  000010  002474          MOV     #8.,LVLSAV      ;DO 8 LEVELS,GROUP 2
1188  007670  005037  002436          101$:   CLR     INTFLG          ;CLEAR INT FLAG
1189  007674  106427  000000                  MTPS    #PR0
1190  007700  012737  000004  002124          MOV     #4,$GDDAT
1191  007706  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE FOUR INTERRUPTS
1192  007714  106427  000340                  MTPS    #PR7
1193  007720  023737  002124  002126          CMP     $GDDAT,$BDDAT
1194  007726  001401                          BEQ     102$
1195  007730  104012                          ERROR   12              ;INTERRUPT TEST ERROR
1196  007732  112711  000140          102$:   MOVB    #CHPISR,(R1)    ;CLEAR HIGHEST PRIORITY ISR
1197  007736  112711  000120                  MOVB    #SIRR,(R1)      ;SET ALL IRR BITS
1198  007742  005337  002474                  DEC     LVLSAV          ;DONE 8 LEVELS?
1199  007746  001350                          BNE     101$            ;DO NEXT LEVEL IN ROTATION,GP2
1200  007750  112711  000100                  MOVB    #CIRR,(R1)      ;CLEAR IRR BITS,GROUP 2
1201  007754  012737  101750  002124  3$:     MOV     #101750,$GDDAT  ;RDY,DIR,I/E,CHIP - 35X
```

```
1202  007762  010037  002122                  MOV     R0,$BDADR       ;CSRA
1203  007766  011037  002126                  MOV     (R0),$BDDAT
1204  007772  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
1205  010000  023737  002124  002126          CMP     $GDDAT,$BDDAT
1206  010006  001401                          BEQ     4$
1207  010010  104002                          ERROR   2               ;CSRA REG ERROR
1208  010012  012737  000350  002124  4$:     MOV     #350,$GDDAT     ;STORE EXPECTED
1209  010020  010137  002122                  MOV     R1,$BDADR       ;CSRC
1210  010024  011137  002126                  MOV     (R1),$BDDAT     ;
1211  010030  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
1212  010036  023737  002124  002126          CMP     $GDDAT,$BDDAT
1213  010044  001401                          BEQ     5$
1214  010046  104002                          ERROR   2               ;CSRC ERROR
1215  010050  010237  002122          5$:     MOV     R2,$BDADR       ;CSRB ADDRESS
1216  010054  005037  002126                  CLR     $BDDAT
1217  010060  005037  002124                  CLR     $GDDAT
1218  010064  112710  000240                  MOVB    #MISR,(R0)      ;READ ISR
1219  010070  111237  002126                  MOVB    (R2),$BDDAT
1220  010074  023737  002124  002126          CMP     $GDDAT,$BDDAT
1221  010102  001401                          BEQ     6$
1222  010104  104006                          ERROR   6               ;ISR ERROR,GP1
1223  010106  112710  000244          6$:     MOVB    #MIMR,(R0)      ;MODE BITS FOR IMR
1224  010112  111237  002126                  MOVB    (R2),$BDDAT
1225  010116  023737  002124  002126          CMP     $GDDAT,$BDDAT
1226  010124  001401                          BEQ     7$
1227  010126  104005                          ERROR   5               ;IMR ERROR,GP1
1228  010130  112710  000250          7$:     MOVB    #MIRR,(R0)      ;MODE BITS FOR IRR
1229  010134  111237  002126                  MOVB    (R2),$BDDAT
1230  010140  023737  002124  002126          CMP     $GDDAT,$BDDAT
1231  010146  001401                          BEQ     10$
1232  010150  104003                          ERROR   3               ;IRR ERROR
1233  010152  112710  000254          10$:    MOVB    #MACR,(R0)      ;READ ACR
1234  010156  111237  002126                  MOVB    (R2),$BDDAT
1235  010162  023737  002124  002126          CMP     $GDDAT,$BDDAT
1236  010170  001401                          BEQ     11$
1237  010172  104004                          ERROR   4               ;ACR ERROR
1238  010174  010337  002122          11$:    MOV     R3,$BDADR       ;CSRD ADDRESS
1239  010200  112711  000240                  MOVB    #MISR,(R1)      ;READ ISR ,GP2
1240  010204  111337  002126                  MOVB    (R3),$BDDAT
1241  010210  023737  002124  002126          CMP     $GDDAT,$BDDAT
1242  010216  001401                          BEQ     12$
1243  010220  104006                          ERROR   6               ;ISR ERROR,GP1
1244  010222  112711  000244          12$:    MOVB    #MIMR,(R1)      ;READ IMR GP2
1245  010226  111337  002126                  MOVB    (R3),$BDDAT
1246  010232  023737  002124  002126          CMP     $GDDAT,$BDDAT
1247  010240  001401                          BEQ     13$
1248  010242  104005                          ERROR   5               ;IMR ERROR,GP2
1249
1250  010244  112711  000250          13$:    MOVB    #MIRR,(R1)      ;READ IRR
1251  010250  111337  002126                  MOVB    (R3),$BDDAT
1252  010254  023737  002124  002126          CMP     $GDDAT,$BDDAT
1253  010262  001401                          BEQ     14$
1254  010264  104003                          ERROR   3               ;IRR ERROR,GP2
1255  010266  112711  000254          14$:    MOVB    #MACR,(R1)      ;READ ACR
1256  010272  111337  002126                  MOVB    (R3),$BDDAT
1257  010276  023737  002124  002126          CMP     $GDDAT,$BDDAT
```

```
1258  010304  001401                          BEQ     200$
1259  010306  104004                          ERROR   4                       ;ACR ERROR,GP2
1260  010310  105010                  200$:   CLRB    (R0)                    ;INIT GROUP 1 MODE BITS
1261  010312  105011                          CLRB    (R1)                    ;INIT GROUP 2 MODE BITS
1262  010314  012737  101700  002124          MOV     #101700,$GDDAT          ;EXPECTED CSRA
1263  010322  010037  002122                  MOV     R0,$BDADR               ;CSRA
1264  010326  011037  002126                  MOV     (R0),$BDDAT
1265  010332  042737  000007  002126          BIC     #7,$BDDAT               ;CLEAR UNDEFINED BITS
1266  010340  023737  002124  002126          CMP     $GDDAT,$BDDAT
1267  010346  001401                          BEQ     201$
1268  010350  104002                          ERROR   2                       ;CSRA REG ERROR
1269  010352  012737  000200  002124  201$:   MOV     #200,$GDDAT             ;STORE EXPECTED
1270  010360  010137  002122                  MOV     R1,$BDADR               ;CSRC
1271  010364  011137  002126                  MOV     (R1),$BDDAT             ;
1272  010370  042737  000007  002126          BIC     #7,$BDDAT               ;CLEAR UNDEFINED BITS
1273  010376  023737  002124  002126          CMP     $GDDAT,$BDDAT
1274  010404  001401                          BEQ     15$
1275  010406  104002                          ERROR   2                       ;CSRC ERROR
1276  010410  012702  000100          15$:    MOV     #64.,R2
1277  010414  012704  017116                  MOV     #VBUF,R4                ;VECTOR BUFFER
1278  010420  012737  000300  002124          MOV     #300,$GDDAT             ;START WITH FIRST VECTOR 300
1279  010426  010437  002122          16$:    MOV     R4,$BDADR               ;VECTOR BUFFER ADDRESS
1280  010432  011437  002126                  MOV     (R4),$BDDAT
1281  010436  023737  002124  002126          CMP     $GDDAT,$BDDAT
1282  010444  001401                          BEQ     17$                     ;VECTORED PROPERLY
1283  010446  104007                          ERROR   7                       ;VECTOR ADDR MEM ERROR
1284  010450  062737  000004  002124  17$:    ADD     #4,$GDDAT
1285  010456  062704  000002                  ADD     #2,R4                   ;NEXT BUFFER ADDRESS
1286  010462  005302                          DEC     R2                      ;COMPLETED 64 BUFFER CHECKS?
1287  010464  001360                          BNE     16$                     ;CHECK NEXT VECTOR BUFFER LOCATION
1288                                                                          ;THAT WAS STORED BY THE INTERRUPT
1289                                                                          ;SERVICE ROUTINES.
1290
1291
1292                                  ;DON'T REPORT 'END OF PASS' UNTIL ALL SELECTED DRV11J'S HAVE BEEN TESTED.
1293  010466  013701  002412          NXDEV:  MOV     DRCSA,R1                ;INIT TO SETUP DRV11J ADDRESS
1294  010472  000241                  NXDEV1: CLC                             ;CLEAR CARRY FOR DEVICE MAP
1295  010474  006037  002434                  ROR     DMAP                    ;LOOK FOR NEXT DRV11J
1296  010500  001412                          BEQ     $EOP                    ;BR IF ALL TESTED
1297  010502  162701  000020                  SUB     #20,R1                  ;NEXT DRV11J STARTS -20 FROM PAST CSR
1298  010506  005237  002202                  INC     $UNIT                   ;UPDAT UNIT #
1299  010512  032737  000001  002434          BIT     #1,DMAP                 ;IS UNIT SELECTED?
1300  010520  001764                          BEQ     NXDEV1                  ;BR IF NOT
1301  010522  000137  003132                  JMP     NEXPAS                  ;TEST NEXT DRV11J
1302
1303                                  .SBTTL  END OF PASS ROUTINE
1304
1305                                  ;*********************************************************************
1306                                  ;*INCREMENT THE PASS NUMBER ($PASS)
1307                                  ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
1308                                  ;*IF THERES A MONITOR GO TO IT
1309                                  ;*IF THERE ISN'T JUMP TO START
1310
1311  010526                          $EOP:
1312  010526  000240                          NOP
1313  010530  005037  002102                  CLR     $TSTNM                  ;ZERO THE TEST NUMBER
```

```
1314  010534  005037  002160                CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
1315  010540  005237  002176                INC     $PASS           ;;INCREMENT THE PASS NUMBER
1316  010544  042737  100000  002176        BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
1317  010552  005327                        DEC     (PC)+           ;;LOOP?
1318  010554  000001              $EOPCT:   .WORD   1
1319  010556  003022                        BGT     $DOAGN          ;;YES
1320  010560  012737                        MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
1321  010562  000001              $ENDCT:   .WORD   1
1322  010564  010554                        $EOPCT
1323  010566  104401  010633                TYPE    ,$ENDMG         ;;TYPE 'END PASS #'
1324  010572  013746  002176                MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
1325  010576  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
1326  010600  104401  010630                TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
1327  010604  013700  000042      $GET42:   MOV     @#42,R0         ;;GET MONITOR ADDRESS
1328  010610  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
1329  010612  000005                        RESET                   ;;CLEAR THE WORLD
1330  010614  004710              $ENDAD:   JSR     PC,(R0)         ;;GO TO MONITOR
1331  010616  000240                        NOP                     ;;SAVE ROOM
1332  010620  000240                        NOP                     ;;FOR
1333  010622  000240                        NOP                     ;;ACT11
1334  010624                      $DOAGN:
1335  010624  000137                        JMP     @(PC)+          ;;RETURN
1336  010626  003066              $RTNAD:   .WORD   START1
1337  010630     377     377      $ENULL:   .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
1338  010633     015  042412  042116  $ENDMG:   .ASCIZ  <15><12>/END PASS #/
1339  010640  050040  051501  020123
1340  010646  000043
```

```
1341                                    .SBTTL  PROGRAM SUBROUTINES
1342
1343
1344                                    ;;*****************************************************************
1345                                    ;CLEAR ALL CONTROL/STATUS REGISTERS
1346                                    ;;*****************************************************************
1347  010650  010046                    CLRCSR: MOV     R0,-(SP)        ;SAVE R0
1348  010652  106427  000340                    MTPS    #PR7            ;PSW = 340
1349  010656  012737  000340  002456           MOV     #PVMA,VECVAL    ;INIT VECTOR ADDR MEM = 340
1350  010664  012737  000300  002452           MOV     #300,VECLOC     ;INIT VECTOR ADDR TO 300
1351  010672  013700  002412                    MOV     DRCSA,R0        ;START OF CSR ADDRESS
1352  010676  005037  002124                    CLR     $GDDAT          ;CLEAR EXPECTED
1353  010702  005037  002126                    CLR     $BDDAT          ;CLEAR REC'D
1354  010706  005010                            CLR     (R0)            ;CLEAR CSRA;CHIP RESET GROUP 1
1355  010710  105060  000005                    CLRB    5(R0)           ;CLEAR HIGH BYTE CSRB
1356  010714  005060  000010                    CLR     10(R0)          ;CLEAR CSRC;CHIP RESET GROUP 2
1357  010720  105060  000015                    CLRB    15(R0)          ;CLEAR HIGH BYTE CSRD
1358  010724  012600                            MOV     (SP)+,R0        ;RETURN R0
1359  010726  000205                            RTS     R5
1360
1361
1362                                    ;CLEAR IRR REGISTERS,GROUP 1,GROUP 2 WITH CHIP RESET
1363  010730  010046                    CLRIRR: MOV     R0,-(SP)
1364  010732  013700  002412                    MOV     DRCSA,R0        ;START OF CSR ADDRESS
1365  010736  105060  000011                    CLRB    11(R0)          ;CSRC TO INPUT MODE
1366  010742  112760  000001  000001           MOVB    #BIT0,1(R0)     ;CSRA TO OUTPUT MODE
1367  010750  005060  000002                    CLR     2(R0)           ;CLEAR DBRA
1368  010754  105010                            CLRB    (R0)            ;CHIP RESET OF GROUP1
1369  010756  105060  000010                    CLRB    10(R0)          ;CHIP RESET OF GROUP 2
1370  010762  012600                            MOV     (SP)+,R0        ;RESTORE REGISTER
1371  010764  000205                            RTS     R5              ;EXIT
1372
1373                                    ;ROUTINE TO FILL VECTOR MEMORY FOR VECTOR UNIQUENESS TEST
1374                                    ;300-474        GROUP 1
1375                                    ;500-674        GROUP 2
1376  010766  013700  002412            VECFIL: MOV     DRCSA,R0        ;START GROUP 1
1377  010772  013701  002416                    MOV     DRCSB,R1
1378  010776  012702  000002                    MOV     #2,R2           ;TWO GROUPS
1379  011002  012737  000060  002454           MOV     #60,VECPAT      ;VECTOR START 300
1380  011010  012737  000370  002456  1$:      MOV     #370,VECVAL     ;START VECTOR LEVEL 0,BY4
1381  011016  012737  000010  002466           MOV     #8.,LVLCNT      ;LEVELS 0-7
1382  011024  113710  002456          2$:     MOVB    VECVAL,(R0)
1383  011030  012737  000004  002470           MOV     #4,BYCNT        ;DO FOUR BYTE COUNTS
1384  011036  113711  002454          3$:     MOVB    VECPAT,(R1)     ;WRITE VECTOR
1385  011042  005237  002454                    INC     VECPAT          ;SETUP FOR NEXT VECTOR
1386  011046  005337  002470                    DEC     BYCNT           ;DONE FOUR BYTE COUNTS?
1387  011052  001371                            BNE     3$              ;DO NEXT VECTOR
1388  011054  005337  002466          4$:     DEC     LVLCNT          ;DONE ALL LEVELS IN GROUP?
1389  011060  001403                            BEQ     5$
1390  011062  005237  002456                    INC     VECVAL          ;INC NEXT LEVEL
1391  011066  000756                            BR      2$              ;DO NEXT LEVEL
1392  011070  005302                  5$:       DEC     R2              ;DONE BOTH GROUPS?
1393  011072  001405                            BEQ     6$              ;DONE,EXIT
1394  011074  013700  002422                    MOV     DRCSC,R0        ;DO GROUP 2
1395  011100  013701  002426                    MOV     DRCSD,R1
1396  011104  000741                            BR      1$
```

```
1397  011106  000205                  6$:     RTS     R5              ;EXIT
1398
1399                                  ;ROUTINE TO STORE VECTOR AREA 300-674 WITH INTERRUPT SERVICE
1400  011110  012700  000300          STRVEC: MOV     #300,R0         ;START AT VECTOR 300
1401  011114  012701  012044                  MOV     #INT0,R1        ;FIRST SERVICE ROUTINE
1402  011120  012702  000100                  MOV     #64.,R2         ;STORE 64 SERVICE ROUTINES
1403  011124  010120                  1$:     MOV     R1,(R0)+
1404  011126  012720  000340                  MOV     #340,(R0)+
1405  011132  062701  000006                  ADD     #6,R1           ;SETUP FOR NEXT VECTOR STORAGE
1406  011136  005302                          DEC     R2              ;DONE WITH 64 VECTOR ROUTINES?
1407  011140  001371                          BNE     1$
1408  011142  000205                          RTS     R5              ;EXIT
1409
1410                                  ;ROUTINE TO CLEAR VECTOR BUFFER
1411  011144  012700  017116          CLRBF:  MOV     #VBUF,R0        ;VECTOR BUFFER START
1412  011150  012701  000100                  MOV     #64.,R1
1413  011154  005020                  1$:     CLR     (R0)+
1414  011156  005301                          DEC     R1
1415  011160  001375                          BNE     1$
1416  011162  000205                          RTS     R5
1417
1418                                  ;ROUTINE TO STORE 0-202 INTO BUFFER AREA TO ALLOW
1419                                  ;VECTOR TESTS TO 0-200.
1420  011164  005000                  TOBUF:  CLR     R0
1421  011166  012701  016712                  MOV     #DBUF,R1        ;BUFFER START
1422  011172  012021                  1$:     MOV     (R0)+,(R1)+     ;STORE 0-202 INTO BUFFER
1423  011174  022700  000204                  CMP     #204,R0         ;FINISHED?
1424  011200  001374                          BNE     1$              ;NO
1425  011202  000205                          RTS     R5              ;RETURN
1426
1427                                  ;STORE BUFFER AREA INTO LOCATIONS 0-202
1428                                  ;AFTER VECTOR TESTS
1429  011204  010046                  FRMBUF: MOV     R0,-(SP)        ;SAVE R0
1430  011206  010146                          MOV     R1,-(SP)
1431  011210  005000                          CLR     R0
1432  011212  012701  016712                  MOV     #DBUF,R1        ;BUFFER START
1433  011216  012120                  1$:     MOV     (R1)+,(R0)+     ;STORE BUFFER INTO LOCS 0-202
1434  011220  022700  000204                  CMP     #204,R0         ;FINISHED?
1435  011224  001374                          BNE     1$              ;NO
1436  011226  013777  002460  170704          MOV     SWRSAV,@SWR     ;STORE LOCATION 176,SOFT SWR
1437  011234  012601                          MOV     (SP)+,R1        ;RESTORE R1
1438  011236  012600                          MOV     (SP)+,R0        ;RESTORE R0
1439  011240  000205                          RTS     R5              ;RETURN
1440
1441                                  ;RESTORE VECTOR JUST TESTED TO ORIGINAL TRAP FOR 204-1774
1442  011242  022737  000001  002210  RESTRP: CMP     #1,$ENV         ;CHECK FOR APT
1443  011250  001421                          BEQ     1$              ;YES,BRANCH
1444  011252  032777  010000  170660          BIT     #SW12,@SWR      ;CHECK TYPE OF TRAP RETURN
1445  011260  001412                          BEQ     10$             ;RESTORE ILLEGAL VECTOR ROUTINE
1446  011262  062737  000002  002452          ADD     #2,VECLOC       ;RESTORE TRAP
1447  011270  013723  002452                  MOV     VECLOC,(R3)+    ;TO VECTOR
1448  011274  005013                          CLR     (R3)            ;RESTORE HALT
1449  011276  062737  000002  002452          ADD     #2,VECLOC       ;SETUP FOR NEXT
1450  011304  000412                          BR      2$              ;RETURN
1451  011306  012723  011562          10$:    MOV     #TRPALL,(R3)+   ;RESTORE ILLEGAL TRAP ROUTINE
1452  011312  000402                          BR      11$             ;RESTORE PSW
```

```
1453  011314  012723  011544          1$:     MOV     #TRPOUT,(R3)+   ;RESTORE APT SUB TRAP
1454                                                                    ;IF ON APT
1455  011320  012713  000340          11$:    MOV     #340,(R3)       ;RESTORE PSW SAVE
1456  011324  062737  000004  002452          ADD     #4,VECLOC       ;UPDATE FOR NEXT VECTOR
1457  011332  000205                  2$:     RTS     R5              ;RETURN
1458
1459                                  ;ROUTINE TO SET UP TRAP CATCHER 204-1774 IN STANDALONE MODE OR
1460                                  ;A COMMON VECTOR ROUTINE SETUP UNDER APT.
1461                                  ;VECTOR ROUTINE IS USED ONLY IN AN APT ENVIRONMENT AND WILL STORE
1462                                  ;A COMMON SUBROUTINE IN VECTOR AREA 204-1774.
1463
1464  011334  010046                  TRPCAT: MOV     R0,-(SP)        ;SAVE R0
1465  011336  010146                          MOV     R1,-(SP)
1466  011340  022737  000001  002210          CMP     #1,$ENV         ;CHECK FOR APT
1467  011346  001426                          BEQ     APTVEC          ;YES,SET UP TRAPS FOR APT
1468  011350  012700  000204                  MOV     #204,R0         ;START CATCHER AT 204
1469  011354  032777  010000  170556          BIT     #SW12,@SWR      ;TEST IF SW12 IS SET
1470  011362  001010                          BNE     1$              ;YES,BR AND STORE REGULAR TRAP
1471                                                                  ;CATCHER(.+2,HALT) IN LOCATIONS
1472                                                                  ;204-1774
1473  011364  012720  011562          11$:    MOV     #TRPALL,(R0)+   ;NO,PLACE ILLEGAL VECTOR RETURN
1474  011370  012720  000340                  MOV     #340,(R0)+      ;ROUTINES IN LOCATIONS 204-1774
1475  011374  022700  002000                  CMP     #2000,R0        ;STORED ALL VECTORS?
1476  011400  001371                          BNE     11$
1477  011402  000421                          BR      ENDTRP          ;RETURN AFTER VECTOR STORAGE
1478  011404  010001                  1$:     MOV     R0,R1           ;DRV11J CAN VECTOR 0-1774
1479  011406  005721                          TST     (R1)+
1480  011410  010120                          MOV     R1,(R0)+
1481  011412  005020                          CLR     (R0)+
1482  011414  022701  001776                  CMP     #1776,R1        ;CHECK FOR VECTOR END
1483  011420  001371                          BNE     1$
1484  011422  000411                          BR      ENDTRP          ;RETURN IN STAND ALONE MODE
1485  011424  012701  000204          APTVE . MOV     #204,R1         ;STARTING VECTOR ADDRESS
1486  011430  012721  011544          1$:     MOV     #TRPOUT,(R1)+   ;ILL INTERRUPT ROUTINE
1487  011434  012721  000340                  MOV     #340,(R1)+      ;PSW NEXT
1488  011440  022701  002000                  CMP     #2000,R1        ;DONE?
1489  011444  001371                          BNE     1$              ;DO NEXT VECTOR
1490  011446  012601                  ENDTRP: MOV     (SP)+,R1
1491  011450  012600                          MOV     (SP)+,R0
1492  011452  000205                          RTS     R5              ;EXIT
1493
1494                                  ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 204-1774
1495  011454  005237  002436          INTSR1  INC     INTFLG          ;COUNT INTERRUPT
1496  011460  012737  000001  002124          MOV     #1,$GDDAT       ;STORE EXPECTED
1497  011466  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
1498  011474  023737  002124  002126          CMP     $GDDAT,$BDDAT   ;SHOULD BE ONE INTERRUPT
1499  011502  001401                          BEQ     1$
1500  011504  104013                          ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
1501  011506  000002                  1$:     RTI                     ;RETURN FROM INT.
1502
1503                                  ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 204-1774
1504                                  ;FOR A BYTE COUNT OF 1 TO 4 VECTORS IN VECTOR ADDR MEMORY.
1505  011510  005237  002436          INTBY4: INC     INTFLG          ;COUNT INTERRUPT
1506  011514  013737  002472  002124          MOV     BYNUM,$GDDAT    ;STORE EXPECTED
1507  011522  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
1508  011530  023737  002124  002126          CMP     $GDDAT,$BDDAT   ;INTERRUPTS SHOULD NOT EXCEED BYTE COUNT
```

```
1509  011536  002001                        BGE     1$
1510  011540  104013                        ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
1511                                                                  ;MORE INTERRUPTS THAN BYTE COUNT
1512  011542  000002                1$:     RTI                     ;RETURN FROM INT.
1513
1514                                ;COMMON ILLEGAL VECTOR RETURN ROUTINE FOR APT MODE.
1515  011544  011637  002120        TRPOUT: MOV     (SP),$GDADR     ;SAVE ADDRESS OF TEST
1516  011550  004537  011204                JSR     R5,FRMBUF       ;RETURN LOCATIONS 0-202
1517  011554  104014                        ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
1518  011556  000000                        HALT                    ;CANNOT CONTINUE
1519  011560  000000                        HALT                    ;CANNOT CONTINUE
1520
1521                                ;COMMON ILLEGAL VECTOR RETURN ROUTINE FOR VECTOR AREA
1522                                ;204-1774
1523  011562  011637  002120        TRPALL: MOV     (SP),$GDADR     ;SAVE PC OF TEST
1524  011566  104014                        ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
1525  011570  000002                        RTI                     ;RETURN
1526
1527                                ;RESTORE VECTOR JUST TESTED TO ERROR SUBROUTINE FOR 0-202
1528  011572  022737  000001  002210 RES200: CMP     #1,$ENV         ;CHECK FOR APT
1529  011600  001405                        BEQ     1$              ;YES,BRANCH
1530  011602  012723  012000                MOV     #TRP200,(R3)+   ;ERROR TRAP SUB. FOR 0-200
1531  011606  012713  000340                MOV     #340,(R3)       ;PSW
1532  011612  000404                        BR      2$              ;RETURN
1533  011614  012723  011544        1$:     MOV     #TRPOUT,(R3)+   ;RESTORE APT SUB TRAP
1534                                                                  ;IF ON APT
1535  011620  012713  000340                MOV     #340,(R3)       ;RESTORE PSW SAVE
1536  011624  062737  000004  002452 2$:     ADD     #4,VECLOC       ;UPDATE FOR NEXT VECTOR
1537  011632  000205                        RTS     R5              ;RETURN
1538
1539                                ;ROUTINE TO SET UP TRAP SUBROUTINE 0-200 IN STANDALONE MODE OR
1540                                ;A COMMON ERROR VECTOR ROUTINE SETUP UNDER APT.
1541
1542  011634  010046                CAT200: MOV     R0,-(SP)        ;SAVE R0
1543  011636  017737  170276  002460        MOV     @SWR,SWRSAV     ;SAVE LOCATION 176,SOFT SWR
1544  011644  022737  000001  002210        CMP     #1,$ENV         ;CHECK FOR APT
1545  011652  001411                        BEQ     APT200          ;YES,SET UP TRAPS FOR APT
1546  011654  005000                        CLR     R0              ;START CATCHER SUB AT 0
1547  011656  012720  012000        1$:     MOV     #TRP200,(R0)+   ;STORE VECTOR TRAP SUBROUTINE
1548  011662  012720  000340                MOV     #340,(R0)+      ;PSW
1549  011666  022700  000204                CMP     #204,R0         ;FINISHED 0-202
1550  011672  001371                        BNE     1$
1551  011674  000410                        BR      END200          ;RETURN IN STAND ALONE MODE
1552  011676  005000                APT200: CLR     R0              ;START AT VECTOR 0
1553  011700  012720  011544        1$:     MOV     #TRPOUT,(R0)+   ;VECTOR ERROR ROUTINE
1554  011704  012720  000340                MOV     #340,(R0)+      ;PSW NEXT
1555  011710  022700  000204                CMP     #204,R0         ;DONE?
1556  011714  001371                        BNE     1$              ;DO NEXT VECTOR
1557  011716  012600                END200: MOV     (SP)+,R0
1558  011720  012713  011734                MOV     #INTSR3,(R3)    ;STORE VECTOR ROUTINE
1559  011724  012763  000340  000002        MOV     #340,2(R3)      ;STORE PSW
1560  011732  000205                        RTS     R5              ;EXIT
1561
1562
1563                                ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 0-200,BYTE COUNT 0-3
1564                                ;FOR BYTE COUNTS OF 1 TO 4 VECTORS IN VECTOR ADDRESS MEMORY.
```

```
1565  011734  005237  002436          INTSR3: INC     INTFLG          ;COUNT INTERRUPT
1566  011740  013737  002472  002124          MOV     BYNUM,$GDDAT    ;STORE EXPECTED
1567  011746  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
1568  011754  023737  002124  002126          CMP     $GDDAT,$BDDAT   ;INTERRUPTS SHOULD NOT EXCEED BYTE COUNT
1569  011762  00200                          BGE     1$
1570  011764  00453?  .11204                 JSR     R5,FRMBUF       ;RESTORE 0-202 BEFORE ERROR
1571  011770  104013                         ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
1572                                                                  ;MORE INTERRUPTS THAN BYTE COUNT
1573  011772  004537  011634                 JSR     R5,CAT200       ;RESTORE VECTOR SUBROUTINES
1574  011776  000002                  1$:    RTI                     ;RETURN FROM INT.
1575
1576                                   ;COMMON ILLEGAL VECTOR ERROR RETURN ROUTINE FOR 0-200
1577  012000  011637  002120          TRP200: MOV     (SP),$GDADR     ;SAVE PC OF TEST
1578  012004  00453?  011204                  JSR     R5,FRMBUF       ;RETURN LOC 0-200
1579  012010  104014                          ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
1580  012012  004537  011634                  JSR     R5,CAT200       ;RESTORE VECTOR SUBROUTINES FOR PROCEED
1581  012016  000002                          RTI                     ;RETURN
1582
1583                                   ;VECTOR SERVICE ROUTINE FOR VECTOR UNIQUENESS TEST
1584  012020  012524                  SETVEC: MOV     (R5)+,(R4)+     ;STORE VALUE IN BUFFER
1585  012022  005726                          TST     (SP)+           ;RESTORE STACK FOR RTI
1586  012024  005237  002436                  INC     INTFLG          ;COUNT INTERRUPT
1587  012030  022737  000100  002436          CMP     #64.,INTFLG
1588  012036  002001                          BGE     1$              ;64 EXPECTED INTERRUPTS?
1589  012040  104013                          ERROR   13              ;ERROR ,MORE THAN 64 INTERRUPTS
1590  012042  000002                  1$:     RTI                     ;RETURN FOR NEXT INTERRUPT
1591
1592                                   ;INTERRUPT SERVICE ROUTINES THAT ARE STORED FROM 300-674
1593                                   ;IN THE VECTOR UNIQUENESS TEST.
1594  012044  004537  012020          INT0:   JSR     R5,SETVEC       ;STORE 300 INTO VBUF,TO BE
1595  012050  000300                          300                     ;CHECKED AFTER ALL INTERRUPTS.
1596  012052  004537  012020          INT1:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1597  012056  000304                          304
1598  012060  004537  012020          INT2:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1599  012064  000310                          310
1600  012066  004537  012020          INT3:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1601  012072  000314                          314
1602  012074  004537  012020          INT4:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1603  012100  000320                          320
1604  012102  004537  012020          INT5:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1605  012106  000324                          324
1606  012110  004537  012020          INT6:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1607  012114  000330                          330
1608  012116  004537  012020          INT7:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1609  012122  000334                          334
1610  012124  004537  012020          INT8:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1611  012130  000340                          340
1612  012132  004537  012020          INT9:   JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1613  012136  000344                          344
1614  012140  004537  012020          INT10:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1615  012144  000350                          350
1616  012146  004537  012020          INT11:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1617  012152  000354                          354
1618  012154  004537  012020          INT12:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1619  012160  000360                          360
1620  012162  004537  012020          INT13:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
```

```
1621  012166  000364                          364
1622  012170  004537  012020      INT14:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1623  012174  000370                          370
1624  012176  004537  012020      INT15:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1625  012202  000374                          374
1626  012204  004537  012020      INT16:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1627  012210  000400                          400
1628  012212  004537  012020      INT17:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1629  012216  000404                          404
1630  012220  004537  012020      INT18:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1631  012224  000410                          410
1632  012226  004537  012020      INT19:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1633  012232  000414                          414
1634  012234  004537  012020      INT20:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1635  012240  000420                          420
1636  012242  004537  012020      INT21:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1637  012246  000424                          424
1638  012250  004537  012020      INT22:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1639  012254  000430                          430
1640  012256  004537  012020      INT23:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1641  012262  000434                          434
1642  012264  004537  012020      INT24:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1643  012270  000440                          440
1644  012272  004537  012020      INT25:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1645  012276  000444                          444
1646  012300  004537  012020      INT26:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1647  012304  000450                          450
1648  012306  004537  012020      INT27:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1649  012312  000454                          454
1650  012314  004537  012020      INT28:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1651  012320  000460                          460
1652  012322  004537  012020      INT29:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1653  012326  000464                          464
1654  012330  004537  012020      INT30:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1655  012334  000470                          470
1656  012336  004537  012020      INT31:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1657  012342  000474                          474
1658  012344  004537  012020      INT32:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1659  012350  000500                          500
1660  012352  004537  012020      INT33:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1661  012356  000504                          504
1662  012360  004537  012020      INT34:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1663  012364  000510                          510
1664  012366  004537  012020      INT35:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1665  012372  000514                          514
1666  012374  004537  012020      INT36:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1667  012400  000520                          520
1668  012402  004537  012020      INT37:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1669  012406  000524                          524
1670  012410  004537  012020      INT38:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1671  012414  000530                          530
1672  012416  004537  012020      INT39:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1673  012422  000534                          534
1674  012424  004537  012020      INT40:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1675  012430  000540                          540
1676  012432  004537  012020      INT41:      JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
```

```
1677  012436  000544                          544
1678  012440  004537  012020      INT42:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1679  012444  000550                          550
1680  012446  004537  012020      INT43:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1681  012452  000554                          554
1682  012454  004537  012020      INT44:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1683  012460  000560                          560
1684  012462  004537  012020      INT45:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1685  012466  000564                          564
1686  012470  004537  012020      INT46:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1687  012474  000570                          570
1688  012476  004537  012020      INT47:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1689  012502  000574                          574
1690  012504  004537  012020      INT48:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1691  012510  000600                          600
1692  012512  004537  012020      INT49:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1693  012516  000604                          604
1694  012520  004537  012020      INT50:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1695  012524  000610                          610
1696  012526  004537  012020      INT51:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1697  012532  000614                          614
1698  012534  004537  012020      INT52:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1699  012540  000620                          620
1700  012542  004537  012020      INT53:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1701  012546  000624                          624
1702  012550  004537  012020      INT54:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1703  012554  000630                          630
1704  012556  004537  012020      INT55:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1705  012562  000634                          634
1706  012564  004537  012020      INT56:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1707  012570  000640                          640
1708  012572  004537  012020      INT57:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1709  012576  000644                          644
1710  012600  004537  012020      INT58:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1711  012604  000650                          650
1712  012606  004537  012020      INT59:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1713  012612  000654                          654
1714  012614  004537  012020      INT60:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1715  012620  000660                          660
1716  012622  004537  012020      INT61:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1717  012626  000664                          664
1718  012630  004537  012020      INT62:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1719  012634  000670                          670
1720  012636  004537  012020      INT63:  JSR     R5,SETVEC       ;VECTOR UNIQUENESS TEST, INT. SERV RTN
1721  012642  000674                          674
1722
1723                              .SBTTL  PATTERNS FOR REGISTER R/W
1724                              ;
1725                              ;PATTERNS USED FOR LOADING/READING REGISTERS
1726                              ;ISR    INTERRUPT SERVICE REGISTER
1727                              ;IRR    INTERRUPT REQUEST REGISTER
1728                              ;IMR    INTERRUPT MASK    REGISTER
1729                              ;
1730  012644    001     376       BGCHP3: .BYTE   1,376
1731  012646    002     375               .BYTE   2,375
1732  012650    004     373               .BYTE   4,373
```

```
1733  012652     010     367                    .BYTE   10,367
1734  012654     020     357                    .BYTE   20,357
1735  012656     040     337                    .BYTE   40,337
1736  012660     100     277                    .BYTE   100,277
1737  012662     200     177                    .BYTE   200,177
1738  012664  000000                  EDCHP3: 000000
1739
1740
```

```
                                        .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
                                ;*

012666  105737  002157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
012672  100002                          BPL     1$              ;;BR IF YES
012674  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
012676  000430                          BR      3$              ;;LEAVE
012700  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
012702  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
012706  122737  000001  002210          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
012714  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
012716  132737  000100  002211          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
012724  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
012726  010037  012736                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
012732  004737  013156                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
012736  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
012740  132737  000040  002211  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
012746  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
012750  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
012752  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
012754  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
012756  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
012760  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
012764  000002                          RTI                     ;;RETURN
012766  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
012772  001430                          BEQ     8$
012774  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
013000  001006                          BNE     5$
013002  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
013004  104401                          TYPE                    ;;TYPE A CR AND LF
013006  002165                          $CRLF
013010  105037  013144                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
013014  000755                          BR      2$              ;;GET NEXT CHARACTER
013016  004737  013100          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
013022  123726  002156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
013026  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
013030  013746  002154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
013034  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
013040  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
1797  013042  004737  013100                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
1798  013046  105337  013144                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1799  013052  000770                          BR      7$              ;;LOOP
1800
1801                                  ;HORIZONTAL TAB PROCESSOR
1802
1803  013054  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1804  013060  004737  013100          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
1805  013064  132737  000007  013144          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1806  013072  001372                          BNE     9$              ;;TAB STOP
1807  013074  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
1808  013076  000724                          BR      2$              ;;GET NEXT CHARACTER
1809  013100  105777  167044          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1810  013104  100375                          BPL     $TYPEC
1811  013106  116677  000002  167036          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1812  013114  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1813  013122  001003                          BNE     1$              ;;BRANCH IF NO
1814  013124  105037  013144                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1815  013130  000406                          BR      $TYPEX          ;;EXIT
1816  013132  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1817  013140  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
1818  013142  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
1819  013144  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
1820  013146  000207          $TYPEX: RTS     PC
1821
1822                                  .SBTTL  APT COMMUNICATIONS ROUTINE
1823
1824                                  ;;*****************************************************************
1825  013150  112737  000001  013414  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
1826  013156  112737  000001  013412  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
1827  013164  000403                          BR      $ATYC
1828  013166  112737  000001  013414  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
1829  013174                          $ATYC:
1830  013174  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1831  013176  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1832  013200  105737  013412                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
1833  013204  001450                          BEQ     5$              ;;IF NOT:  BR
1834  013206  122737  000001  002210          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
1835  013214  001031                          BNE     3$              ;;IF NOT:  BR
1836  013216  132737  000100  002211          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
1837  013224  001425                          BEQ     3$              ;;IF NOT:  BR
1838  013226  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
1839  013232  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1840  013240  005737  002170          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
1841  013244  001375                          BNE     1$              ;;IF NOT:  WAIT
1842  013246  010037  002204                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
1843  013252  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
1844  013254  001376                          BNE     2$
1845  013256  163700  002204                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
1846  013262  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
1847  013264  010037  002206                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
1848  013270  012737  000004  002170          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
1849  013276  000413                          BR      5$
1850  013300  017637  000004  013324  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
1851  013306  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
1852  013314  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
```

```
1853  013320  004737  012666                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
1854  013324  000000                  4$:     .WORD   0
1855  013326                          5$:
1856  013326  105737  013414          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
1857  013332  001416                          BEQ     12$             ;;IF NOT:  BR
1858  013334  005737  002210                  TST     $ENV            ;;RUNNING UNDER APT?
1859  013340  001413                          BEQ     12$             ;;IF NOT:  BR
1860  013342  005737  002170          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
1861  013346  001375                          BNE     11$             ;;IF NOT:  WAIT
1862  013350  017637  000004  002172          MOV     @4(SP),$FATAL   ;;GET ERROR #
1863  013356  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1864  013364  005237  002170                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
1865  013370  105037  013414          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
1866  013374  105037  013413                  CLRB    $LFLG           ;;CLEAR LOG FLAG
1867  013400  105037  013412                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
1868  013404  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1869  013406  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1870  013410  000207                          RTS     PC              ;;RETURN
1871  013412     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
1872  013413     000          $LFLG:  .BYTE   0               ;;LOG FLAG
1873  013414     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
1874          013416                  .EVEN
1875          000200          APTSIZE=200
1876          000001          APTENV=001
1877          000100          APTSPOOL=100
1878          000040          APTCSUP=040
1879                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
1880
1881                          ;;******************************************************************
1882                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1883                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
1884                          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1885                          ;*CALL:
1886                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1887                          ;*      TYPOS                   ;;CALL FOR TYPEOUT
1888                          ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1889                          ;*      .BYTE   M               ;;M=1 OR 0
1890                          ;*                                      ;;1=TYPE LEADING ZEROS
1891                          ;*                                      ;;0=SUPPRESS LEADING ZEROS
1892                          ;*
1893                          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1894                          ;*$TYPOS OR $TYPOC
1895                          ;*CALL:
1896                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1897                          ;*      TYPON                   ;;CALL FOR TYPEOUT
1898                          ;*
1899                          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1900                          ;*CALL:
1901                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1902                          ;*      TYPOC                   ;;CALL FOR TYPEOUT
1903
1904  013416  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
1905  013422  116637  000001  013641          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
1906  013430  112637  013643                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
1907  013434  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
1908  013440  000406                          BR      $TYPON
```

```
1909  013442  112737  000001  013641  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
1910  013450  112737  000006  013643          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
1911  013456  112737  000005  013640  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
1912  013464  010346                          MOV     R3,-(SP)        ;;SAVE R3
1913  013466  010446                          MOV     R4,-(SP)        ;;SAVE R4
1914  013470  010546                          MOV     R5,-(SP)        ;;SAVE R5
1915  013472  113704  013643                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
1916  013476  005404                          NEG     R4
1917  013500  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
1918  013504  110437  013642                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
1919  013510  113704  013641                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
1920  013514  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
1921  013520  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
1922  013522  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO 'C'
1923  013524  000404                          BR      3$              ;;GO DO MSB
1924  013526  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
1925  013530  006105                          ROL     R5
1926  013532  006105                          ROL     R5
1927  013534  010503                          MOV     R5,R3
1928  013536  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
1929  013540  053371  013642                  DECB    $OMODE          ;;TYPE THIS DIGIT?
1930  013544  100016                          BPL     7$              ;;BR IF NO
1931  013546  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
1932  013552  001002                          BNE     4$              ;;TEST FOR 0
1933  013554  005704                          TST     R4              ;;SUPPRESS THIS 0?
1934  013556  001403                          BEQ     5$              ;;BR IF YES
1935  013560  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
1936  013562  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
1937  013566  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
1938  013572  110337  013636                  MOVB    R3,8$           ;;SAVE FOR TYPING
1939  013576  104401  013636                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
1940  013602  105337  013640          7$:     DECB    $OCNT           ;;COUNT BY 1
1941  013606  003347                          BGT     2$              ;;BR IF MORE TO DO
1942  013610  002402                          BLT     6$              ;;BR IF DONE
1943  013612  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
1944  013614  000744                          BR      2$              ;;GO DO THE LAST DIGIT
1945  013616  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
1946  013620  012604                          MOV     (SP)+,R4        ;;RESTORE R4
1947  013622  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1948  013624  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
1949  013632  012616                          MOV     (SP)+,(SP)
1950  013634  000002                          RTI                     ;;RETURN
1951  013636  000                     8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
1952  013637  000                             .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
1953  013640  000             $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
1954  013641  000             $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
1955  013642  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
1956                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1957
1958                          ;;**********************************************************************
1959                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
1960                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
1961                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
1962                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
1963                          ;*REPLACED WITH SPACES.
1964                          ;*CALL:
```

```
                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                        ;*      TYPDS                   ;;GO TO THE ROUTINE
                        $TYPDS:
013644 010046           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013646 010146           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013650 010246           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013652 010346           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
013654 010546           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013656 012746 020200    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
013662 016605 000020    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
013666 100004           BPL     1$              ;;BR IF INPUT IS POS.
013670 005405           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
013672 112766 000055 000001     MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
013700 005000           1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
013702 012703 014060    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
013706 112723 000040    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
013712 005002           2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
013714 016001 014050    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
013720 160105           3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
013722 002402           BLT     4$              ;;BR IF DONE
013724 005202           INC     R2              ;;INCREASE THE BCD DIGIT BY 1
013726 000774           BR      3$
013730 060105           4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
013732 005702           TST     R2              ;;CHECK IF BCD DIGIT=0
013734 001002           BNE     5$              ;;FALL THROUGH IF 0
013736 105716           TSTB    (SP)            ;;STILL DOING LEADING 0'S?
013740 100407           BMI     7$              ;;BR IF YES
013742 106316           5$:     ASLB    (SP)            ;;MSD?
013744 103003           BCC     6$              ;;BR IF NO
013746 116663 000001 177777     MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
013754 052702 000060    6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
013760 052702 000040    7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
013764 110223           MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
013766 005720           TST     (R0)+           ;;JUST INCREMENTING
013770 020027 000010    CMP     R0,#10          ;;CHECK THE TABLE INDEX
013774 002746           BLT     2$              ;;GO DO THE NEXT DIGIT
013776 003002           BGT     8$              ;;GO TO EXIT
014000 010502           MOV     R5,R2           ;;GET THE LSD
014002 000764           BR      6$              ;;GO CHANGE TO ASCII
014004 105726           8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
014006 100003           BPL     9$              ;;BR IF NO
014010 116663 177777 177776     MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
014016 105013           9$:     CLRB    (R3)            ;;SET THE TERMINATOR
014020 012605           MOV     (SP)+,R5        ;;POP STACK INTO R5
014022 012603           MOV     (SP)+,R3        ;;POP STACK INTO R3
014024 012602           MOV     (SP)+,R2        ;;POP STACK INTO R2
014026 012601           MOV     (SP)+,R1        ;;POP STACK INTO R1
014030 012600           MOV     (SP)+,R0        ;;POP STACK INTO R0
014032 104401 014060    TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
014036 016666 000002 000004     MOV     2(SP),4(SP)     ;;ADJUST THE STACK
014044 012616           MOV     (SP)+,(SP)
014046 000002           RTI                     ;;RETURN TO USER
014050 023420           $DTBL:  10000.
014052 001750           1000.
014054 000144           100.
```

```
2021  014056  000012                        10.
2022  014060  000004                $DBLK:  .BLKW   4
2023                                .SBTTL  ERROR HANDLER ROUTINE
2024
2025                                ;;*****************************************************************
2026                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2027                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2028                                ;*AND GO TO SWRCK ON ERROR
2029                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2030                                ;*SW15=1         HALT ON ERROR
2031                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
2032                                ;*SW09=1         LOOP ON ERROR
2033                                ;*CALL
2034                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2035
2036  014070                        $ERROR:
2037  014070  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2038  014072  104407                        CKSWR                   ;GO LOOK FOR SWR CHANGE
2039  014074  105237  002103        7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2040  014100  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2041  014102  013777  002102  166032        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2042  014110  005237  002112                INC     $ERTTL          ;;INC THE ERROR COUNT
2043  014114  011637  002116                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2044  014120  162737  000002  002116        SUB     #2,$ERRPC
2045  014126  117737  165764  002114        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2046  014134  032777  020000  165776        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2047  014142  001004                        BNE     20$             ;;SKIP TYPEOUTS
2048  014144  004737  014244                JSR     PC,SWRCK        ;;GO TO USER ERROR ROUTINE
2049  014150  104401  002165                TYPE    ,$CRLF
2050  014154                        20$:
2051  014154  122737  000001  002210        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
2052  014162  001007                        BNE     2$              ;;NO,SKIP APT ERROR REPORT
2053  014164  113737  002114  014176        MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
2054  014172  004737  013166                JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
2055  014176     000                21$:    .BYTE   0
2056  014177     000                        .BYTE   0
2057  014200  000777                22$:    BR      22$             ;;APT ERROR LOOP
2058  014202  005777  165732        2$:     TST     @SWR            ;;HALT ON ERROR
2059  014206  100002                        BPL     3$              ;;SKIP IF CONTINUE
2060  014210  000000                        HALT                    ;;HALT ON ERROR!
2061  014212  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2062  014214  032777  001000  165716  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
2063  014222  001402                        BEQ     4$              ;;BR IF NO
2064  014224  013716  002110                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
2065  014230  005737  002162        4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
2066  014234  001402                        BEQ     5$              ;;BR IF NONE
2067  014236  013716  002162                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
2068  014242                        5$:
2069  014242  000002                        RTI                     ;;RETURN
2070                                ;;*****************************************************************
2071                                ;GO TYPE ERROR
2072                                ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
2073                                ;;*****************************************************************
2074  014244  113737  002102  002432  SWRCK:  MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ER
2075  014252  004737  014262                JSR     PC,$ERRTYP      ;GO TYPE ERROR
2076  014256  104407                        CKSWR                   ;GO LOOK FOR SWR CHANGE
```

```
2077  014260  000207                        RTS     PC              ;RETURN TO ERROR HANDLER
2078                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2079
2080                                  ;;*****************************************************************
2081                                  ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
2082                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE ''ERROR TABLE'' ($ERRTB),
2083                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2084
2085  014262                          $ERRTYP: ''
2086  014262  104401  002165                  TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
2087  014266  010046                          MOV     R0,-(SP)        ;;SAVE R0
2088  014270  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
2089  014272  153700  002114                  BISB    @#$ITEMB,R0
2090  014276  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
2091                                                                  ;;TYPE THE PC OF THE ERROR
2092  014300  013746  002116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
2093                                                                  ;;ERROR ADDRESS
2094  014304  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2095  014306  000426                          BR      6$              ;;GET OUT
2096  014310  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
2097  014312  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
2098  014314  006300                          ASL     R0
2099  014316  006300                          ASL     R0
2100  014320  062700  002252                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
2101  014324  012037  014334                  MOV     (R0)+,2$        ;;PICKUP ''ERROR MESSAGE'' POINTER
2102  014330  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
2103  014332  104401                          TYPE                    ;;TYPE THE ''ERROR MESSAGE''
2104  014334  000000                  2$:     .WORD   0               ;;''ERROR MESSAGE'' POINTER GOES HERE
2105  014336  104401  002165                  TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
2106  014342  012037  014352          3$:     MOV     (R0)+,4$        ;;PICKUP ''DATA HEADER'' POINTER
2107  014346  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
2108  014350  104401                          TYPE                    ;;TYPE THE ''DATA HEADER''
2109  014352  000000                  4$:     .WORD   0               ;;''DATA HEADER'' POINTER GOES HERE
2110  014354  104401  002165                  TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
2111  014360  011000                  5$:     MOV     (R0),R0         ;;PICKUP ''DATA TABLE'' POINTER
2112  014362  001004                          BNE     7$              ;;GO TYPE THE DATA
2113  014364  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
2114  014366  104401  002165                  TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
2115  014372  000207                          RTS     PC              ;;RETURN
2116  014374                          7$:
2117  014374  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
2118  014376  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2119  014400  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
2120  014402  001770                          BEQ     6$              ;;BR IF NO
2121  014404  104401  014412                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
2122  014410  000771                          BR      7$              ;;LOOP
2123  014412  020040      000         8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
2124            014416                        .EVEN
2125                                  .SBTTL  SCOPE HANDLER ROUTINE
2126
2127                                  ;;*****************************************************************
2128                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2129                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2130                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2131                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2132                                  ;*SW14=1         LOOP ON TEST
```

```
                                        ;*SW11=1        INHIBIT ITERATIONS
                                        ;*SW09=1        LOOP ON ERROR
                                        ;*SW08=1        LOOP ON TEST IN SWR<7:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT

014416                                  $SCOPE:
014416  104407                                 CKSWR                           ;;TEST  FOR CHANGE IN SOFT-SWR
014420  032777  040000  165512          1$:     BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
014426  001114                                  BNE     $OVER                   ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
014430  000416                          $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
014432  013746  000004                          MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
014436  012737  014456  000004                  MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
014444  005737  177060                          TST     @#177060                ;;TIME OUT ON XOR?
014450  012637  000004                          MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
014454  000463                                  BR      $SVLAD                  ;;GO TO THE NEXT TEST
014456  022626                          5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
014460  012637  000004                          MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
014464  000423                                  BR      7$                      ;;LOOP ON THE PRESENT TEST
014466                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
014466  032777  000400  165444                  BIT     #BIT08,@SWR             ;;LOOP ON SPEC. TEST?
014474  001404                                  BEQ     2$                      ;;BR IF NO
014476  127737  165436  002102                  CMPB    @SWR,$TSTNM             ;;ON THE RIGHT TEST?    SWR<7:0>
014504  001465                                  BEQ     $OVER                   ;;BR IF YES
014506  105737  002103                  2$:     TSTB    $ERFLG                  ;;HAS AN ERROR OCCURRED?
014512  001421                                  BEQ     3$                      ;;BR IF NO
014514  123737  002115  002103                  CMPB    $ERMAX,$ERFLG           ;;MAX. ERRORS FOR THIS TEST OCCURRED?
014522  101015                                  BHI     3$                      ;;BR IF NO
014524  032777  001000  165406                  BIT     #BIT09,@SWR             ;;LOOP ON ERROR?
014532  001404                                  BEQ     4$                      ;;BR IF NO
014534  013737  002110  002106          7$:     MOV     $LPERR,$LPADR           ;;SET LOOP ADDRESS TO LAST SCOPE
014542  000446                                  BR      $OVER
014544  105037  002103                  4$:     CLRB    $ERFLG                  ;;ZERO THE ERROR FLAG
014550  005037  002160                          CLR     $TIMES                  ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
014554  000415                                  BR      1$                      ;;ESCAPE TO THE NEXT TEST
014556  032777  004000  165354          3$:     BIT     #BIT11,@SWR             ;;INHIBIT ITERATIONS?
014564  001011                                  BNE     1$                      ;;BR IF YES
014566  005737  002176                          TST     $PASS                   ;;IF FIRST PASS OF PROGRAM
014572  001406                                  BEQ     1$                      ;;      INHIBIT ITERATIONS
014574  005237  002104                          INC     $ICNT                   ;;INCREMENT ITERATION COUNT
014600  023737  002160  002104                  CMP     $TIMES,$ICNT            ;;CHECK THE NUMBER OF ITERATIONS MADE
014606  002024                                  BGE     $OVER                   ;;BR IF MORE ITERATION REQUIRED
014610  012737  000001  002104  1$:             MOV     #1,$ICNT                ;;REINITIALIZE THE ITERATION COUNTER
014616  013737  014674  002160                  MOV     $MXCNT,$TIMES           ;;SET NUMBER OF ITERATIONS TO DO
014624  105237  002102                  $SVLAD: INCB    $TSTNM                  ;;COUNT TEST NUMBERS
014630  113737  002102  002174                  MOVB    $TSTNM,$TESTN           ;;SET TEST NUMBER IN APT MAILBOX
014636  011637  002106                          MOV     (SP),$LPADR             ;;SAVE SCOPE LOOP ADDRESS
014642  011637  002110                          MOV     (SP),$LPERR             ;;SAVE ERROR LOOP ADDRESS
014646  005037  002162                          CLR     $ESCAPE                 ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
014652  112737  000001  002115                  MOVB    #1,$ERMAX               ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
014660  013777  002102  165254  $OVER:         MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER
014666  013716  002106                          MOV     $LPADR,(SP)             ;;FUDGE RETURN ADDRESS
014672  000002                                  RTI                             ;;FIXES PS
014674  000001                  $MXCNT: 1.                                      ;;MAX. NUMBER OF ITERATIONS
```

```
                                        .SBTTL  TTY INPUT ROUTINE
                                        ;;*********************************************************************
                                        .ENABL  LSB
                                        ;;*********************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                        ;*WHEN OPERATING IN TTY FLAG MODE.
014676  022737  000176  002140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
014704  001074                        BNE     15$             ;;BRANCH IF NO
014706  105777  165232                TSTB    @$TKS           ;;CHAR THERE?
014712  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
014714  117746  165226                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
014720  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
014724  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
014730  001062                        BNE     15$             ;;NO, RETURN TO USER
014732  123727  002134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
014740  001456                        BEQ     15$             ;;BRANCH IF YES
                                
014742  104401  015423                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
014746  104401  015430        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
014752  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
014756  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014760  104401  015441                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
014764  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
014766  005046                        CLR     -(SP)           ;;THE NEW SWR
014770  105777  165150        7$:     TSTB    @$TKS           ;;CHAR THERE?
014774  100375                        BPL     7$              ;;IF NOT TRY AGAIN
                                
014776  117746  165144                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
015002  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
                                
                                
                                
015006  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
015012  001005                        BNE     10$             ;;BRANCH IF NOT
015014  104401  015416                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
015020  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
015024  000757                        BR      19$             ;;LET'S TRY IT AGAIN
                                
                                
015026  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
015032  001022                        BNE     16$             ;;BRANCH IF NO
015034  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
015040  001403                        BEQ     11$             ;;BRANCH IF YES
015042  016677  000002  165070        MOV     2(SP),@SWR      ;;SAVE NEW SWR
015050  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
015054  104401  002165        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
015060  123727  002135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
015066  001003                        BNE     15$             ;;BRANCH IF NOT
015070  012777  000100  165046        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
015076  000002                15$:    RTI                     ;;RETURN
015100  004737  013100        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
015104  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
```

```
2245  015110  002420                        BLT     18$             ;;BRANCH IF YES
2246  015112  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
2247  015116  003015                        BGT     18$             ;;BRANCH IF YES
2248  015120  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2249  015124  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
2250  015130  001403                        BEQ     17$             ;;BRANCH IF YES
2251  015132  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
2252  015134  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
2253  015136  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
2254  015140  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2255  015144  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2256  015150  000707                        BR      7$              ;;GET THE NEXT ONE
2257  015152  104401  002164        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
2258  015156  000720                        BR      20$             ;;SIMULATE CONTROL-U
2259                                .DSABL  LSB
2260
2261
2262                                ;;*****************************************************************
2263                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2264                                ;*CALL:
2265                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2266                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2267                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
2268                                ;
2269
2270  015160  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2271  015162  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
2272  015170  105777  164750        1$:     TSTB    @$TKS           ;;WAIT FOR
2273  015174  100375                        BPL     1$              ;;A CHARACTER
2274  015176  117766  164744  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
2275  015204  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2276  015212  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
2277  015220  001013                        BNE     3$              ;;BRANCH IF NO
2278  015222  105777  164716        2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
2279  015226  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
2280  015230  117746  164712                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
2281  015234  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2282  015240  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2283  015244  001366                        BNE     2$              ;;IF NOT DISCARD IT
2284  015246  000750                        BR      1$              ;;YES, RESUME
2285  015250  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
2286  015256  002407                        BLT     4$              ;;BRANCH IF YES
2287  015260  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2288  015266  003003                        BGT     4$              ;;BRANCH IF YES
2289  015270  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2290  015276  000002                4$:     RTI                     ;;GO BACK TO USER
2291                                ;;*****************************************************************
2292                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2293                                ;*CALL:
2294                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2295                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2296                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2297
2298  015300  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2299  015302  012703  015406        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2300  015306  022703  015416        2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
```

```
2301  015312  101405                  BLOS    4$              ;;BR IF YES
2302  015314  104410                  RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2303  015316  112613                  MOVB    (SP)+,(R3)      ;;GET CHARACTER
2304  015320  122713  000177  10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2305  015324  001003                  BNE     3$              ;;SKIP IF NOT
2306  015326  104401  002164  4$:     TYPE    ,$QUES          ;;TYPE A '?'
2307  015332  000763                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
2308  015334  111337  015404  3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2309  015340  104401  015404          TYPE    ,9$
2310  015344  122723  000015          CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2311  015350  001356                  BNE     2$              ;;LOOP IF NOT RETURN
2312  015352  105063  177777          CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2313  015356  104401  002166          TYPE    ,$LF            ;;TYPE A LINE FEED
2314  015362  012603                  MOV     (SP)+,R3        ;;RESTORE R3
2315  015364  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2316  015366  016666  000004  000002  MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2317  015374  012766  015406  000004  MOV     #$TTYIN,4(SP)
2318  015402  000002                  RTI                     ;;RETURN
2319  015404     000  9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
2320  015405     000          .BYTE   0                       ;;TERMINATOR
2321  015406  000010  $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
2322  015416  052536  005015  000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
2323  015423     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
2324  015430  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2325  015436  020075     000
2326  015441     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2327  015446  036440  000040
2328                                  .SBTTL  POWER DOWN AND UP ROUTINES
2329
2330                                  ;;*****************************************************************
2331                                  ;POWER DOWN ROUTINE
2332  015452  012737  015616  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2333  015460  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2334  015466  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2335  015470  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2336  015472  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2337  015474  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2338  015476  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2339  015500  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2340  015502  017746  164432          MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
2341  015506  010637  015622          MOV     SP,$SAVR6       ;;SAVE SP
2342  015512  012737  015524  000024  MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2343  015520  000000                  HALT
2344  015522  000776                  BR      .-2             ;;HANG UP
2345
2346                                  ;;*****************************************************************
2347                                  ;POWER UP ROUTINE
2348  015524  012737  015616  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2349  015532  013706  015622          MOV     $SAVR6,SP       ;;GET SP
2350  015536  005037  015622          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2351  015542  005237  015622  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
2352  015546  001375                  BNE     1$              ;;OF  WORD
2353  015550  012677  164364          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
2354  015554  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
2355  015556  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
2356  015560  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
2357  015562  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
2358  015564  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
2359  015566  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
2360  015570  012737  015452  000024      MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2361  015576  012737  000340  000026      MOV     #340,@#PWRVEC+2 ;;PRIO:7
2362  015604  104401                      TYPE                    ;;REPORT THE POWER FAILURE
2363  015606  015624              $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
2364  015610  012716                      MOV     (PC)+,(SP)      ;;RESTART AT START1
2365  015612  003066              $PWRAD: .WORD   START1          ;;RESTART ADDRESS
2366  015614  000002                      RTI
2367  015616  000000              $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
2368  015620  000776                      BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2369  015622  000000              $SAVR6: 0                       ;;PUT THE SP HERE
2370  015624  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
2371  015632  051101  042524  020104
2372  015640  051106  046517  050040
2373  015646  051127  043040  044501
2374  015654  000114
2375                                  .EVEN
2376                                  .SBTTL  TRAP DECODER
2377
2378                                  ;;**************************************************************
2379                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2380                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2381                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2382                                  ;*GO TO THAT ROUTINE.
2383
2384  015656  010046              $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2385  015660  016600  000002              MOV     2(SP),R0        ;;GET TRAP ADDRESS
2386  015664  005740                      TST     -(R0)           ;;BACKUP BY 2
2387  015666  111000                      MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2388  015670  006300                      ASL     R0              ;;POSITION FOR INDEXING
2389  015672  016000  015712              MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2390  015676  000200                      RTS     R0              ;;GO TO ROUTINE
2391
2392
2393                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2394
2395  015700  011646              $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2396  015702  016666  000004  000002      MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2397  015710  000002                      RTI                     ;;RESTORE THE PSW
2398
2399                                  .SBTTL  TRAP TABLE
2400
2401                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2402                                  ;*BY THE "TRAP" INSTRUCTION.
2403
2404                                  ;       ROUTINE
2405                                  ;       -------
2406  015712  015700              $TRPAD: .WORD   $TRAP2
2407  015714  012666                      $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2408  015716  013442                      $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2409  015720  013416                      $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2410  015722  013456                      $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2411  015724  013644                      $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
2412
```

```
                                        $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
                                        $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
                                        $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
                                        $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE

                                .SBTTL  ASCII MESSAGES
016736  005015  053103  051104  TITLED: .ASCIZ  <15><12>/CVDRDA  DRV11J DIAG TEST PART 2/<15><12>
016744  040504  020040  051104
016752  030526  045061  042040
016760  040511  020107  042524
016766  052123  050040  051101
016774  020124  006462  000012
016002  005015  051104  030526  TLCABL: .ASCIZ  <15><12>/DRV11J CABLE REQ'D/<15><12>
016010  045061  041440  041101
016016  042514  051040  050505
016024  042047  005015     000
016031     122  043505  052040  EM1:    .ASCIZ  /REG TIMEOUT ER/
016036  046511  047505  052125
016044  042440  000122
016050  042522  020107  042522  EM2:    .ASCIZ  'REG READ/WRITE ER'
016056  042101  053457  044522
016064  042524  042440  000122
016072  051111  020122  042522  EM3:    .ASCIZ  /IRR REG ER/
016100  020107  051105     000
016105     101  051103  051040  EM4:    .ASCIZ  /ACR REG ER/
016112  043505  042440  000122
016120  046511  020122  042522  EM5:    .ASCIZ  /IMR REG ER/
016126  020107  051105     000
016133     111  051123  051040  EM6:    .ASCIZ  /ISR REG ER/
016140  043505  042440  000122
016146  046111  042514  040507  EM7:    .ASCIZ  /ILLEGAL VECTOR MEM ADDR ER/
016154  020114  042526  052103
016162  051117  046440  046505
016170  040440  042104  020122
016176  051105     000
016201     103  044510  020120  EM10:   .ASCIZ  /CHIP STAT ER/
016206  052123  052101  042440
016214  000122
016216  046111  042514  040507  EM11:   .ASCIZ  /ILLEGAL INTERRUPT RECEIVED/
016224  020114  047111  042524
016232  051122  050125  020124
016240  042522  042503  053111
016246  042105     000
016251     111  052116  051105  EM12:   .ASCIZ  /INTERRUPT TEST ERROR/
016256  052522  052120  052040
016264  051505  020124  051105
016272  047522  000122
016276  052515  052114  050111  EM13:   .ASCIZ  /MULTIPLE INTERRUPTS RECEIVED/
016304  042514  044440  052116
016312  051105  052522  052120
016320  020123  042522  042503
016326  053111  042105     000
016333     126  041505  020124  EM14:   .ASCIZ  /VECT ADDR MEM ER/
016340  042101  051104  046440
```

```
2469  016346  046505  042440  000122
2470  016354  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
2471  016362  020040  051524  047124
2472  016370  046525  020040  052502
2473  016376  040523  051104  020040
2474  016404  054105  041520  020124
2475  016412  020040  041522  042126
2476  016420     000
2477  016421     105  051122  041520  DH2:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD/
2478  016426  020040  052040  052123
2479  016434  052516  020115  041040
2480  016442  051525  042101  020122
2481  016450  053040  046501  020040
2482  016456  020040  040440  042104
2483  016464  020122  020040  042440
2484  016472  050130  052103  020040
2485  016500  051040  053103  000104
2486  016506  051105  050122  020103  DH3:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  VAM     ADDR/
2487  016514  020040  051524  047124
2488  016522  046525  020040  052502
2489  016530  040523  051104  020040
2490  016536  040526  020115  020040
2491  016544  020040  042101  051104
2492  016552     000
2493  016553     105  051122  041520  DH4:    .ASCIZ  /ERRPC   TSTNUM  TESTPC  BUSADR  VAM     ADDR/
2494  016560  052011  052123  052516
2495  016566  004515  042524  052123
2496  016574  041520  041011  051525
2497  016602  042101  020122  053040
2498  016610  046501  020040  020040
2499  016616  040440  042104  000122
2500
2501                                  .EVEN
2502  016624  002116  002432  002122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
2503  016632  002124  002126  000000
2504  016640  002116  002432  002122  DT2:    $ERRPC,TSTNUM,$BDADR,VECVAL,VECLOC,$GDDAT,$BDDAT,0
2505  016646  002456  002452  002124
2506  016654  002126  000000
2507  016660  002116  002432  002122  DT3:    $ERRPC,TSTNUM,$BDADR,VECVAL,VECLOC,0
2508  016666  002456  002452  000000
2509  016674  002116  002432  002120  DT4:    $ERRPC,TSTNUM,$GDADR,$BDADR,VECVAL,VECLOC,0
2510  016702  002122  002456  002452
2511  016710  000000
2512
2513
2514                          ;;*****************************************************************
2515                          ;'DBUF' IS THE STORAGE AREA FOR SYSMAC LOCATIONS 0-202
2516                          ;PRESERVED IN ORDER TO TEST DRV11J VECTORING TO
2517                          ;VECTOR SPACE 0-200.
2518                          ;;*****************************************************************
2519  016712  000102          DBUF:   .BLKW   66.             ;1ST ADRS OF DATA STORAGE AREA
2520
2521                          ;;*****************************************************************
2522                          ;VECTOR BUFFER USED FOR VECTOR UNIQUENESS TEST
2523                          ;FOR 64 INTERRUPT VECTORS.
2524                          ;;*****************************************************************
```

```
2525  017116  000100          VBUF:   .BLKW   64.
2526  017316                  ENDBUF:
2527          000001                  .END
```

```
ABASE = 164160              135#    269     310     406     407     408     409     410     411     412     413
ACDW1 = 000000              269     312
ACDW2 = 000000              269
ACPUOP= 000000              269     284
ACRLOC  002450              425#
ADDW0 = 000000              269
ADDW1 = 000000              269
ADDW10= 000000              269
ADDW11= 000000              269
ADDW12= 000000              269
ADDW13= 000000              269
ADDW14= 000000              269
ADDW15= 000000              269
ADDW2 = 000000              269
ADDW3 = 000000              269
ADDW4 = 000000              269
ADDW5 = 000000              269
ADDW6 = 000000              269
ADDW7 = 000000              269
ADDW8 = 000000              269
ADDW9 = 000000              269
ADEVCT= 000000              269     275
ADEVM = 000001              136#    269     311
AENV  = 000000              269     280
AENVM = 000000              269     281
AFATAL= 000000              269     272
AMADR1= 000000              269     297
AMADR2= 000000              269     301
AMADR3= 000000              269     304
AMADR4= 000000              269     307
AMAMS1= 000000              269     291
AMAMS2= 000000              269     299
AMAMS3= 000000              269     302
AMAMS4= 000000              269     305
AMSGAD= 000000              269     277
AMSGLG= 000000              269     278
AMSGTY= 000000              269     271
AMTYP1= 000000              269     292
AMTYP2= 000000              269     300
AMTYP3= 000000              269     303
AMTYP4= 000000              269     306
APASS = 000000              269     274
APRIOR= 000000              269
APTCSU= 000040             1773    1878#
APTENV= 000001             1766    1834    1876#   2051
APTSIZ= 000200              476    1875#
APTSPO= 000100             1768    1836    1877#
APTVEC  011424             1467    1485#
APT200  011676             1545    1552#
ASWREG= 000000              269     282
ATESTN= 000000              269     273
AUNIT = 000000              269     276
AUSWR = 000000              269     283
AVECT1= 000000              269     308
AVECT2= 000000              269     309
BDSAV   002464              431#
```

```
BGCHP3  012644            541    655    747    870   1730#
BIT0   = 000001            120# 1366
BIT00  = 000001            110#  120
BIT01  = 000002            109#  119
BIT02  = 000004            108#  118
BIT03  = 000010            107#  117
BIT04  = 000020            106#  116
BIT05  = 000040            105#  115
BIT06  = 000100            104#  114
BIT07  = 000200            103#  113
BIT08  = 000400            102#  112   2155
BIT09  = 001000            101#  111   2062   2163
BIT1   = 000002            119#  570    674    790    917   1046   1175
BIT10  = 002000            100#
BIT11  = 004000             99# 2170
BIT12  = 010000             98#
BIT13  = 020000             97# 2046
BIT14  = 040000             96# 2141
BIT15  = 100000             95#  137
BIT2   = 000004            118#
BIT3   = 000010            117#
BIT4   = 000020            116#
BIT5   = 000040            115#
BIT6   = 000100            114#
BIT7   = 000200            113#
BIT8   = 000400            112#  138
BIT9   = 001000            111#  139    626    662    757    878
BPTVEC= 000014             127#
BYCNT   002470             433#  769*   772*   891*   895*  1383*  1386*
BYNUM   002472             434#  744*   769    791    845*   867*   891    918    984*  1506   1566
CAT200  011634             884   906    914    926    938    946    954    962    971  1542#  1573   1580
CHPISR= 000140             155#  612   1182   1196
CIMR   = 000040            145# 1024   1036   1154   1165
CIRMR  = 000020            142#
CIRR   = 000100            150# 1186   1200
CISR   = 000160            156#  964
CKSWR  = 104407            846   985   2037   2038   2061   2076   2140  2415#
CLRBF   011144            1008  1138   1411#
CLRCSR  010650             540   654    745    868   1007   1137  1347#
CLRIRR  010730             543   657    758    879   1013   1143  1363#
CR     = 000015             35# 1812   1822
CRLF   = 000200             36# 1783   1822
CSIMR  = 000050            146#  542    656    749    872
CSIRMR= 000030             143#
CSIRR  = 000110            151#
CSISR  = 000170            157#  751
DBUF    016712            1421  1432   2519#
DDISP  = 177570             42#  251    464
DH1     016354             332   368   2470#
DH2     016421             338   344    350    356    362    374    386    392  2477#
DH3     016506             380  2486#
DH4     016553             398  2493#
DIR    = 000400            138#
DISPLA  002142             251#  464*   472*  2041*  2185*
DISPRE  000174             180#  472
DMAP    002434             419#  510*   511*   514   1295*  1299
```

```
DRCSA   002412  406# 513* 517  535  537  649  651  740  742  863  865  1017 1147
                1293 1351 1364 1376
DRCSB   002416  408# 536  650  741  864  1019 1149 1377
DRCSC   002422  410# 639  729  853  992  1018 1148 1394
DRCSD   002426  412# 640  730  854  994  1020 1150 1395
DRDBA   002414  407#
DRDBB   002420  409#
DRDBC   002424  411#
DRDBD   002430  413# 520
DSWR  = 177570   41# 250  463
DT1     016624  333  369  2502#
DT2     016640  339  345  351  357  363  375  387  393  2504#
DT3     016660  381  2507#
DT4     016674  399  2509#
EDCHP3  012664  633  723  1738#
EMTVEC= 000030  130# 448* 449*
EM1     016031  331  2431#
EM10    016201  373  2450#
EM11    016216  379  2453#
EM12    016251  385  2458#
EM13    016276  391  2462#
EM14    016333  367  2467#
EM2     016050  337  2434#
EM3     016072  343  2437#
EM4     016105  349  2439#
EM5     016120  355  2441#
EM6     016133  361  2443#
EM7     016146  397  2445#
ENDBUF  017316  2526#
ENDIRP  011446  1477 1484 1490#
END200  011716  1551 1557#
ERRVEC= 000004  123# 461  462* 473* 2146 2147* 2149* 2152*
FRMBUF  011204  904  912  924  936  944  952  960  969  973  998  1429# 1516 1570
                1578
GNS   = ******  U  179  2407 2408 2409 2410 2411 2413 2415 2416 2417
GRPCNT  002462  430# 539* 637* 653* 727* 743* 851* 866* 990*
GTSWR = 104406  508  2413#
HT    = 000011   33# 1781 1822
IE    = 001000  139#
IMRLOC  002442  422# 542* 552  635* 656* 671  725* 749* 776  838* 872* 899  978*
INTBY4  011510  766  1505#
INTFLG  002436  420# 546* 559  562* 565  568* 572  660* 675  678* 681  684* 691
                768* 779  782* 785  788* 793  890* 902  907* 910  915* 920  1025*
                1030 1033* 1041 1044* 1048 1155* 1160 1163* 1170 1173* 1177 1188* 1191
                1495* 1497 1505* 1507 1565* 1567 1586* 1587
INTSR1  011454  544  658  1495#
INTSR3  011734  888  1558 1565#
INT0    012044  1401 1594#
INT1    012052  1596#
INT10   012140  1614#
INT11   012146  1616#
INT12   012154  1618#
INT13   012162  1620#
INT14   012170  1622#
INT15   012176  1624#
INT16   012204  1626#
```

```
INT17   012212          1628#
INT18   012220          1630#
INT19   012226          1632#
INT2    012060          1598#
INT20   012234          1634#
INT21   012242          1636#
INT22   012250          1638#
INT23   012256          1640#
INT24   012264          1642#
INT25   012272          1644#
INT26   012300          1646#
INT27   012306          1648#
INT28   012314          1650#
INT29   012322          1652#
INT3    012066          1600#
INT30   012330          1654#
INT31   012336          1656#
INT32   012344          1658#
INT33   012352          1660#
INT34   012360          1662#
INT35   012366          1664#
INT36   012374          1666#
INT37   012402          1668#
INT38   012410          1670#
INT39   012416          1672#
INT4    012074          1602#
INT40   012424          1674#
INT41   012432          1676#
INT42   012440          1678#
INT43   012446          1680#
INT44   012454          1682#
INT45   012462          1684#
INT46   012470          1686#
INT47   012476          1688#
INT48   012504          1690#
INT49   012512          1692#
INT5    012102          1604#
INT50   012520          1694#
INT51   012526          1696#
INT52   012534          1698#
INT53   012542          1700#
INT54   012550          1702#
INT55   012556          1704#
INT56   012564          1706#
INT57   012572          1708#
INT58   012600          1710#
INT59   012606          1712#
INT6    012110          1606#
INT60   012614          1714#
INT61   012622          1716#
INT62   012630          1718#
INT63   012636          1720#
INT7    012116          1608#
INT8    012124          1610#
INT9    012132          1612#
IOTVEC= 000020           128#    446*    447*
```

```
IRRLOC  002446          424#    750*    783     839*    873*    908     979*
ISRLOC  002444          423#    751*    827     840*
LF      = 000012         34#   1816    1822
LMD04   = 000200        159#
LMD57   = 000240        160#
LVLCNT  002466          432#    748*    842*    871*    981*   1381*   1388*
LVLSAV  002474          435#   1152*   1184*   1187*   1198*
MACR    = 000254        166#    606     717     821     956    1086    1110    1233    1255
MIMR    = 000244        164#    594     705     809     940    1075    1099    1223    1244
MIRR    = 000250        165#    600     711     815     948    1080    1105    1228    1250
MISR    = 000240        163#    588     613     699     803     828     932     965    1070    1094    1218    1239
NEXPAS  003132          515     517#   1301
NEXPA1  003136          518#    521
NEXPA2  003202          526     528#
NXDEV   010466         1293#
NXDEV1  010472          516    1294#   1300
PACR    = 000300        169#    665    1022    1034
PIMR    = 000260        170#
PIRQ    = 177772         40#
PIRQVE= 000240          134#
PR0     = 000000         57#    554*    569*    673*    685*    777*    789*    900*    916*   1026*   1037*   1045*   1156*
                       1166*   1174*   1189*
PR1     = 000040         58#
PR2     = 000100         59#
PR3     = 000140         60#
PR4     = 000200         61#
PR5     = 000240         62#
PR6     = 000300         63#
PR7     = 000340         64#    528*    564*    576*    680*    695*    784*    798*    909*    927*   1040*   1049*   1169*
                       1178*   1192*   1348*
PS      = 177776         37#     38
PSW     = 177776         38#
PVMA    = 000340        171#    548     667     746     869    1349
PWRMSG  015624         2363    2370#
PWRVEC= 000024          129#    452*    453*   2332*   2333*   2342*   2348*   2360*   2361*
RDCHR   = 104410       2302    2416#
RDLIN   = 104411       2417#
RDY     = 100000        137#
RESTRP  011242          833    1442#
RESVEC= 000010          124#
RES200  011572          972    1528#
SETVEC  012020         1584#   1594    1596    1598    1600    1602    1604    1606    1608    1610    1612    1614    1616
                       1618    1620    1622    1624    1626    1628    1630    1632    1634    1636    1638    1640    1642
                       1644    1646    1648    1650    1652    1654    1656    1658    1660    1662    1664    1666    1668
                       1670    1672    1674    1676    1678    1680    1682    1684    1686    1688    1690    1692    1694
                       1696    1698    1700    1702    1704    1706    1708    1710    1712    1714    1716    1718    1720
SIMR    = 000060        147#    551     670     775     898
SIRR    = 000120        152#    563     679    1027    1038    1157    1167    1183    1197
SSIMR   = 000070        148#
SSIRR   = 000130        153#    750     873
STACK   = 001100         28#
START   002476          183    437#
START1  003066          507    509#   1336    2365
STKLMT= 177774           39#
STRVEC  011110         1010    1140   1400#
SWR     002140          250#    442     463*    465     471*    478*   1436*   1444    1469    1543    2046    2058    2062
```

```
                        2141    2155    2157    2163    2170    2199    2236*   2340    2353*
SWRCK   014244          2048    2074#
SWREG   000176           181#    471    2199    2212
SWRSAV  002460           429#   1436    1543*
SW0     = 000001          92#
SW00    = 000001          82#     92
SW01    = 000002          81#     91
SW02    = 000004          80#     90
SW03    = 000010          79#     89
SW04    = 000020          78#     88
SW05    = 000040          77#     87
SW06    = 000100          76#     86
SW07    = 000200          75#     85
SW08    = 000400          74#     84
SW09    = 001000          73#     83
SW1     = 000002          91#
SW10    = 002000          72#
SW11    = 004000          71#
SW12    = 010000          70#   1444    1469
SW13    = 020000          69#
SW14    = 040000          68#
SW15    = 100000          67#
SW2     = 000004          90#
SW3     = 000010          89#
SW4     = 000020          88#
SW5     = 000040          87#
SW6     = 000100          86#
SW7     = 000200          85#
SW8     = 000400          84#
SW9     = 001000          83#
TBITVE= 000014           125#
TITLED  015736           502    2420#
TKVEC   = 000060         132#
TLCABL  016002           503    2426#
TOBUF   011164           497    1420#
TPVEC   = 000064         133#
TRAPVE= 000034           131#    450*    451*
TRPALL  011562          1451    1473    1523#
TRPCAT  011334           522     752    1009    1139    1464#
TRPOUT  011544          1453    1486    1515#   1533    1553
TRP200  012000          1530    1547    1577#
TRTVEC= 000014           126#
TSTNUM  002432           418#   2074*   2502    2504    2507    2509
TST1    000206           534#
TST2    004054           638     648#
TST3    004554           728     739#
TST4    005452           852     862#
TST5    006444          1004#
TST6    007362          1134#
TYPDS   = 104405        1325    2411#
TYPE    = 104401         502     503    1323    1326    1786    1939    2014    2049    2086    2103    2105    2108    2110
                        2114    2121    2210    2211    2214    2227    2238    2257    2306    2309    2313    2362    2407#
TYPOC   = 104402        2094    2118    2213    2408#
TYPON   = 104404        2410#
TYPOS   = 104403        2409#
VBUF    017116          1014    1117    1144    1277    1411    2525#
```

```
VECFIL  010766          1015  1145  1376#
VECLOC  002452           426#  754*  762   834   875*  883   974   1350*  1446*  1447  1449*  1456*  1536*
                        2504  2507  2509
VECPAT  002454           427#  763*  764*  765*  771   885*  886*  887*   894   1379*  1384  1385*
VECVAL  002456           428#  746*  770   841*  847   869*  893   980*   986   1012*  1142*  1349*  1380*
                        1382  1390*  2504  2507  2509
XXDP    002440          421#  488*  500
$APTHD  002000          210   216#
$ASTAT= ****** U        1856  1871
$ATYC   013174          1827  1829#
$ATY1   013150          1825#
$ATY3   013156          1771  1826#
$ATY4   013166          1828#  2054
$AUTOB  002134          247#  492*  498   506   2207  2328
$BASE   002244          310#  512
$BDADR  002122          242#  550*  578*  584*  619*  669*  696*  774*   799*  897*  928*  1021*  1054*
                        1061*  1067*  1093*  1119*  1151*  1202*  1209*  1215*  1238*  1263*  1270*  1279*  2502
                        2504  2507  2509
$BDDAT  002126          244#  572*  573   579*  580*  581   585*  589*   590   595*  596   601*  602
                         607*  608   614*  615   620*  621*  622   627*   628*  629   691*  692   697*
                         700*  701   706*  707   712*  713   718*  719    793*  795   800*  804*  805
                         810*  811   816*  817   822*  823   829*  830    920*  922   929*  933*  934
                         941*  942   949*  950   957*  958   966*  967   1048*  1050  1055*  1056*  1057
                        1062*  1063*  1064  1068*  1071*  1072  1076*  1077  1081*  1082  1087*  1088  1092*
                        1095*  1096  1100*  1101  1106*  1107  1112*  1113  1120*  1121  1177*  1179  1191*
                        1193  1203*  1204*  1205  1210*  1211*  1212  1216*  1219*  1220  1224*  1225  1229*
                        1230  1234*  1235  1240*  1241  1245*  1246  1251*  1252  1256*  1257  1264*  1265*
                        1266  1271*  1272*  1273  1280*  1281  1353*  1497*  1498  1507*  1508  1567*  1568
                        2502  2504
$CDW1   002250           312#
$CHARC  013144          1788*  1798*  1805  1814*  1819#
$CKSWR  014676          2199#  2415
$CMTAG  002100           230#   439   440   448   454   455
$CM3  = 000000           260#
$CNTLG  015423          2210  2323#
$CNTLU  015416          2227  2322#
$CPUOP  002216           284#
$CRLF   002165           263#  1787  1822  2049  2070  2086  2105  2110  2114  2238  2322
$DBLK   014060          1980  2014  2022#
$DEVCT  002200           275#   524*
$DEVM   002246           311#   510
$DOAGN  010624          1319  1328  1334#
$DTBL   014050          1983  2018#
$ENDAD  010614           193  1330#
$ENDCT  010562          1321#
$ENDMG  010633          1323  1338#
$ENULL  010630          1326  1337#
$ENV    002210           280#   490   525  1442  1466  1528  1544  1766  1834  1858  2051
$ENVM   002211           281#   476  1768  1773  1836
$EOP    010526          1296  1311#
$EOPCT  010554          1318#  1322
$ERFLG  002103           233#  2039*  2070  2130  2159  2161  2167*  2189
$ERMAX  002115           239#   456*  2161  2184*  2189
$ERROR  014070           448   2036#
$ERRPC  002116           240#  2043*  2044*  2045  2070  2092  2502  2504  2507  2509
$ERRTB  002252           329#  2100
```

```
$ERRTY  014262  2075  2085#
$ERTTL  002112  237#  2042*  2070
$ESCAP  002162  261#  455*  2065  2067  2070  2183*
$ETABL  002210  279#
$ETEND  002252  222  313#
$FATAL  002172  272#  480*  1862*
$FFLG   013414  1825*  1828*  1856  1865*  1873#
$FILLC  002156  258#  1791  1822
$FILLS  002155  257#  1822
$GDADR  002120  241#  1515*  1523*  1577*  2509
$GDDAT  002124  243#  571*  573  577*  581  586*  587*  590  593*  596  599*  602  605*
                608  611*  615  618*  622  625*  629  690*  692  698*  701  704*  707
                710*  713  716*  719  791*  795  801*  802*  805  808*  811  814*  817
                820*  823  826*  830  918*  922  930*  931*  934  939*  942  947*  950
                955*  958  963*  967  1047*  1050  1053*  1057  1060*  1064  1069*  1072  1077
                1082  1085*  1088  1091*  1096  1101  1107  1111*  1113  1118*  1121  1124*  1176*
                1179  1190*  1193  1201*  1205  1208*  1212  1217*  1220  1225  1230  1235  1241
                1246  1252  1257  1262*  1266  1269*  1273  1278*  1281  1284*  1352*  1496*  1498
                1506*  1508  1566*  1568  2502  2504
$GET42  010604  1327#
$GTSWR  014746  2211#  2413
$HD   = 000003  11  12
$HIBTS  002000  217#
$ICNT   002104  234#  2174*  2175  2177*  2188
$ILLUP  015616  2332  2348  2367#
$INTAG  002135  248#  2239  2328
$ITEMB  002114  238#  2045*  2053  2070  2089
$LF     002166  264#  1822  2070  2313  2322
$LFLG   013413  1866*  1872#
$LPADR  002106  235#  457*  2165*  2181*  2186  2188
$LPERR  002110  236#  458*  538*  652*  753*  874*  2064  2165  2182*  2188
$MADR1  002222  297#
$MADR2  002226  301#
$MADR3  002232  304#
$MADR4  002236  307#
$MAIL   002170  218  222  270#  475  1766  2051  2180
$MAMS1  002220  291#
$MAMS2  002224  299#
$MAMS3  002230  302#
$MAMS4  002234  305#
$MBADR  002002  218#
$MFLG   013412  1826*  1832  1867*  1871#
$MNEW   015441  2214  2326#
$MSGAD  002204  277#  1842*  1845
$MSGLG  002206  278#  1847*
$MSGTY  002170  271#  481*  1840  1848*  1860  1864*
$MSWR   015430  2211  2324#
$MTYP1  002221  292#
$MTYP2  002225  300#
$MTYP3  002231  303#
$MTYP4  002235  306#
$MXCNT  014674  2178  2188#
$NULL   002154  256#  1793  1822
$NWTST= 000001  531#  645#  736#  859#  1001#  1131#
$OCNT   013640  1911*  1940*  1953#
$OMODE  013642  1906*  1910*  1915  1918*  1929*  1955#
```

```
$OVER   014660          2142   2158   2166   2176   2185#
$PASS   002176           274#   475*  1315*  1316*  1324   1337   2172   2189
$PASTM  002006           220#
$PWRAD  015612          2365#
$PWRDN  015452           452   2332#  2360
$PWRMG  015606          2363#
$PWRUP  015524          2342   2348#
$QUES   002164           262#  1822   2070   2257   2306   2322
$RDCHR  015160          2270#  2416
$RDDEC= ****** U        2418
$RDLIN  015300          2298#  2417
$RDOCT= ****** U        2418
$RDSZ = 000010          2291#
$RTNAD  010626          1336#
$R2A  = ****** U        2418
$SAVRE= ****** U        2418
$SAVR6  015622          2341*  2349   2350*  2351*  2369#
$SCOPE  014416           446   2139#
$SETUP= 000117           436#   445    446    448    450    452    454    455    457   1313   2037   2061   2069
                        2140   2194   2328
$STUP = 177777           436#
$SVLAD  014624          2150   2179#
$SVPC = 000106           191#   196
$SWR  = 165400            11    12#    13#    19    20    21    22    23    24    260    261    262    454
                         455    457    458    535   649   740   863   1005  1135  1308   1314   1329   1335
                        1337   2029   2030   2031  2032  2042  2046  2058  2062  2070   2131   2132   2133
                        2134   2135   2141   2153  2155  2156  2159  2160  2161  2168   2169   2170   2182
                        2185   2188   2366
$SWREG  002212           282#   478
$SWRMK= 000000            24     25   2135   2136   2157
$TESTN  002174           273#   482*  2180*
$TIMES  002160           260#   454*  1314*  2168*  2175   2178*  2188
$TKB    002146           253#  2192   2203   2220   2274   2280
$TKS    002144           252#  2192   2201   2217   2241*  2272   2278
$TN   = 000007            11#    14#   531    535#   638    645    649#   728    736    740#   852    859    863#
                        1001   1005#  1131   1135#
$TPB    002152           255#  1811*  1822
$TPFLG  002157           259#  1760   1822
$TPS    002150           254#  1809   1822
$TRAP   015656           450   2384#
$TRAP2  015700          2395#  2406
$TRP  = 000012          2399#  2408#  2409#  2410#  2411#  2412#  2413   2414#  2415   2416#  2417#  2418#
$TRPAD  015712          2389   2406#
$TSTM   002004           219#
$TSTNM  002102           232#  1313*  2041   2070   2074   2130   2157   2179*  2180   2185   2189
$TTYIN  015406          2299   2300   2317   2321#
$TYPBN= ****** U        2412
$TYPDS  013644          1968#  2411
$TYPE   012666          1760#  1853   2399   2407
$TYPEC  013100          1790   1797   1804   1809#  1810   2243
$TYPEX  013146          1815   1817   1820#
$TYPOC  013442          1909#  2408
$TYPON  013456          1908   1911#  2410
$TYPOS  013416          1904#  2409
$UNIT   002202           276#   509*   524   1298*
$UNITM  002010           221#
```

F 6



```
$USWR   002214          283#
$VECT1  002240          308#
$VECT2  002242          309#
$XTSTR  014430          2144#
$$GET4= 000000          1329#
$DFILL  013641          1905*   1909*   1919    1954#
$40CAT= ****** U        2048    2141
        017316          175#    179#    184#    191     192#    194#    196#    197#    206     207#    209#    211#    229#
                        265     443     457     458     1337    1341    1822    1874#   2022#   2070    2124#   2188    2189
                        2192    2321#   2322    2328    2344    2368    2519#   2525#
.$ASTA= ****** U        1826    1829
.$X   = 002000          206#    211
```

```
COMMEN     1#    135#
ENDCOM     1#    135#
ERROR     29#    561    567    575    583    592    598    604    610    617    624    631    677    683    694
          703    709    715    721    781    787    797    807    813    819    825    832    905    913    925
          937    945    953    961    970   1032   1043   1052   1059   1066   1074   1079   1084   1090   1098
         1103   1109   1115   1123   1162   1172   1181   1195   1207   1214   1222   1227   1232   1237   1243
         1248   1254   1259   1268   1275   1283   1500   1510   1517   1524   1571   1579   1589
ESCAPE     1#    135#
GETPRI     1#    135#
GETSWR     1#    135#
MULT       1#    135#
NEWTST     1#    135#   531    645    736    859   1001   1131
POP        1#    135#  1868   1869   2009   2353   2354
PUSH       1#    135#  1829   1831   1852   1968   2334   2340
REPORT     1#    135#
SCOPE     30#    534    648    739    862   1004   1134
SETPRI     1#    135#
SETTRA  2399#   2408   2409   2410   2411   2413   2415   2416   2417
SETUP      1#    135#   437
SKIP       1#    135#   638    728    852
SLASH      1#    135#
SPACE    135#
STARS      1#    135#   189    203    205    212    225    265    268    531    533    645    647    736    738
          859    861   1001   1003   1131   1133   1305   1344   1346   1745   1824   1881   1958   2025   2070
         2073   2080   2127   2191   2194   2262   2291   2330   2346   2378   2514   2518   2521   2524
SWRSU      1#    135#   459#
TRMTRP  2399#
TYPBIN     1#    135#
TYPDEC     1#    135#  1324
TYPNAM     1#    135#
TYPNUM     1#    135#
TYPOCS     1#    135#
TYPOCT     1#    135#  2092   2116   2212
TYPTXT     1#    135#
$$CMRE   223#
$$CMTM   223#
$$ESCA     1#    135#
$$NEWT     1#    135#   531    645    736    859   1001   1131
$$SET   2399#   2408   2409   2410   2411   2413   2415   2416   2417
$$SETM   475#
$$SKIP     1#    135#   638    728    852
.EQUAT     1#     25
.HEADE     1#
.KT11      1#
.SETUP     1#    436
.SWRHI     1#     15
.SWRLO    25#
.$ACT1     1#    187
.$APTB     1#    266#
.$APTH     1#    201
.$APTY     1#   1822
.$ASTA     1#
.$CATC     1#    173
.$CMTA     1#    223
.$DB2D     1#
.$DB20     1#
```

```
.$DIV       1#
.$EOP       1#   1303
.$ERRO      1#   2023
.$ERRT      1#   2078
.$MULT      1#
.$POWE      1#   2328
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#   2189
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#   2125
.$SIZE      1#
.$SUPR      1#
.$TRAP      1#   2376
.$TYPB      1#
.$TYPD      1#   1956
.$TYPE      1#   1743
.$TYPO      1#   1879
.$40CA      1#
.1170       1#

. ABS.  017316     000

ERRORS DETECTED:  0

DSKZ:CVDRDA,DSKZ:CVDRDA/CRF/SOL=DSKZ:SYSMAC.SML,DSKZ:CVDRDA.P11
RUN-TIME: 41 41 3 SECONDS
RUN-TIME RATIO: 145/86=1.6
CORE USED:  33K  (65 PAGES)
```