LPA/LPS

DIAGNOSTIC TEST 3

MD-11-DRLPJ-A

EP-DRLPJ-A-DL

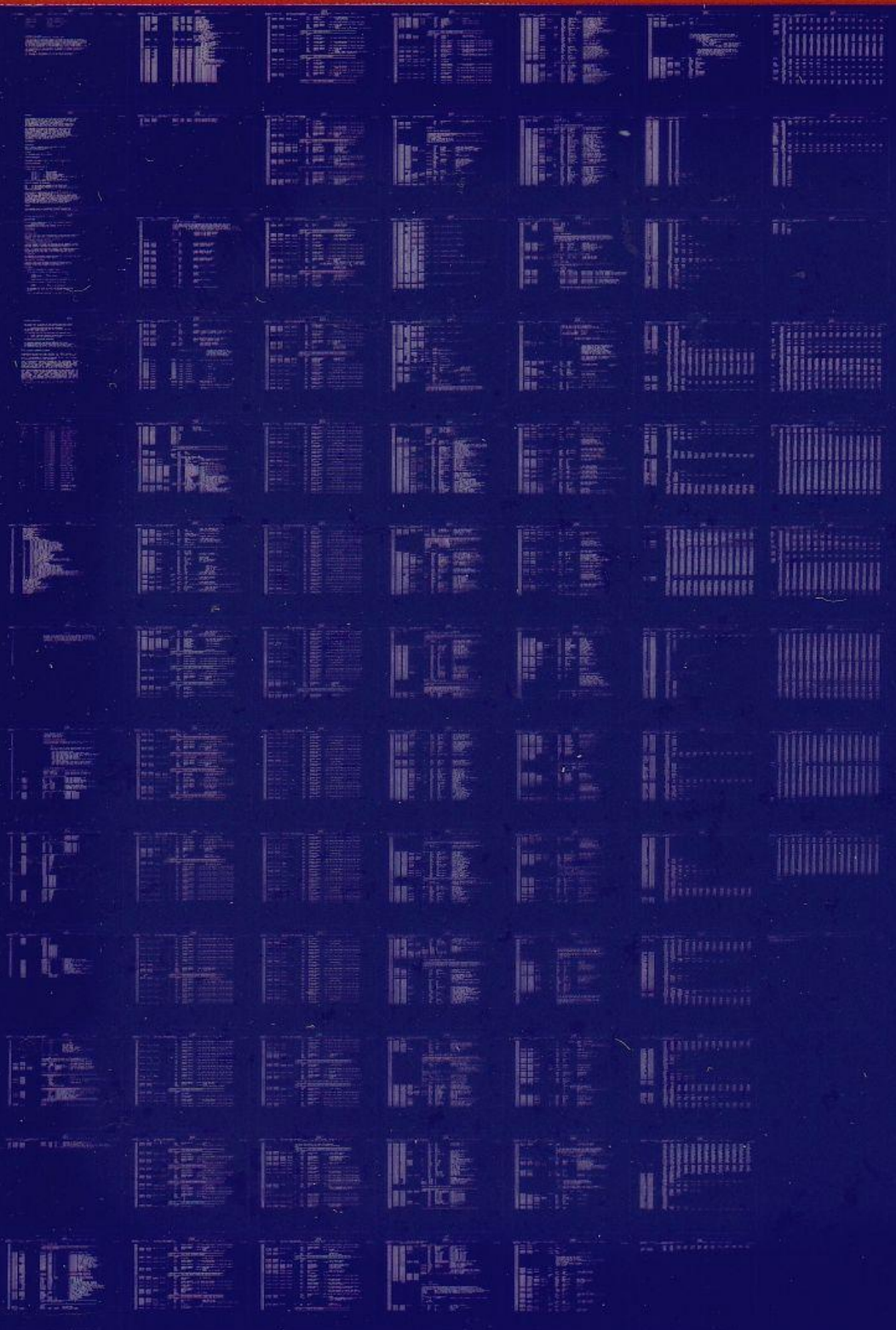
COPYRIGHT © 1978

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA



EOF1DRLPBASBQ411 00010000 780223 IDENTIFICATIBOP10 411 DEHDR1DRLPJASEQ 00010000 780223
SEQ 0001

PRODUCT CODE: MAINDEC-11-DRLPJ-A-D

PRODUCT NAME: LPA/LPS DIAGNOSTIC TEST 3

DATE: JANUARY 1976

MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1978
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE IN EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

1. ABSTRACT

SEQ 0002

THIS PROGRAM IS A LOGIC TEST OF THE "LPS11-DRA" DIGITAL INPUT OUTPUT CONTROL OPTION. ALL FUNCTIONS OF THE OPTION WILL BE TESTED. DUE TO THE FLEXIBILITY OF THE OPTION, THE OPERATOR MAY BE REQUIRED TO sUPPLY OPTION CHARASTICS.
THE PROGRAM WILL HANDLE ALL CONFIGURATIONS OF SW15 THRU SW00 AND JUMPERS W15 THRU W00. THE FOLLOWING JUMPERS MUST BE INSERTED TO EXECUTE THE LOGIC TESt: W16-W17-W18-W19.

THIS PROGRAM IS A MODIFIED VERSION OF "MD-11-DZLPI-B". IT WAS MODIFIED TO ENABLE THE OPERATOR TO CHECK OUT THE LPS11 OPTION WHEN IT IS ON THE LPA11-KX I/O BUS. NO RECABLING IS NEEDED. SOME TEST DONE IN THE ORIGINAL DIAGNOSTIC SUCH AS ARBITRATION TESt, WERE DELETED AS THEY COULD NOT BE CHECKED. IF THIS DIAGNOSTIC DOESN'T FIND A SUSPECTED PROBLEM, YOU MAY HAVE TO RUN "MD-11-DZLPI-B" YOU SHOULD RUN "MD-11-DRLPA" BEFORE RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 11.

2. REQUIREMENTS

2.1 EQUIPTMENT

PDP-11 FAMILY COMPUTER WITH 16K WORDS OF MEMORY
LPS11 WITH DRA OPTION INSTALLED
TELETYPE (OR EQUIVALENT)

2.2 STORAGE

THIS PROGRAM USES 16K OF MEMORY.

3. LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

```
SW 15 = 1       HALT ON ERROR
SW 14 = 1       LOOP ON TEST
SW 13 = 1       INHIBIT ERROR TYPOUTS
SW 11 = 1       INHIBIT INTERATIONS
SW 10 = 1       DRA INPUT-OUTPUT CABLE NOT CONNECTED
SW 09 = 1       LOOP ON ERROR
SW 08 = 1       LOOP ON TESt IN SWR <7:0>
```

REFER TO 10. FOR SOFTWARE SWITCH REGISTER OPERATION.

4.2 STARTING ADDRESS OR ADDRESSES

```
200     IS THE STARTING ADDRESS OF THE LOGIC TEST.
204     IS THE RESTART ADDRESS OF THE LOGIC TEST.
210     IS THE STARTING ADDRESS OF THE COMBINED FUNCTION LOOP.
```

5. OPERATING PROCEDURE

SEQ 0003

THE FOLLOWING JUMPERS MUST BE INSTALLED TO EXECUTE THE LOGIC TEST: W16-W17-W18-W19.
IF THE CUSTOMER HAS SELECTED THE "A" SECTION OF THESE JUMPER IT MUST BE RETURNED TO THE "FACTORY" POSITION BEFORE RUNNING THE LOGIC TEST. ** WORSE CASE WILL ONLY BE CHANGING FOUR JUMPERS. **
THE OPERATOR MUST INSERT THE CORRECT INFORMATION IN THE SWITCH REGISTER WHEN REQUESTED BY THE PROGRAM OR AN ERROR WILL OCCUR.
ONCE STARTED THE TEST WILL RUN IN IT'S NORMAL MANNER WITHOUT OPERATOR INTERVENTION OR SWITCH SELECTION.

6. ERRORS

THIS PROGRAM USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR ERROR REPOTRING AND TYPEOUT. REFER TO tHE "ERROR POINTER TABLE"

ERROR REPOTRING AND TYPEOUT. REFER TO tHE "ERROR POINTER TABLE"

FOR TYPE AND DEScRIPTION OF ERRORS.

7. RESTRICTIONS

THE FOLLOWING JUMPERS MUST BE IN THE "FACTORY" POSITION:
W16-W17-W18-W19.
THE OPERATOR MUST SUPPLY THE CORRECT JUMPER AND SWITCH CONFIGURATION OR AN ERROR WILL OCCUR.

8. MISCELANEOUS

8.1 EXECUTION TIME

THE LOGIC TESt WILL TAKE APPROXIMATELY 5 SECONDS FOR COMPLETION AND WILL TYPE 'END PASS'. FURTHER PASSES TAKE APPROXIMATELY 1 MIN.
THE COMBINED FUNCTION LOOP WILL NEVER EXIT.

8.2 DEVICE ADDRESS PROGRAM LOCATIONS

LOCATION "$BASE" CONTAINS THE DRA BASE DEViCE ADDRESS (170410)
LOCATION "$VECTi" CONTAINS THE DRA BASE INTERRUPT VECTOR (350)
LOCATION "$PRIOR" CONTAINS THE DRA BR LEVEL (200)(4)

*NOTE: IF THESE LOCATIONS ARE CHANGED, THE OPERATOR MUST START THE TEST AGAIN AT LOC. 200. THE PROGRAM WILL USE THE BASE ADDRESS AND VECTOR AND UPDATE THE ACTUAL PROGRAM VALUES.

8.3 ACT/XXDP/APT

THE PROGRAM IS CHAINABLE UNDER XXDP AND ACT. THE HOOKS FOR "APT" ARE PROVIDED BUT HAVE NOT BEEN TESTED.

8.4 USER LINK TO I/O DEVICE

A SPECIAL USER LINK HAS BEEN PROVIDED IN ORDER FOR THE OPERATOR TO EXAMINE OR MODIFY LOCATIONS ON THE LPA11-KX I/O BUS. (NOTE: THIS CANNOT BE DONE DIRECTLY.)

PROCEDURE:

1) START THE PROCESSOR AT LOCATION $UTK:

SEQ 0004

2) THE DIALOG TO EXAMINE A LOCATION IS AS FOLLOWS:

```
E OR D              "E"
DEVICE ADDRS=       "OCTAL ADDRS"
XXXXXX
```

WHERE XXXXXX IS THE CONTENTS OF THE SPECIFIED LOC.

3) THE DIALOG TO MODIFY A LOCATION IS AS FOLLOWS:

```
E OR D              "D"
DATA=               "DATA TO BE DEPOSITED"
```

4) THE PROGRAM WILL STAY IN THIS LOOP UNTIL THE OPERATOR IS FINISHED. AT THIS TIME THE PROCESSOR SHOULD BE HALTED.

NOTE: THE OPERATORS RESPONSE IS ENCLOSED IN QUOTES.

9. PROGRAM DESCRIPTION

THE LOGIC TEST IS A TEST OF THE CONTROL AND INPUT/OUTPUT REGISTERS. ALL JUMPERS AND SWITCHES COMBINATIONS EXCEPT:

W16A-W17A-W18A-W19A CAN BE DIAGNOSED.

THE PROGRAM CHECKS THAT THE DRA CAN INTERRUPT AND THAT "RESET" WILL WORK CORRECTLY.

THE COMBINED FUNCTION LOOP PROVIDES THE OPERATOR WITH:

1. SCOPE LOOP FOR INVERTED SIGNALS (W16A-W17A-W18A-W19A).
2. SLOW LOOP FOR TESTING RELAY CLOSURE.

10. SOFTWARE SWITCH REGISTER OPERATION

THE PROGRAM SUPPORTS NON-SWITCH REGISTER CPU TYPES.
A CHANGE IN SWR VALUE IS ACCOMPLISHED BY TYPING A "CTRL G".
THE RESPONSE WILL BE "SWR = " AND WAIT FOR A NEW VALUE.
THE OPERATOR NOW INPUTS THE NEW VALUE AND TERMINATES WITH A "CR".

11. LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

DIAGNOSTICS FOR THE LPA11 ARE WRITTEN AT THREE LEVELS: (1) TOTAL PDP-11 SYSTEM, (2) LPA11 SYSTEM; AND, (3) LPA11 OPTIONS.

LEVEL 1, IS DESIGNED TO ISOLATE A FAILURE TO THE LPA11 SYSTEM. ALL OPTIONS ON THE PDP-11 ARE EXERCISED.

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN THE USER RUNS DRLPA HE CAN GENERALLY TELl WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY "LOOK" ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

LEVEL THREE DIAGNOSTICS AID IN DETERMINING IF THE ERROR WAS IN FACT ON THE OPTION THE DRLPA SPECIFIED. THE USER MAY "LOOP" ON THE ERROR. WITHIN LEVEL THREE, THERE ARE TWO GROUPS OF DIAGNOSTICS. THE FIRST GROUP REQUIRES NO "EXTRA" WORK BY THE USER IN ORDER TO RUN. GROUP "A" DIAGNOSTICS DO NOT CHECK ARBITRATION, AND REQUIRE EXTRA TIME FOR EXECUTION. THE SECOND GROUP (GROUP "B") REQUIRES THAT THE USER RECONFIGURE THE PDP-11 SYSTEM. THIS RECONFIGURATION INVOLVES CABLING THE UNIBUS TO THE LPA'S I/O BUS.

THE DIAGNOSTIC FOR THE M8254 FALLS INTO THE GROUP "B" CATAGORY.

F01

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:

SEQ 0006

OPTION	GROUP	DIAG. #	DIAG. TITLE
LPA11-KX	LEVEL 2	MD-11-DRLPA	LPA11-K SYSTEM DIAG.
M8254	"B"	MD-11-DRLPN	M8254 (IPBM) DIAG.
AA11-K	A	MD-11-DRLPB	AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11	A	MD-11-DRLPC	LPA/AR11 DIAG. #1
	A	MD-11-DRLPD	LPA/AR11 DIAG. #2
	A	MD-11-DRLPE	LPA/AR11 DIAG. #3
	B	MD-11-DZARA	AR11 DIAG. #1
	B	MD-11-DZARB	AR11 DIAG. #2
	B	MD-11-DZARC	AR11 DIAG. #3
DR11-K	A	MD-11-DRLPF	LPA/DR11-K DIAG.
	B	MD-11-DZDRG	DR11-K DIAG.
KW11-K	A	MD-11-DRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS11	A	MD-11-DRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-DRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-DRLPJ	LPA/LPS11 DIAG. #3
	B	MD-11-DZLPC	LPS11 DIAG. #1
	B	MD-11-DZLPD	LPS11 DIAG. #2
	B	MD-11-DZLPI	LPS11 DIAG. #3
AD11-K	A	MD-11-DRLPK	LPA/AD11-K DIAG.
	B	MD-11-DZADL	AD11-K DIAG.
M8200-YC	B	MD-11-DZLPL	LPA/M8200-YC BASIC MICRO-CPU R/W TEST
	B	MD-11-DZLPM	LPA/M8200-YC JMP+ROM READ TEST

```
                        .REM    [

                                LPA.MAC

WELCOME, THIS DIAGNOSTIC IS ONE IN A SERIES OF DIAGNOSTIC
DESIGNED IN ORDER TO AID YOU IN TESTING THE LPA-11XX OPTION.
I HOPE THAT YOU HAVE READ THE DOCUMENTATION SECTION OF THIS
DIAGNOSTIC.  IF YOU HAVE,YOU KNOW ABOUT ALL OF THE DIAGNOSTICS
THAT ARE AVAILIBLE FOR TESTING THE LPA SYSTEM.

                                GOOD LUCK !

[
.GLOBL  DRLPX2
```

```
                        .TITLE  MAINDEC-11-DRLPJ-A      LPA/LPS-11-DRA DIAGNOSTIC
                        ;*COPYRIGHT (C) 1978
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY EDWARD BADGER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                                .REM    !

                                THIS IS A LIST OF TESTS DELETED FROM THIS DIAGNOSTIC.
                                THESE TEST COULD NOT BE DONE THROUGH THE LPA-11

                                  TEST FOR UNEXPECTED INTER.
                                  TEST THAT THE INPUT CAN INT USNG MAINT. BIT
                                  TEST THAT THE INPUT INT. CLEARS INT. ENABLE VIA MANT BIT
                                  TEST THAT THE OUTPUT CAN INTER. USING THE MAINT. BIT.
                                  TEST FOR INTER. FORM DRA INPUTS
                                  TEST FOR INTER. FROM DRA OUTPUTS
                                  TEST FOR INTER. FROM DRA INPUTS ON LEVEL INDICATED -1
                                    VIA MAINT. INIT.
                                  TEST THAT OUTPUT REG. CAN HOLD A COUNT PATTERN
                                  TEST THAT OUTPUT CAN HOLD BYTE COUNT PATTERN
                                  TEST THAT OUTPUT CAN HOLD HIGH BYTE COUNT PATTERN

                                !
                        .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
```

```
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  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)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
```

```
                        
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  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
170410                          ABASE=170410
000350                          AVECT1=350
000200                          APRIOR=200
```

```
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------                  ---
                                ;*        15                    HALT ON ERROR
                                ;*        14                    LOOP ON TEST
                                ;*        13                    INHIBIT ERROR TYPEOUTS
                                ;*        11                    INHIBIT ITERATIONS
                                ;*        10                    DRA INPUT OUTPUT CABLE NOT CONNECTED
                                ;*         9                    LOOP ON ERROR
                                ;*         8                    LOOP ON TEST IN SWR<7:0>

                                .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  001564                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001570                  JMP     @#BEGIN1        ;JUMP TO THE RESTART ADDRESS
000210  000137  011764                  JMP     @#EXTTST        ;JUMP TO THE MISC. TEST

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  012324                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000                 $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
```

```
225  001002  001176                $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
226  001004  000012                $TSTM:  .WORD   10.      ;;RUN TIM OF LONGEST TEST
227  001006  000074                $PASTM: .WORD   60.      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
228  001010  000074                $UNITM: .WORD   60.      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
229  001012  000052                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                  000000  $CMTAG:         .WORD   0               ;;START OF COMMON TAGS
001102                     000  $TSTNM: .BYTE    0               ;;CONTAINS THE TEST NUMBER
001103                     000  $ERFLG: .BYTE    0               ;;CONTAINS ERROR FLAG
001104                  000000  $ICNT:  .WORD    0               ;;CONTAINS SUBTEST ITERATION COUNT
001106                  000000  $LPADR: .WORD    0               ;;CONTAINS SCOPE LOOP ADDRESS
001110                  000000  $LPERR: .WORD    0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112                  000000  $ERTTL: .WORD    0               ;;CONTAINS TOTAL ERRORS DETECTED
001114                     000  $ITEMB: .BYTE    0               ;;CONTAINS ITEM CONTROL BYTE
001115                     001  $ERMAX: .BYTE    1               ;;CONTAINS MAX. ERRORS PER TEST
001116                  000000  $ERRPC: .WORD    0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120                  000000  $GDADR: .WORD    0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122                  000000  $BDADR: .WORD    0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124                  000000  $GDDAT: .WORD    0               ;;CONTAINS 'GOOD' DATA
001126                  000000  $BDDAT: .WORD    0               ;;CONTAINS 'BAD' DATA
001130                  000000          .WORD    0               ;;RESERVED--NOT TO BE USED
001132                  000000          .WORD    0
001134                     000  $AUTOB: .BYTE    0               ;;AUTOMATIC MODE INDICATOR
001135                     000  $INTAG: .BYTE    0               ;;INTERRUPT MODE INDICATOR
001136                  000000          .WORD    0
001140                  177570  SWR:    .WORD    DSWR            ;;ADDRESS OF SWITCH REGISTER
001142                  177570  DISPLAY: .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144                  177560  $TKS:   177560                  ;;TTY KBD STATUS
001146                  177562  $TKB:   177562                  ;;TTY KBD BUFFER
001150                  177564  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152                  177566  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154                     000  $NULL:  .BYTE    0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155                     002  $FILLS: .BYTE    2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156                     012  $FILLC: .BYTE    12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157                     000  $TPFLG: .BYTE    0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160                  000000  $REGAD: .WORD    0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REGO) WAS OBTAINED
001162                  000000  $REGO:  .WORD    0               ;;CONTAINS (($REGAD)+0)
001164                  000000  $REG1:  .WORD    0               ;;CONTAINS (($REGAD)+2)
001166                  000000  $TIMES: 0                        ;;MAX. NUMBER OF ITERATIONS
001170                  000000  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001172                     077  $QUES:  .ASCII   /?/             ;;QUESTION MARK
001173                     015  $CRLF:  .ASCII   <15>            ;;CARRIAGE RETURN
001174                  000012  $LF:    .ASCIZ   <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001176                          $MAIL:                          ;;APT MAILBOX
001176                  000000  $MSGTY: .WORD    AMSGTY         ;;MESSAGE TYPE CODE
001200                  000000  $FATAL: .WORD    AFATAL         ;;FATAL ERROR NUMBER
```

```
001202  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001204  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
001206  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001210  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001212  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001214  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001216                  $ETABLE:                ;;APT ENVIRONMENT TABLE
001216     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001217     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001220  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001222  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001224  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                        ;*                      BITS 15-11=CPU TYPE
                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                              11/70=06,PDQ=07,Q=10
                        ;*                      BIT 10=REAL TIME CLOCK
                        ;*                      BIT  9=FLOATING POINT PROCESSOR
                        ;*                      BIT  8=MEMORY MANAGEMENT
001226     000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001227     000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                        ;*                      MEM.TYPE BYTE  --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              500 NSEC MOS=003
001230  000000          $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                        ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001232     000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001233     000          $MTYP2: .BYTE   AMTYP2  ;;MEM. TYPE,BLK#2
001234  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001236     000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001237     000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001240  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001242     000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001243     000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001244  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001246  000350          $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001250  000000          $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001252  170410          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001254  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001256  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001260  000000          $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001262  000000          $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001264  000000          $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001266  000000          $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001270  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001272  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORd#4
001274  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORd#5
001276  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001300  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001302  000000          $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001304  000000          $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001306  000000          $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001310  000000          $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001312  000000          $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
```

```
  338  001314  000000                    $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
  339  001316  000000                    $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
  340  001320  000000                    $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
  341
  342
  343  001322                          $ETEND:
  344
```

```
                                .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

001322                          $ERRTB:

                                ;ITEM   1
001322  012376                          EM1             ;STATUS REGISTER IN ERROR
001324  013016                          DH1             ;ERRPC  STATUS EXPECTED
001326  013202                          DT1             ;$ERRPC $BDDAT $GDDAT
001330  013230                          DF1

                                ;ITEM   2
001332  012427                          EM2             ;INPUT REGISTER IN ERROR
001334  013047                          DH2             ;ERRPC INPUT EXPECTED
001336  013202                          DT1             ;$ERRPC $BDDAT $GDDAT
001340  013230                          DF1

                                ;ITEM 3
001342  012457                          EM3             ;OUTPUT REGISTER IN ERROR
001344  013100                          DH3             ;ERRPC OUTPUT EXPECTED
001346  013202                          DT1             ;$ERRPC $BDDAT $GDDAT
001350  013230                          DF1

                                ;ITEM   4
001352  012510                          EM4             ;INPUT FAILED TO INTERRUPT
001354  013131                          DH4             ;ERRPC
001356  013212                          DT4             ;$ERRPC
001360  013230                          DF1

                                ;ITEM   5
001362  012542                          EM5             ;OUTPUT FAILED TO INTERRUPT
001364  013131                          DH4             ;ERRPC
001366  013212                          DT4             ;$ERRPC
001370  013230                          DF1

                                ;ITEM   6
001372  012575                          EM6             ;UNEXPECTED INTERRUPT
001374  013131                          DH4             ;ERRPC
001376  013212                          DT4             ;$ERRPC
001400  013230                          DF1

                                ;ITEM   7
001402  012622                          EM7             ;OPERATOR INTERVENTION ERROR
```

```
                                                DH4             ;ERRPC
                                                DT4             ;$ERRPC
                                                DF1

                                    ;ITEM       10
                                                EM10            ;INTERRUPT INPUT BIT FAILED TO SET INPUT READY
                                                DH10            ;ERRPC    STATUS    EXPECTED  INPUT BIT
                                                DT10            ;$ERRPC  $BDDAT  $GDDAT  BRLEV3
                                                DF10

                                    ;ITEM       11
                                                EM11            ;NON-INTERRUPTING INPUT BIT SET INPUT READY
                                                DH10            ;ERRPC  STATUS  EXPECTED  INPUT BIT
                                                DT10            ;$ERRPC  $BDDAT  $GDDAT  BRLEV3

                                    DRADD:      170410                  ;DRA STARTING ADDRESS
                                    DRIV:       350                     ;DRA STARTING INTERRUPT VECTOR
                                    DRBRL:      200                     ;DRA BR LEVEL


                                                ;
                                                ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO MAY BE
                                                ;                               CHANGED BY THE USER TO rEFLECT
                                                ;                               A DIFFERENT KMC-11 ADDR. THE
                                                ;                               REST OF THE ADDRESSES WILL
                                                ;                               BE CHANGED BY THE PROGRAM.
                                                ;

001436                              LPCI:
001436  170460                      KMADO:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.

001440                              LPMR:
001440  170461                      KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
001442                              LPCO:
001442  170462                      KMAD2:  .WORD   170460+2                ;>PATCH         <;
001444                              LPSO:
001444  170463                      KMAD3:  .WORD   170460+3                ;>THIS AREA     <
001446                              LPADL:
001446  170464                      KMAD4:  .WORD   170460+4                ;
001450                              LPADH:
001450  170465                      KMAD5:  .WORD   170460+5                ;>DO NOT        <
001452                              LPMS1:
001452  170466                      KMAD6:  .WORD   170460+6                ;>PATCH         <
001454                              LPMS2:
001454  170467                      KMAD7:  .WORD   170460+7                ;>THIS AREA     <

001456  000350                      VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
001460  000354                      VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2

001462  000004                      VERSN:  .WORD   4               ;CURRENT VERSION NUMBER OF MICROCODE.

001464  000000                      .DVLs:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
001466  000020                              .BLKW   16.             ;/BY INIT.
```

```
                                GRSTAT: 170410          ;DR STATUS
                                GRDAI:  170412          ;INPUT REG.
                                GRDIO:  170414          ;OUTPUT REG.
                                GRBHIO: 170415          ;OUTPUT REG HIGH BYTE
                                NOTLCH: 0
                                NOTINT: 0
                                GRIVA:  310
                                GRIVSA: 312
                                GRIVB:  314
                                GRIVSB: 316
                                DIOBRL: 200
                                BRLEV1: 0
                                BRLEV2: 0
                                BRLEV3: 0
                                $TMDAT: 0

                                ;;*****************************************************************
                                        ;DIGITAL I-O LOGIC TEST
                                ;;*****************************************************************
001564                          BEGIN:
001564  005000                          CLR     R0
001566  000402                          BR      RBEG                    ;
001570  012700  177777          BEGIN1: MOV     #-1,R0
001574  000240                  RBEG:   NOP
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001576  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001602  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001604  022706  001140                  CMP     #$WR,R6 ;;DONE?
001610  001374                          BNE     .-6             ;;LOOP BACK IF NO
001612  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001616  012737  013332  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001624  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001632  012737  013614  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001640  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001646  012737  016210  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001654  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001662  012737  015240  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001670  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001676  013737  012304  012276          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001704  005037  001166                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001710  005037  001170                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001714  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001722  012737  001722  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001730  012737  001730  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001736  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001742  012737  001776  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001750  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001756  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001764  022777  177777  177146          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001772  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
```

```
                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001774  000403                          BR      65$     ;;BRANCH IF NO TIMEOUT
001776  012716  002004          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002002  000002                          RTI
002004  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002012  012737  000174  001142          MOV     #DISPREG,DISPLAY
002020  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002024  005037  001204                  CLR     $PASS           ;;CLEAR PASS COUNT
002030  132737  000200  001217          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002036  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002040  012737  001220  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002046                          67$:
002046  013737  001252  001430  DONOT:  MOV     $BASE,DRADD

                                ;
                                ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
                                ;

002054  010046                          MOV     R0,-(SP)
002056  010146                          MOV     R1,-(SP)
002060  013700  001436                  MOV     KMADD,R0        ;GET KMC-11 ADDRESS.
002064  012701  001440                  MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.

002070  005200                  64$:    INC     R0              ;UPDATE ADDR.
002072  010021                          MOV     R0,(1)+         ;WRITE ADDR.
002074  020127  001456                  CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
002100  001373                          BNE     64$             ;NO - DO NEXT ADDR.
002102  005037  001464                  CLR     .DVLS           ;CLR ADDR. LIST.
002106  012601                          MOV     (SP)+,R1
002110  012600                          MOV     (SP)+,R0
002112  005700                          TST     R0                      ;TEST R0
002114  001402                          BEQ     2$                      ;BR IF CLEARED
002116  000137  002610          1$:     JMP     3$                      ;JUMP IF SET
002122  005737  000042          2$:     TST     @#42                    ;TEST IF IN "CHAIN MODE"
002126  001373                          BNE     1$                      ;BR IF YES
002130  104401  002136                  TYPE    ,66$            ;;TYPE ASCIZ STRING
002134  000432                          BR      65$             ;;GET OVER THE ASCIZ
                                ;;66$:  .ASCIZ  <15><12><15><12>#LPA/LPS-11-DRA DIGITAL INPUT OUTPUT LOGIC TEST#
002222                          65$:
002222  104401  002230                  TYPE    ,68$            ;;TYPE ASCIZ STRING
002226  000413                          BR      67$             ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <15><12>/MAINDEC-11-DRLPJ-A/
002256                          67$:
002256  104401  002264                  TYPE    ,70$            ;;TYPE ASCIZ STRING
002262  000435                          BR      69$             ;;GET OVER THE ASCIZ
                                ;;70$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-LATCHING BITS/
002356                          69$:
002356  004737  013254                  JSR     PC,GETSWR               ;GET THE SWITCH REGISTER VALUE
002362  017737  176552  001536          MOV     @SWR,NOTLCH             ;SAVE SWITCHES
002370  104401  002376                  TYPE    ,72$            ;;TYPE ASCIZ STRING
002374  000437                          BR      71$             ;;GET OVER THE ASCIZ
                                ;;72$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-INTERRUPTING BITS/
002474                          71$:
```

```
561  002474  004737  013254                JSR     PC,GETSWR               ;GET THE SWITCH REGISTER VALUE
562  002500  017737  176434  001540        MOV     @SWR,NOTINT             ;SAVE SWITCHES
563  002506  104401  002514                TYPE    ,74$                    ;;TYPE ASCIZ STRING
564  002512  000434                        BR      73$                     ;;GET OVER THE ASCIZ
565                                  ;;74$:  .ASCIZ  <15><12>/SET SWITCH REGISTER WITH THE DESIRED PROGRAM OPTIONS/
566  002604                          73$:
567  002604  004737  013254                JSR     PC,GETSWR               ;GET THE SWITCH REGISTER VALUE
568  002610  013737  001430  001526  3$:   MOV     DRADD,GRSTAT            ;LOAD INITIAL ADDRESS
569  002616  013737  001430  001530        MOV     DRADD,GRDAI             ;LOAD 2ND ADDRESS
570  002624  062737  000002  001530        ADD     #2,GRDAI
571  002632  013737  001430  001532        MOV     DRADD,GRDIO             ;LOAD 3RD ADDRESS
572  002640  062737  000004  001532        ADD     #4,GRDIO
573  002646  013737  001430  001534        MOV     DRADD,GRBHIO            ;LOAD 4TH ADDRESS
574  002654  062737  000005  001534        ADD     #5,GRBHIO
575  002662  000240                IOTEST: NOP     ;
576  002664  012706  001100                MOV     #STACK,SP               ;LOAD STACK
577                                  ;;**************************************************************
578                                  ;*TEST 1         TEST FOR NO BUS ERRORS
579                                  ;;**************************************************************
580  002670  000004                  TST1:   SCOPE
581  002672  012737  000000  001562          MOV     #0,$TMDAT
582
583                                  ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
584
585                                  ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
586
587                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
588
589                                  ;;**************************************************************
590                                  ;*TEST 2         TEST THAT OUTPUT REG. CAN HOLD #-1
591                                  ;;**************************************************************
592  002730  000004                  TST2:   SCOPE
593  002732  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED
594
595                                  ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
596
597                                  ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
598  002760  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
599  002766  001401                          BEQ     TST3                    ;;BR IF EQUAL
600  002770  104003                          ERROR   3                       ;REG WILL NOT HOLD ONES
601
602                                  ;;**************************************************************
603                                  ;*TEST 3         TEST THAT RESET CLEARS OUTPUT REG.
604                                  ;;**************************************************************
605  002772  000004                  TST3:   SCOPE
606  002774  012737  000004  001166          MOV     #4,$TIMES               ;;DO 4 ITERATIONS
607  003002  005037  001124                  CLR     $GDDAT
608  003006  012737  177777  001562          MOV     #-1,$TMDAT
609
610                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
611  003024  004737  020120                  JSR     PC,$RESET
612
613                                  ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
614  003040  005737  001126                  TST     $BDDAT
```

```
                                        BEQ     TST4            ;;BR IF EQUAL
                                        ERROR   3
                                ;;**************************************************************
                                ;*TEST 4         TEST THAT OUTPUT REG. CAN HOLD #52525
                                ;;**************************************************************
                                TST4:   SCOPE
                                        MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE

                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
                                        CMP     $GDDAT,$BDDAT           ;COMPARE
                                        BEQ     TST5            ;;BR IF EQUAL
                                        ERROR   3                       ;DATA NOT=52525

                                ;;**************************************************************
                                ;*TEST 5         TEST THAT OUTPUT REG. CAN HOLD #125252
                                ;;**************************************************************
                                TST5:   SCOPE
                                        MOV     #125252,$GDDAT          ;LOAD EXPECTED VALUE

                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
                                        CMP     $GDDAT,$BDDAT           ;COMPARE
                                        BEQ     TST6            ;;BR IF EQUAL
                                        ERROR   3                       ;DATA NOT=125252

                                ;;**************************************************************
                                ;*TEST 6         FLOAT A 1 ACROSS THE OUTPUT REGISTER
                                ;;**************************************************************
                                TST6:   SCOPE
                                        MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
                                        MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

                                1$:     MOV     #0,$TMDAT               ;CLEAR OUTPUT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        BIS     $GDDAT,$TMDAT           ;READ OUTPUT REG.

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
                                        CMP     $GDDAT,$BDDAT           ;TEST RESULTS
                                        BEQ     2$              ;BR IF EQUAL ?
                                        ERROR   3

                                2$:     ASL     $GDDAT                  ;SHIFT EXPECTED DATA
                                        BNE     1$              ;BR UNTIL DONE
                                ;;**************************************************************
                                ;*TEST 7         FLOAT A 0 ACROSS THE OUTPUT REGISTER
                                ;;**************************************************************
                                TST7:   SCOPE
                                        MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
```

```
                                                ;LOAD EXPECTED VALUE

1$:     MOV     #-1,$TMDAT                      ;LOAD OUTPUT TO A ONE

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEViCE REG GRDIO
        BIC     $GDDAT,$TMDAT                   ;CLEAR A BIT

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
        COM     $BDDAT                          ;COMPLEMENT IT
        CMP     $GDDAT,$BDDAT                   ;EQUAL ?
        BEQ     2$                              ;BR IF EQUAL
        ERROR   3

2$:     ASL     $GDDAT                          ;SHIFT LEFT
        BNE     1$                              ;BRANCH UNTIL DONe

;;*********************************************************************
;*TEST 10        TEST FOR SLOW OUTPUT GATES WITH #125252
;;*********************************************************************
TST10:  SCOPE
        MOV     #125252,$GDDAT                  ;LOAD EXPECTED VALUE

        ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
        COM     $TMDAT

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMdAT.
        COM     $TMDAT

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMdAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
        COM     $TMDAT

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
        COM     $TMDAT

        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
        COM     $TMDAT
```

```
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
003604  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
003630  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REC GRDIO,PUT DATA IN $TMDAT.
003654  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
003700  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
003714  000240                                  NOP
003716  000240                                  NOP
                                        ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
003730  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;TEST REGISTER
003736  001401                                  BEQ     TST11           ;;BR IF CORRECT
003740  104003                                  ERROR   3

                                        ;;*******************************************************************
                                        ;*TEST 11        TEST FOR SLOW OUTPUT GATES WITH #52525
                                        ;;*******************************************************************
003742  000004                        TST11:  SCOPE
003744  012737  052525  001124                  MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE
                                        ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
003772  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004016  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004042  005137  001562                          COM     $TMDAT
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004066  005137  001562                          COM     $TMDAT
```

```
                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004112  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004136  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004162  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004206  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004232  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                        ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
004256  005137  001562                          COM     $TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
004272  000240                                  NOP

                                        ;*      MOV     @GRDIO,$BDDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
004304  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;TEST PATTERN
004312  001401                                  BEQ     TST12           ;;BR IF EQUAL
004314  104003                                  ERROR   3                       ;OUTPUT REGISTER IN ERROR

                                        ;;*****************************************************************
                                        ;*TEST 12       TEST RELAY #1 STATUS BIT
                                        ;;*****************************************************************
004316  000004                          TST12:  SCOPE
004320  012737  000000  001562                  MOV     #0,$TMDAT

                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
004336  012737  177777  001562                  MOV     #-1,$TMDAT

                                        ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
004354  012737  000001  001124                  MOV     #BIT0,$GDDAT            ;LOAD EXPECTED

                                        ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT

                                        ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
```

```
                                                                                ;TEST IT
004402  033737  001124  001126        BIT     $GDDAT,$BDDAT
004410  001001                        BNE     TST13                   ;;BR IF SET
004412  104001                        ERROR   1                       ;ERROR, RELAY 1 FAILED TO sET

                        ;;**************************************************************
                        ;*TEST 13        TEST RELAY #2 STATUS BIT
                        ;;**************************************************************
004414  000004          TST13:  SCOPE
004416  012737  000400  001124        MOV     #BIT8,$GDDAT            ;LOAD EXPECT

                        ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT

                        ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
004444  033737  001124  001126        BIT     $GDDAT,$BDDAT
004452  001001                        BNE     TST14                   ;;BR IF SET
004454  104001                        ERROR   1                       ;ERROR, RELAY 2 FAILED TO SET

                        ;;**************************************************************
                        ;*TEST 14        TEST OUTPUT DATA ACCEPT FLAG
                        ;;**************************************************************
004456  000004          TST14:  SCOPE
004460  012737  100000  001124        MOV     #BIT15,$GDDAT           ;LOAD EXPECTED

                        ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT

                        ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
004506  033737  001124  001126        BIT     $GDDAT,$BDDAT
004514  001001                        BNE     TST15                   ;;BR IF SET
004516  104001                        ERROR   1                       ;ERROR, BIT 15 FAILED TO sET

                        ;;**************************************************************
                        ;*TEST 15        TEST OUTPUT INTERRUPT ENABLE
                        ;;**************************************************************
004520  000004          TST15:  SCOPE
004522  012737  000000  001562        MOV     #0,$TMDAT               ;CLEAR STATUS

                        ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
004540  012737  040000  001124        MOV     #BIT14,$GDDAT           ;LOAD EXPECTED

                        ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT

                        ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
004566  033737  001124  001126        BIT     $GDDAT,$BDDAT
004574  001001                        BNE     TST16                   ;;BR IF SET
004576  104001                        ERROR   1                       ;ERROR BIT 14 FAILED TO SET


                        ;;**************************************************************
                        ;*TEST 16        TEST INPUT DATA READY FLAG
                        ;;**************************************************************
004600  000004          TST16:  SCOPE
004602  012737  000200  001124        MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
```

```
                                ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
                                ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
004630  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004636  001401                         BEQ     TST17           ;;BR IF EQUAL
004640  104001                         ERROR   1               ;ERROR, BIT 7 FAILED TO SET

                                ;;*************************************************************
                                ;*TEST 17        TEST INPUT INTERRUPT ENABLE
                                ;;*************************************************************
004642  000004                  TST17:  SCOPE
004644  012737  000000  001562          MOV     #0,$TMDAT               ;CLEAR STATUS

                                ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
004662  012737  000100  001124          MOV     #BIT6,$GDDAT            ;LOAD EXPECTED

                                ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT

                                ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
004710  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004716  001401                         BEQ     TST20           ;;BR IF EQUAL
004720  104001                         ERROR   1               ;ERROR, BIT 6 FAILED TO SET

                                ;;*************************************************************
                                ;*TEST 20        TEST THAT RESET CLEARS THE DIGITAL STATUS REG.
                                ;;*************************************************************
004722  000004                  TST20:  SCOPE
004724  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
004732  005037  001124                  CLR     $GDDAT

                                ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
004746  012737  140701  001562          MOV     #BIT15!BIT14!BIT8!BIT7!BIT6!BIT0,$TMDAT

                                ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
004764  004737  020120                  JSR     PC,$RESET

                                ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
005000  042737  100000  001126          BIC     #BIT15,$BDDAT
005006  005737  001126                  TST     $BDDAT
005012  001401                         BEQ     TST21           ;; BR IF EQUAL
005014  104001                         ERROR   1

                                ;;*************************************************************
                                ;*TEST 21        TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
                                ;;*************************************************************
005016  000004                  TST21:  SCOPE
005020  032777  002000  174112          BIT     #BIT10,@SWR             ;TEST SWITCH BIT
005026  001402                         BEQ     1$                      ;BRANCH IF DOWN
005030  000137  011176                  JMP     DRT21                   ;BYPASS SOME TEST USING THE EXTERNAL CABLE
005034                          1$:
005034  012737  000000  001562          MOV     #0,$TMDAT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
005052  012737  177777  001562          MOV     #-1,$TMDAT
```

```
                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
005070  005037  001124                CLR     $GDDAT                  ;CLEAR EXPECTED
005074  012737  000000  001562        MOV     #0,$TMDAT               ;LOAD THE OUTPUT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005122  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
005130  001401                        BEQ     TST22           ;;BR IF EQUAL
005132  104002                        ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT REG.

                                ;;**************************************************************
                                ;*TEST 22        TEST INPUT WITH #-1
                                ;;**************************************************************
005134  000004                  TST22:  SCOPE
005136  012737  000000  001562        MOV     #0,$TMDAT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
005154  012737  177777  001562        MOV     #-1,$TMDAT

                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
005172  012737  177777  001124        MOV     #-1,$GDDAT              ;LOAD EXPECTED

                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005220  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
005226  001401                        BEQ     TST23           ;;BR IF EQUAL
005230  104002                        ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT
                                                                ;IS WRAP-AROUND CABLE CONNECTED ???

                                ;;**************************************************************
                                ;*TEST 23        TEST INPUT WITH #52525
                                ;;**************************************************************
005232  000004                  TST23:  SCOPE
005234  012737  000000  001562        MOV     #0,$TMDAT

                                ;*      MOV     $TMdAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
005252  012737  177777  001562        MOV     #-1,$TMDAT

                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
005270  012737  052525  001124        MOV     #52525,$GDDAT           ;LOAD EXPECTED

                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005316  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
005324  001401                        BEQ     TST24           ;;BR IF EQUAL
005326  104002                        ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT

                                ;;**************************************************************
                                ;*TEST 24        TEST INPUT WITH #125252
                                ;;**************************************************************
```

```
                                TST24:  SCOPE
005330  000004
005332  012737  000000  001562          MOV     #0,$TMDAT
                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
005350  012737  177777  001562          MOV     #-1,$TMDAT
                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
005366  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED
                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005414  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005422  001401                          BEQ     TST25                   ;;BR IF EQUAL
005424  104002                          ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT

                                ;;*********************************************************************
                                ;*TEST 25       FLOAT A 1 ACROSS NON-LATCHING INPUT BITS
                                ;*********************************************************************
005426  000004                  TST25:  SCOPE
005430  012737  005456  001110          MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN

005436  012737  000001  001556          MOV     #BIT0,BRLEV2            ;LOAD EXPECTED
005444  013737  001536  001560          MOV     NOTLCH,BRLEV3           ;GET NON-LATCH
005452  005137  001560                  COM     BRLEV3                  ;COMPLEMENT
005456  013737  001556  001124  2$:     MOV     BRLEV2,$GDDAT           ;LOAD GOOD
005464  033737  001124  001536          BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCH
005472  001451                          BEQ     1$                      ;BR IF LATCHING

                                ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005514  043737  001560  001126          BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
005522  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005530  001401                          BEQ     3$                      ;;BR IF EQUAL
005532  104002                          ERROR   2                       ;INPUT REGISTER IN ERROR
                                                                        ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

                                ;SUB-TEST CLEAR THE OUTPUT BIT AND TEST THE INPUT DOES NOT LATCH

005534                          3$:

                                ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
005544  043737  001124  001562          BIC     $GDDAT,$TMDAT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005572  043737  001560  001126          BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
005600  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
005604  033737  001556  001126          BIT     BRLEV2,$BDDAT           ;TEST FOR BIT
005612  001401                          BEQ     1$                      ;;BR IF CLEARED
005614  104002                          ERROR   2                       ;INPUT BIT LATCHED IN ERROR
```

```
                                                                ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

005616  006337  001556        1$:     ASL     BRLEV2                  ;CHANGE PATTERN
005622  001315                        BNE     2$                      ;BR UNTIL DONE

                        ;;**********************************************************************
                        ;*TEST 26        FLOAT A 1 ACROSS LATCHING INPUT BITS
                        ;;**********************************************************************
005624  000004           TST26:  SCOPE
005626  012737  005642  001110        MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
005634  012737  000001  001124        MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

005642  033737  001124  001536  2$:    BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCHING
005650  001042                        BNE     1$                      ;BR IF NON-LATCH

005652  012737  000000  001562        MOV     #0,$TMDAT

                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
005670  012737  177777  001562        MOV     #-1,$TMDAT

                        ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI

                        ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
005716  012737  000000  001562        MOV     #0,$TMDAT

                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                        ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
005744  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
005752  001401                        BEQ     1$                      ;;BR IF EQUAL
005754  104002                        ERROR   2                       ;INPUT REGISTER FAILED TO LATCH DATA
                                                        ;IS THIS REALLY A "LPS-11-DRA" ??       FIRST ERROR
                                                        ;IF RUNNING ON A "LPS-11-DR"  IF "DR" USE MD-11-DZLPD

005756  006337  001124        1$:     ASL     $GDDAT                  ;CHANGE PATTERN
005762  001327                        BNE     2$                      ;BR UNTIL DONE

                        ;;**********************************************************************
                        ;*TEST 27        FLOAT A 0 ACROSS LATCHING INPUT BITS
                        ;;**********************************************************************
005764  000004           TST27:  SCOPE
005766  012737  006002  001110        MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
005774  012737  000001  001560        MOV     #BIT0,BRLEV3            ;LOAD EXPECTED

006002  033737  001560  001536  2$:    BIT     BRLEV3,NOTLCH           ;TEST FOR LATCHING
006010  001055                        BNE     1$                      ;BR IF NOT

006012  012737  000000  001562        MOV     #0,$TMDAT

                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMdAT TO DEVICE REG GRDIO
006030  012737  177777  001562        MOV     #-1,$TMDAT
```

```
1101                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1102    006046  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD
1103    006054  043737  001536  001124          BIC     NOTLCH,$GDDAT
1104    006062  000240                          NOP
1105    006064  000240                          NOP
1106    006066  043737  001560  001124          BIC     BRLEV3,$GDDAT           ;MAKE BRLEV3
1107
1108                                    ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
1109    006104  012737  000000  001562          MOV     #0,$TMDAT
1110
1111                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1112
1113
1114                                    ;*      MOV     @GRDAI,$BDDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $BDDAT.
1115    006132  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1116    006140  001401                          BEQ     1$              ;;BR IF EQUAL
1117    006142  104002                          ERROR   2               ;INPUT REGISTER FAILED TO LATCH DATA
1118
1119    006144  006337  001560          1$:     ASL     BRLEV3                  ;CHANGE PATTERN
1120    006150  001314                          BNE     2$                      ;BR UNTIL DONE
1121
1122
1123                                    ;;**********************************************************************
1124                                    ;*TEST 30        TEST FOR SLOW INPUT GATES WITH #125252
1125                                    ;;**********************************************************************
1126    006152  000004                  TST30:  SCOPE
1127
1128    006154  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED
1129    006162  043737  001536  001124          BIC     NOTLCH,$GDDAT           ;CONVERT
1130    006170  013700  001124                  MOV     $GDDAT,R0               ;LOAD PATTERN
1131    006174  012737  000000  001562          MOV     #0,$TMDAT
1132
1133                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1134    006212  012737  177777  001562          MOV     #-1,$TMDAT
1135
1136                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1137
1138    006230  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1139
1140                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1141    006244  012737  000000  001562          MOV     #0,$TMDAT
1142
1143                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1144
1145                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1146    006272  013701  001562                  MOV     $TMDAT,R1
1147
1148                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1149    006306  050137  001562                  BIS     R1,$TMDAT
1150
1151                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1152    006322  005100                          COM     R0
1153    006324  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1154
```

```
1155                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1156  006340  012737  000000  001562            MOV     #0,$TMDAT
1157
1158                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1159
1160                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1161  006366  013701  001562                    MOV     $TMDAT,R1
1162
1163                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1164  006402  050137  001562                    BIS     R1,$TMDAT
1165
1166                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1167  006416  005100                            COM     R0
1168  006420  010037  001562                    MOV     R0,$TMDAT               ;LOAD OUTPUT
1169
1170                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1171  006434  012737  000000  001562            MOV     #0,$TMDAT
1172
1173                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1174
1175                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1176  006462  013701  001562                    MOV     $TMDAT,R1
1177
1178                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1179  006476  050137  001562                    BIS     R1,$TMDAT
1180
1181                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1182  006512  005100                            COM     R0
1183  006514  010037  001562                    MOV     R0,$TMDAT               ;LOAD OUTPUT
1184
1185                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1186  006530  012737  000000  001562            MOV     #0,$TMDAT
1187
1188                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO dEVICE REG GRDIO
1189
1190                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1191  006556  013701  001562                    MOV     $TMDAT,R1
1192
1193                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1194  006572  050137  001562                    BIS     R1,$TMDAT
1195
1196                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1197  006606  005100                            COM     R0
1198  006610  010037  001562                    MOV     R0,$TMDAT               ;LOAD OUTPUT
1199
1200                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1201  006624  012737  000000  001562            MOV     #0,$TMDAT
1202
1203                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMdAT TO DEViCE REG GRDIO
1204
1205                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1206  006652  013701  001562                    MOV     $TMDAT,R1
1207
1208                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
```

```
1209  006666  050137  001562                BIS     R1,$TMDAT
1210
1211                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1212  006702  005100                       COM     R0
1213  006704  010037  001562                MOV     R0,$TMDAT               ;LOAD OUTPUT
1214
1215                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1216  006720  012737  000000  001562        MOV     #0,$TMDAT
1217
1218                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1219
1220                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1221  006746  013701  001562                MOV     $TMDAT,R1
1222
1223                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1224  006762  050137  001562                BIS     R1,$TMDAT
1225
1226                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1227  006776  005100                       COM     R0
1228  007000  010037  001562                MOV     R0,$TMDAT               ;LOAD OUTPUT
1229
1230                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1231  007014  012737  000000  001562        MOV     #0,$TMDAT
1232
1233                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1234
1235                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1236  007042  013701  001562                MOV     $TMDAT,R1
1237
1238                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1239  007056  050137  001562                BIS     R1,$TMDAT
1240
1241                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1242  007072  005100                       COM     R0
1243  007074  010037  001562                MOV     R0,$TMDAT               ;LOAD OUTPUT
1244
1245                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1246  007110  012737  000000  001562        MOV     #0,$TMDAT
1247
1248                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1249
1250                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1251  007136  013701  001562                MOV     $TMDAT,R1
1252
1253                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1254  007152  050137  001562                BIS     R1,$TMDAT
1255
1256                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1257  007166  005100                       COM     R0
1258  007170  010037  001562                MOV     R0,$TMDAT               ;LOAD OUTPUT
1259
1260                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1261  007204  012737  000000  001562        MOV     #0,$TMDAT
1262
```

```
1263                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1264
1265                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1266  007232  013701  001562                      MOV     $TMDAT,R1
1267
1268                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1269  007246  050137  001562                      BIS     R1,$TMDAT
1270
1271                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1272  007262  005100                              COM     R0
1273  007264  010037  001562                      MOV     R0,$TMDAT               ;LOAD OUTPUT
1274
1275                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1276  007300  012737  000000  001562              MOV     #0,$TMDAT
1277
1278                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1279
1280                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1281  007326  013701  001562                      MOV     $TMDAT,R1
1282
1283                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1284  007342  050137  001562                      BIS     R1,$TMDAT
1285
1286                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1287  007356  005100                              COM     R0
1288  007360  010037  001562                      MOV     R0,$TMDAT               ;LOAD OUTPUT
1289
1290                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1291  007374  012737  000000  001562              MOV     #0,$TMDAT
1292
1293                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1294
1295                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1296  007422  013701  001562                      MOV     $TMDAT,R1
1297
1298                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1299  007436  050137  001562                      BIS     R1,$TMDAT
1300
1301                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1302  007452  005100                              COM     R0
1303
1304  007454  010137  001126                      MOV     R1,$BDDAT               ;LOAD READ
1305  007460  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
1306  007466  001401                              BEQ     TST31           ;;BR IF EQUAL
1307  007470  104002                              ERROR   2                       ;INPUT GATE SLOW
1308
1309                                    ;;*************************************************************************
1310                                    ;*TEST 31        TEST FOR SLOW INPUT GATES WITH #52525
1311                                    ;;*************************************************************************
1312  007472  000004                    TST31:  SCOPE
1313
1314  007474  012737  052525  001124              MOV     #52525,$GDDAT           ;SETUP EXPECTED
1315  007502  043737  001536  001124              BIC     NOTLCH,$GDDAT           ;CONVERT
1316  007510  013700  001124                      MOV     $GDDAT,R0
```

```
1317  007514  012737  000000  001562          MOV     #0,$TMDAT
1318
1319                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1320  007532  012737  177777  001562          MOV     #-1,$TMDAT
1321
1322                                  ;*        MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1323
1324  007550  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1325
1326                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1327  007564  012737  000000  001562          MOV     #0,$TMDAT
1328
1329                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1330
1331                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1332  007612  013701  001562                  MOV     $TMDAT,R1
1333
1334                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1335  007626  050137  001562                  BIS     R1,$TMDAT
1336
1337                                  ;*        MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1338  007642  005100                          COM     R0
1339  007644  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1340
1341                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1342  007660  012737  000000  001562          MOV     #0,$TMDAT
1343
1344                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1345
1346                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1347  007706  013701  001562                  MOV     $TMDAT,R1
1348
1349                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1350  007722  050137  001562                  BIS     R1,$TMDAT
1351
1352                                  ;*        MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1353  007736  005100                          COM     R0
1354  007740  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1355
1356                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1357  007754  012737  000000  001562          MOV     #0,$TMDAT
1358
1359                                  ;*        MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1360
1361                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1362  010002  013701  001562                  MOV     $TMDAT,R1
1363
1364                                  ;*        MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1365  010016  050137  001562                  BIS     R1,$TMDAT
1366
1367                                  ;*        MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1368  010032  005100                          COM     R0
1369  010034  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1370
```

```
1371                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1372  010050  012737  000000  001562                MOV     #0,$TMDAT
1373
1374                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1375
1376                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1377  010076  013701  001562                        MOV     $TMDAT,R1
1378
1379                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1380  010112  050137  001562                        BIS     R1,$TMDAT
1381
1382                                        ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1383  010126  005100                                COM     R0
1384  010130  010037  001562                        MOV     R0,$TMDAT               ;LOAD OUTPUT
1385
1386                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1387  010144  012737  000000  001562                MOV     #0,$TMDAT
1388
1389                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1390
1391                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMdAT.
1392  010172  013701  001562                        MOV     $TMDAT,R1
1393
1394                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMdAT.
1395  010206  050137  001562                        BIS     R1,$TMDAT
1396
1397                                        ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1398  010222  005100                                COM     R0
1399  010224  010037  001562                        MOV     R0,$TMDAT               ;LOAD OUTPUT
1400
1401                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1402  010240  012737  000000  001562                MOV     #0,$TMDAT
1403
1404                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1405
1406                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1407  010266  013701  001562                        MOV     $TMDAT,R1
1408
1409                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1410  010302  050137  001562                        BIS     R1,$TMDAT
1411
1412                                        ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1413  010316  005100                                COM     R0
1414  010320  010037  001562                        MOV     R0,$TMDAT               ;LOAD OUTPUT
1415
1416                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1417  010334  012737  000000  001562                MOV     #0,$TMDAT
1418
1419                                        ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1420
1421                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1422  010362  013701  001562                        MOV     $TMDAT,R1
1423
1424                                        ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
```

```
1425  010376  050137  001562                  BIS     R1,$TMDAT
1426
1427                                  ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1428  010412  005100                          COM     R0
1429  010414  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1430
1431                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1432  010430  012737  000000  001562          MOV     #0,$TMDAT
1433
1434                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1435
1436                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1437  010456  013701  001562                  MOV     $TMDAT,R1
1438
1439                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1440  010472  050137  001562                  BIS     R1,$TMDAT
1441
1442                                  ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1443  010506  005100                          COM     R0
1444  010510  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1445
1446                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1447  010524  012737  000000  001562          MOV     #0,$TMDAT
1448
1449                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1450
1451                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1452  010552  013701  001562                  MOV     $TMDAT,R1
1453
1454                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1455  010566  050137  001562                  BIS     R1,$TMDAT
1456
1457                                  ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1458  010602  005100                          COM     R0
1459  010604  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT
1460
1461                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1462  010620  012737  000000  001562          MOV     #0,$TMDAT
1463
1464                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1465
1466                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMdAT.
1467  010646  013701  001562                  MOV     $TMDAT,R1
1468
1469                                  ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1470  010662  050137  001562                  BIS     R1,$TMDAT
1471
1472                                  ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1473  010676  005100                          COM     R0
1474  010700  010037  001562                  MOV     R0,$TMDAT               ;LOAD OUTPUT.
1475
1476                                  ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1477  010714  012737  000000  001562          MOV     #0,$TMDAT
1478
```

```
1479                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1480
1481                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1482  010742  013701  001562                       MOV     $TMDAT,R1
1483
1484                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1485  010756  050137  001562                       BIS     R1,$TMDAT
1486
1487                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1488  010772  005100                               COM     R0
1489
1490  010774  010137  001126                       MOV     R1,$BDDAT               ;LOAD VALUE READ
1491  011000  023737  001124  001126               CMP     $GDDAT,$BDDAT           ;COMPARE
1492  011006  001401                               BEQ     TST32           ;;BR IF EQUAL
1493  011010  104002                               ERROR   2
1494
1495                                    ;;**************************************************************
1496                                    ;*TEST 32       TEST THAT WHEN OUTPUTTING THE INPUT DATA READY FLAG SETS
1497                                    ;;**************************************************************
1498  011012  000004                    TST32:  SCOPE
1499  011014  012737  000200  001124               MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1500  011022  012737  000000  001562               MOV     #0,$TMDAT
1501
1502                                    ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1503
1504                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1505  011050  022727  000000  000000               CMP     #0,#0                   ;DELAY
1506
1507                                    ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
1508  011066  023737  001124  001126               CMP     $GDDAT,$BDDAT           ;COMPARE
1509  011074  001401                               BEQ     TST33           ;;BR IF EQUAL
1510  011076  104001                               ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET
1511
1512                                    ;TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET
1513
1514                                    ;;**************************************************************
1515                                    ;*TEST 33       TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET
1516                                    ;;**************************************************************
1517  011100  000004                    TST33:  SCOPE
1518  011102  012737  100000  001124               MOV     #BIT15,$GDDAT           ;LOAD EXPECTED
1519  011110  005037  001562                       CLR     $TMDAT
1520
1521                                    ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1522
1523                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1524  011134  022727  000000  000000               CMP     #0,#0
1525
1526                                    ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
1527  011152  013700  001562                       MOV     $TMDAT,R0
1528
1529                                    ;*      MOV     @GRSTAT,$BDDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
1530  011166  005737  001126                       TST     $BDDAT
1531  011172  100401                               BMI     TST34           ;;BR IF SET
1532  011174  104001                               ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET
```

```
                                        ;TEST THAT THE DIGITAL I/O DOES NOT INTERRUPT
                                        ;SET INPUT-OUTPUT FLAGS BUT DO NOT ENABLE INTERRUPTS
011176                                  DRT21:
                                        ;;*****************************************************************
                                        ;*TEST 34       TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG
                                        ;;*****************************************************************
011176  000004                          TST34:  SCOPE
011200  032777  002000  167732                  BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
011206  001137                          BNE     TST35           ;;BYPASS IF NO I/O CABLE

011210  012737  011224  001110          MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
011216  012737  000001  001560          MOV     #BIT0,BRLEV3            ;LOAD INTERRUPT BIT
011224  005037  001126          1$:     CLR     $BDDAT                  ;CLEAR BAD DATA
011230  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD GOOD DATA

011236  033737  001560  001540          BIT     BRLEV3,NOTINT           ;TEST IF SET TO INT
011244  001115                          BNE     3$              ;;NO TRY NEXT BIT
011246  012737  000000  001562          MOV     #0,$TMDAT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
011264  012737  177777  001562          MOV     #-1,$TMDAT

                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
011302  012737  000000  001562          MOV     #0,$TMDAT               ;CLEAR STATUS

                                ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMdAT TO DEVICE REG GRSTAT

                                ;*      MOV     BRLEV3,@GRDIO   ;/ PUT DATA FROM BRLEV3 TO DEVICE REG GRDIO

                                ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
                                                                ;SHOULD REMAIN SET VIA DIRECT SET SIDE

                                ;*      MOV     @GRSTAT,$TMDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $TMDAT.
011350  005737  001562                  TST     $TMDAT
011354  100401                          BMI     2$              ;;BR IF SET
011356  104010                          ERROR   10                      ;INPUT INTERRUPT BIT FAILED TO SET INPUT READY
                                                                        ;?? DID OPERATOR GIVE CORRECT
                                                                        ;INPUT INTERRUPT BITS ??

011360                          2$:

                                ;*      MOV     @GRDIO,$TMDAT   ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
011370  043737  001560  001562          BIC     BRLEV3,$TMDAT

                                ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO

                                ;*      MOV     @GRDAI,$TMDAT   ;/READ DEVICE REG GRDAI,PUT DATA IN $TMDAT.
011416  053737  001560  001562          BIS     BRLEV3,$TMDAT

                                ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
011434  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
011440  012737  000000  001562          MOV     #0,$TMDAT               ;CLEAR STATUS
```

```
1587                                    ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1588
1589                                    ;*      MOV     @GRSTAT,$TMDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $TMDAT.
1590  011466  113737  001562  001126          MOVB    $TMDAT,$BDDAT
1591  011474  100001                        BPL     3$              ;;BR IF CLEARED
1592  011476  104001                        ERROR   1                       ;INPUT READY FAILED TO CLEAR
1593
1594  011500  006337  001560          3$:   ASL     BRLEV3                  ;CHANGE BIT
1595  011504  001247                        BNE     1$                      ;BR IF NOT DONE
1596                                  ;;******************************************************************
1597                                  ;*TEST 35        TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG
1598                                  ;;******************************************************************
1599  011506  000004                  TST35:  SCOPE
1600  011510  032777  002000  167422          BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
1601  011516  001075                        BNE     TST36           ;;BYPASS IF NO I/O CABLE
1602
1603  011520  012737  011534  001110          MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
1604  011526  012737  000001  001560          MOV     #BIT0,BRLEV3            ;LOAD NON-INTERRUPT BIT
1605  011534  012737  000200  001126  1$:     MOV     #200,$BDDAT             ;LOAD BAD DATA
1606  011542  005037  001124                  CLR     $GDDAT                  ;CLEAR GOOD DATA
1607
1608  011546  033737  001560  001540          BIT     BRLEV3,NOTINT           ;TEST IF SET TO INT
1609  011554  001453                        BEQ     3$              ;;NO SKIP AND TRY NEXT BIT
1610  011556  012737  000000  001562          MOV     #0,$TMDAT
1611
1612                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1613  011574  012737  177777  001562          MOV     #-1,$TMDAT
1614
1615                                    ;*      MOV     $TMDAT,@GRDAI   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1616  011612  012737  000000  001562          MOV     #0,$TMDAT               ;CLEAR STATUS
1617
1618                                    ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1619  011630  013737  001560  001562          MOV     BRLEV3,$TMDAT           ;LOAD OUTPUT/INPUT
1620
1621                                    ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1622  011646  012737  000000  001562          MOV     #0,$TMDAT               ;CLEAR FLAG FROM DATA READY
1623
1624                                    ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1625                                                                    ;SHOULD REMAIN SET VIA DIRECT SET SIDE
1626
1627                                    ;*      MOV     @GRSTAT,$TMDAT  ;/READ DEVICE REG GRSTAT,PUT DATA IN $TMdAT.
1628  011674  105737  001562                  TSTB    $TMDAT
1629  011700  100001                        BPL     3$              ;;BR IF CLEAR
1630  011702  104011                        ERROR   11                      ;INPUT NON-INTERRUPT BIT SET INPUT READY
1631                                                                  ;?? DID OPERATOR GIVE CORRECT
1632                                                                  ;INPUT INTERRUPT BITS ??
1633
1634
1635  011704  006337  001560          3$:   ASL     BRLEV3                  ;CHANGE BIT
1636  011710  001311                        BNE     1$                      ;BR IF NOT DONE
1637
1638                                  ;;******************************************************************
1639                                  ;*TEST 36        END OF THE PROGRAM
1640                                  ;;******************************************************************
```

```
                                TST36:  SCOPE
1641  011712  000004
1642  011714  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
1643  011722  004737  020120                  JSR     PC,$RESET
1644  011726  013777  001544  167606          MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1645  011734  005077  167604                  CLR     @GRIVSA
1646  011740  013777  001550  167600          MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1647  011746  005077  167576                  CLR     @GRIVSB
1648  011752  005037  177776                  CLR     PSW
1649  011756  000137  012250                  JMP     $EOP
1650
1651
1652                            ;;*****************************************************************
1653                            ;*TEST 37        MISC. EXTERNAL LOGIC TEST
1654                            ;;*****************************************************************
1655  011762  000004            TST37:  SCOPE
1656  011764  012706  001100    EXTTST: MOV     #STACK,SP               ;SET UP STACK
1657  011770  005037  001464            CLR     .DVLS                   ;START THE MICRO-CODE
1658  011774  005037  001124            CLR     $GDDAT
1659
1660                            ;*      MOV     $GDDAT,@GRSTAT  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
1661
1662                            ;*      MOV     $GDDAT,@GRDIO   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
1663
1664                            ;*      MOV     $GDDAT,@GRDAI   ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDAI
1665  012030  012737  000001  001562          MOV     #BIT0,$TMDAT            ;SET RELAY #1
1666
1667                            ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1668  012046  004737  012076            JSR     PC,FLIP                 ;TOGGLE THE INPUT-OUTPUT REG
1669  012052  012737  000400  001562          MOV     #BIT8,$TMDAT            ;SET REALY #2
1670
1671                            ;*      MOV     $TMDAT,@GRSTAT  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1672  012070  004737  012076            JSR     PC,FLIP                 ;TOGGLE THE INPUT-OUTPUT REG
1673  012074  000733                    BR      EXTTST
1674
1675  012076  012737  040000  012246  FLIP:   MOV     #40000,EXTCNT           ;LOAD COUNT
1676  012104                            1$:
1677  012104  012737  000000  001562          MOV     #0,$TMDAT
1678
1679                            ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1680  012122  012737  125252  001562          MOV     #125252,$TMDAT          ;LOAD OUTPUT
1681
1682                            ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1683
1684                            ;*      MOV     @GRDAI,EXTTMP   ;/READ DEVICE REG GRDAI,PUT DATA IN EXTTMP.
1685
1686                            ;*      MOV     EXTTMP,@GRDAI   ;/ PUT DATA FROM EXTTMP TO DEVICE REG GRDAI
1687  012160  012737  000000  001562          MOV     #0,$TMDAT
1688
1689                            ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1690  012176  012737  052525  001562          MOV     #52525,$TMDAT           ;LOAD OUTPU
1691
1692                            ;*      MOV     $TMDAT,@GRDIO   ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1693
1694                            ;*      MOV     @GRDAI,EXTTMP   ;/READ DEVICE REG GRDAI,PUT DATA IN EXTTMP.
```

```
                                        ;*      MOV     EXTTMP,@GRDAI   ;/ PUT DATA FROM EXTTMP TO DEVICE REG GRDAI
012234  005337  012246                          DEC     EXTCNT                  ;FINISHED COUNT
012240  001321                                  BNE     1$
012242  000207                                  RTS     PC                      ;EXIT

012244  000000                        EXTTMP: 0
012246  000000                        EXTCNT: 0

                                        .SBTTL  END OF PASS ROUTINE

                                        ;;*****************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS"
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO FULNUL
                                        ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
                                        ;*SENDMG CAN BE CHANGED TO 7.

012250                                  $EOP:
012250  000004                                  SCOPE
012252  005037  001102                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
012256  005037  001166                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
012262  005237  001204                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
012266  042737  100000  001204                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
012274  005327                                  DEC     (PC)+           ;;LOOP?
012276  000001                        $EOPCT: .WORD   1
012300  003015                                  BGT     $DOAGN          ;;YES
012302  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
012304  000001                        $ENDCT: .WORD   1
012306  012276                                  $EOPCT
012310  104401  012343                          TYPE    ,$ENDMG         ;;TYPE "END PASS"
012314  013700  000042                $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
012320  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
012322  000005                                  RESET                   ;;CLEAR THE WORLD
012324  004710                        $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
012326  000240                                  NOP                     ;;SAVE ROOM
012330  000240                                  NOP                     ;;FOR
012332  000240                                  NOP                     ;;ACT11
012334                                  $DOAGN:
012334  000137                                  JMP     @(PC)+          ;;RETURN
012336  012356                        $RTNAD: .WORD   FULNUL
012340     377        377     000     $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
012343     015  042412  042116        $ENDMG: .ASCIZ  <15><12>/END PASS/
012350  050040  051501  000123
012356  005037  012374                FULNUL: CLR     10$
012362  005337  012374                1$:     DEC     10$
012366  001375                                  BNE     1$
012370  000137  002662                          JMP     IOTEST
012374  000000                        10$:    0

012376  052123  052101  051525        EM1:    .ASCIZ  /STATUS REGISTER IN ERROR/
012404  051040  043505  051511
```

```
1749  012412  042524  020122  047111
1750  012420  042440  051122  051117
1751  012426     000
1752  012427     111  050116  052125  EM2:    .ASCIZ  /INPUT REGISTER IN ERROR/
1753  012434  051040  043505  051511
1754  012442  042524  020122  047111
1755  012450  042440  051122  051117
1756  012456     000
1757  012457     117  052125  052520  EM3:    .ASCIZ  /OUTPUT REGISTER IN ERROR/
1758  012464  020124  042522  044507
1759  012472  052123  051105  044440
1760  012500  020116  051105  047522
1761  012506  000122
1762  012510  047111  052520  020124  EM4:    .ASCIZ  /INPUT FAILED TO INTERRUPT/
1763  012516  040506  046111  042105
1764  012524  052040  020117  047111
1765  012532  042524  051122  050125
1766  012540  000124
1767  012542  052517  050124  052125  EM5:    .ASCIZ  /OUTPUT FAILED TO INTERRUPT/
1768  012550  043040  044501  042514
1769  012556  020104  047524  044440
1770  012564  052116  051105  052522
1771  012572  052120     000
1772  012575     125  042516  050130  EM6:    .ASCIZ  /UNEXPECTED INTERRUPT/
1773  012602  041505  042524  020104
1774  012610  047111  042524  051122
1775  012616  050125  000124
1776  012622  050117  051105  052101  EM7:    .ASCIZ  /OPERATOR INTERVENTION ERROR/
1777  012630  051117  044440  052116
1778  012636  051105  042526  052116
1779  012644  047511  020116  051105
1780  012652  047522  000122
1781  012656  047111  042524  051122  EM10:   .ASCIZ  /INTERRUPT INPUT BIT FAILED TO sET INPUT READY FLAG/
1782  012664  050125  020124  047111
1783  012672  052520  020124  044502
1784  012700  020124  040506  046111
1785  012706  042105  052040  020117
1786  012714  042523  020124  047111
1787  012722  052520  020124  042522
1788  012730  042101  020131  046106
1789  012736  043501     000
1790  012741     116  047117  044455  EM11:   .ASCIZ  /NON-INTERRUPT INPUT BIT SET INPUT READY FLAG/
1791  012746  052116  051105  052522
1792  012754  052120  044440  050116
1793  012762  052125  041040  052111
1794  012770  051440  052105  044440
1795  012776  050116  052125  051040
1796  013004  040505  054504  043040
1797  013012  040514  000107
1798
1799  013016  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   STATUS  EXPECTED/
1800  013024  020040  052123  052101
1801  013032  051525  020040  054105
1802  013040  042520  052103  042105
```

```
1803  013046  000
1804  013047  105 051122 041520 DH2:    .ASCIZ  /ERRPC   INPUT   EXPECTED/
1805  013054 020040 044440 050116
1806  013062 052125 020040 042440
1807  013070 050130 041505 042524
1808  013076 000104
1809  013100 051105 050122 020103 DH3:    .ASCIZ  /ERRPC   OUTPUT  EXPECTED/
1810  013106 020040 052517 050124
1811  013114 052125 020040 054105
1812  013122 042520 052103 042105
1813  013130  000
1814  013131  105 051122 041520 DH4:    .ASCIZ  /ERRPC/
1815  013136  000
1816  013137  105 051122 041520 DH10:   .ASCIZ  /ERRPC   STATUS  EXPECT  INPUT BIT/
1817  013144 020040 051440 040524
1818  013152 052524 020123 042440
1819  013160 050130 041505 020124
1820  013166 044440 050116 052125
1821  013174 041040 052111  000
1822         013202                   .EVEN
1823  013202 001116 001126 001124 DT1:    $ERRPC,$BDDAT,$GDDAT,0
1824  013210 000000
1825  013212 001116 000000        DT4:    $ERRPC,0
1826  013216 001116 001126 001124 DT10:   $ERRPC,$BDDAT,$GDDAT,BRLEV3,0
1827  013224 001560 000000
1828  013230  000   000   000   DF1:    .BYTE   0,0,0,0
1829  013233  000
1830  013234  000   000   000   DF10:   .BYTE   0,0,0,0,0,0
1831  013237  000   000   000
1832  013242 005015 053523 020122 MSGSWR: .ASCIZ  <15><12>/SWR = /
1833  013250 020075  000
1834         013254                   .EVEN
1835                                  .SBTTL  GETSWR SUBROUTINE
1836
1837  013254 022737 000176 001140 GETSWR: CMP     #SWREG,SWR              ;TEST IF REAL SWR
1838  013262 001415               BEQ     1$                      ;BR IF NOT
1839  013264 104401 013272        TYPE    ,65$                ;;TYPE ASCIZ STRING
1840  013270 000410               BR      64$                 ;;GET OVER THE ASCIZ
1841                              ;;65$:  .ASCIZ  <15><12>/DEPRESS CONT./
1842  013312                      64$:
1843  013312 000000               HALT                            ;WAIT FOR OPERATOR
1844  013314 000207               RTS     PC                      ;EXIT
1845  013316 104401               1$:     TYPE
1846  013320 013242               MSGSWR
1847  013322 104411               RDOCT
1848  013324 012677 165610        MOV     (SP)+,@SWR              ;SAVE VALUE TYPED
1849  013330 000207               RTS     PC                      ;EXIT
1850
1851                              .SBTTL  SCOPE HANDLER ROUTINE
1852
1853                              ;;**************************************************************
1854                              ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1855                              ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1856                              ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
```

```
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*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

013332                                  $SCOPE:
013332  104406                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
013334  104406                                 CKSWR
013336  032777  040000  165574          1$:    BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
013344  001114                                 BNE     $OVER           ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
013346  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
013350  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
013354  012737  013374  000004                  MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
013362  005737  177060                          TST     @#177060        ;;TIME OUT ON XOR?
013366  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013372  000463                                  BR      $SVLAD          ;;GO TO THE NEXT TEST
013374  022626                          5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
013376  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013402  000423                                  BR      7$              ;;LOOP ON THE PRESENT TEST
013404                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
013404  032777  000400  165526                  BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
013412  001404                                  BEQ     2$              ;;BR IF NO
013414  127737  165520  001102                  CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
013422  001465                                  BEQ     $OVER           ;;BR IF YES
013424  105737  001103                  2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
013430  001421                                  BEQ     3$              ;;BR IF NO
013432  123737  001115  001103                  CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
013440  101015                                  BHI     3$              ;;BR IF NO
013442  032777  001000  165470                  BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
013450  001404                                  BEQ     4$              ;;BR IF NO
013452  013737  001110  001106          7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
013460  000446                                  BR      $OVER
013462  105037  001103                  4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
013466  005037  001166                          CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
013472  000415                                  BR      1$              ;;ESCAPE TO THE NEXT TEST
013474  032777  004000  165436          3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
013502  001011                                  BNE     1$              ;;BR IF YES
013504  005737  001204                          TST     $PASS           ;;IF FIRST PASS OF PROGRAM
013510  001406                                  BEQ     1$              ;;    INHIBIT ITERATIONS
013512  005237  001104                          INC     $ICNT           ;;INCREMENT ITERATION COUNT
013516  023737  001166  001104                  CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
013524  002024                                  BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
013526  012737  000001  001104          1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
013534  013737  013612  001166                  MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
013542  105237  001102                  $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
013546  113737  001102  001202                  MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
013554  011637  001106                          MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
013560  011637  001110                          MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
013564  005037  001170                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
```

```
1911  013570  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1912  013576  013777  001102  165336  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
1913  013604  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
1914  013610  000002                          RTI                     ;;FIXES PS
1915  013612  001750                  $MXCNT: 1000.                   ;;MAX. NUMBER OF ITERATIONS
1916                                  .SBTTL  ERROR HANDLER ROUTINE
1917
1918                                  ;;**********************************************************************
1919                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1920                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1921                                  ;*AND GO TO $ERRTYP ON ERROR
1922                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1923                                  ;*SW15=1         HALT ON ERROR
1924                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
1925                                  ;*SW09=1         LOOP ON ERROR
1926                                  ;*CALL
1927                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1928
1929  013614                  $ERROR:
1930  013614  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
1931  013616  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
1932  013622  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
1933  013624  013777  001102  165310          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
1934  013632  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
1935  013636  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
1936  013642  162737  000002  001116          SUB     #2,$ERRPC
1937  013650  117737  165242  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
1938  013656  032777  020000  165254          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
1939  013664  001004                          BNE     20$             ;;SKIP TYPEOUTS
1940  013666  004737  014656                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
1941  013672  104401  001173                  TYPE    ,$CRLF
1942  013676                          20$:
1943  013676  122737  000001  001216          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
1944  013704  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
1945  013706  113737  001114  013720          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
1946  013714  004737  015760                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
1947  013720     000          21$:    .BYTE   0
1948  013721     000                  .BYTE   0
1949  013722  000777          22$:    BR      22$             ;;APT ERROR LOOP
1950  013724  005777  165210  2$:     TST     @SWR            ;;HALT ON ERROR
1951  013730  100002                  BPL     3$              ;;SKIP IF CONTINUE
1952  013732  000000                  HALT                    ;;HALT ON ERROR!
1953  013734  104406                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
1954  013736  032777  001000  165174  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
1955  013744  001402                  BEQ     4$              ;;BR IF NO
1956  013746  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
1957  013752  005737  001170          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
1958  013756  001402                  BEQ     5$              ;;BR IF NONE
1959  013760  013716  001170                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
1960  013764                          5$:
1961  013764  022737  012324  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
1962  013772  001001                  BNE     6$              ;;BRANCH IF NO
1963  013774  000000                  HALT                    ;;YES
1964  013776                          6$:
```

```
                                        RTI                             ;;RETURN
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                           ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT

016646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
000004  000002                  MOV     4(SP),2(SP)             ;;INPUT NUMBER
010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
104410                  1$:     RDLIN                           ;;READ AN ASCIZ LINE
012600                          MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
005001                          CLR     R1                      ;;CLEAR DATA WORD
005002                          CLR     R2
112046                  2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
001412                          BEQ     3$                      ;;IF ZERO GET OUT
006301                          ASL     R1                      ;;*2
006102                          ROL     R2
006301                          ASL     R1                      ;;*4
006102                          ROL     R2
006301                          ASL     R1                      ;;*8
006102                          ROL     R2
042716  177770                  BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
062601                          ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
000764                          BR      2$                      ;;LOOP
005726                  3$:     TST     (SP)+                   ;;CLEAN TERmINATOR FROM STACK
010166  000012                  MOV     R1,12(SP)               ;;SAVE THE RESULT
010237  014100                  MOV     R2,$HIOCT
012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
000002                          RTI                             ;;RETURN
000000                  $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
                                .SBTTL  TTY INPUT ROUTINE

                                ;;**********************************************************************
                                .ENABL  LSB

                                ;;**********************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TES! FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
022737  000176  001140  $CKSWR: CMP     #SWREG,SWR              ;;IS THE SOFT-SWR SELECTED?
001074                          BNE     15$                     ;;BRANCH IF NO
105777  165026                  TSTB    @$TKS                   ;;CHAR THERE?
100071                          BPL     15$                     ;;IF NO, DON'T WAIT AROUND
117746  165022                  MOVB    @$TKB,-(SP)             ;;SAVE THE CHAR
```

```
2019  014124  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
2020  014130  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
2021  014134  001062                          BNE     15$             ;;NO, RETURN TO USER
2022  014136  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
2023  014144  001456                          BEQ     15$             ;;BRANCH IF YES
2024
2025  014146  104401  014627                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2026  014152  104401  014634          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2027  014156  013746  000176                  MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
2028  014162  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2029  014164  104401  014645                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2030  014170  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
2031  014172  005046                          CLR     -(SP)           ;;THE NEW SWR
2032  014174  105777  164744          7$:     TSTB    @$TKS           ;;CHAR THERE?
2033  014200  100375                          BPL     7$              ;;IF NOT TRY AGAIN
2034
2035  014202  117746  164740                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2036  014206  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2037
2038
2039
2040  014212  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
2041  014216  001005                          BNE     10$             ;;BRANCH IF NOT
2042  014220  104401  014622                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
2043  014224  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2044  014230  000757                          BR      19$             ;;LET'S TRY IT AGAIN
2045
2046
2047  014232  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
2048  014236  001022                          BNE     16$             ;;BRANCH IF NO
2049  014240  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2050  014244  001403                          BEQ     11$             ;;BRANCH IF YES
2051  014246  016677  000002  164664          MOV     2(SP),@SWR      ;;SAVE NEW SWR
2052  014254  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
2053  014260  104401  001173          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
2054  014264  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2055  014272  001003                          BNE     15$             ;;BRANCH IF NOT
2056  014274  012777  000100  164642          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2057  014302  000002                  15$:    RTI                     ;;RETURN
2058  014304  004737  015672          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2059  014310  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
2060  014314  002420                          BLT     18$             ;;BRANCH IF YES
2061  014316  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
2062  014322  003015                          BGT     18$             ;;BRANCH IF YES
2063  014324  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2064  014330  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
2065  014334  001403                          BEQ     17$             ;;BRANCH IF YES
2066  014336  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
2067  014340  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
2068  014342  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
2069  014344  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2070  014350  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2071  014354  000707                          BR      7$              ;;GET THE NEXT ONE
2072  014356  104401  001172          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
```

```
2073  014362  000720                          BR      20$             ;;SIMULATE CONTROL-U
2074                                  .DSABL  LSB
2075
2076
2077                                  ;;*****************************************************************
2078                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2079                                  ;*CALL:
2080                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2081                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2082                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
2083                                  ;
2084
2085  014364  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2086  014366  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
2087  014374  105777  164544          1$:     TSTB    @$TKS           ;;WAIT FOR
2088  014400  100375                          BPL     1$              ;;A CHARACTER
2089  014402  117766  164540  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
2090  014410  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2091  014416  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
2092  014424  001013                          BNE     3$              ;;BRANCH IF NO
2093  014426  105777  164512          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
2094  014432  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
2095  014434  117746  164506                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
2096  014440  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2097  014444  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2098  014450  001366                          BNE     2$              ;;IF NOT DISCARD IT
2099  014452  000750                          BR      1$              ;;YES, RESUME
2100  014454  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
2101  014462  002407                          BLT     4$              ;;BRANCH IF YES
2102  014464  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2103  014472  003003                          BGT     4$              ;;BRANCH IF YES
2104  014474  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2105  014502  000002                  4$:     RTI                     ;;GO BACK TO USER
2106                                  ;;*****************************************************************
2107                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2108                                  ;*CALL:
2109                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2110                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2111                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2112
2113  014504  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2114  014506  012703  014612          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2115  014512  022703  014622          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
2116  014516  101405                          BLOS    4$              ;;BR IF YES
2117  014520  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2118  014522  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2119  014524  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2120  014530  001003                          BNE     3$              ;;SKIP IF NOT
2121  014532  104401  001172          4$:     TYPE    ,$QUES          ;;TYPE A '?'
2122  014536  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2123  014540  111337  014610          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2124  014544  104401  014610                  TYPE    ,9$
2125  014550  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2126  014554  001356                          BNE     2$              ;;LOOP IF NOT RETURN
```

```
2127  014556  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2128  014562  104401  001174                    TYPE    ,$LF            ;;TYPE A LINE FEED
2129  014566  012603                            MOV     (SP)+,R3        ;;RESTORE R3
2130  014570  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2131  014572  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2132  014600  012766  014612  000004            MOV     #$TTYIN,4(SP)
2133  014606  000002                            RTI                     ;;RETURN
2134  014610     000                  9$:       .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2135  014611     000                            .BYTE   0               ;;TERMINATOR
2136  014612  000010                  $TTYIN:   .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2137  014622  052536  005015     000  $CNTLU:   .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2138  014627     136  006507  000012  $CNTLG:   .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2139  014634  005015  053523  020122  $MSWR:    .ASCIZ  <15><12>/SWR = /
2140  014642  020075     000
2141  014645     040  047040  053505  $MNEW:    .ASCIZ  /  NEW = /
2142  014652  036440  000040
2143
2144                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2145
2146                                  ;;**************************************************************
2147                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2148                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2149                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2150
2151  014656                  $ERRTYP:
2152  014656  104401  001173            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2153  014662  010046                    MOV     R0,-(SP)        ;;SAVE R0
2154  014664  005000                    CLR     R0              ;;PICKUP THE ITEM INDEX
2155  014666  153700  001114            BISB    @#$ITEMB,R0
2156  014672  001004                    BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
2157                                                            ;;TYPE THE PC OF THE ERROR
2158  014674  013746  001116            MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
2159                                                            ;;ERROR ADDRESS
2160  014700  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2161  014702  000426                    BR      6$              ;;GET OUT
2162  014704  005300          1$:       DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
2163  014706  006300                    ASL     R0              ;;      WORK FOR THE ERROR TABLE
2164  014710  006300                    ASL     R0
2165  014712  006300                    ASL     R0
2166  014714  062700  001322            ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
2167  014720  012037  014730            MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
2168  014724  001404                    BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
2169  014726  104401                    TYPE                    ;;TYPE THE "ERROR MESSAGE"
2170  014730  000000          2$:       .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
2171  014732  104401  001173            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2172  014736  012037  014746  3$:       MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
2173  014742  001404                    BEQ     5$              ;;SKIP TYPEOUT IF 0
2174  014744  104401                    TYPE                    ;;TYPE THE "DATA HEADER"
2175  014746  000000          4$:       .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
2176  014750  104401  001173            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2177  014754  011000          5$:       MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
2178  014756  001004                    BNE     7$              ;;GO TYPE THE DATA
2179  014760  012600          6$:       MOV     (SP)+,R0        ;;RESTORE R0
2180  014762  104401  001173            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
```

```
                                        RTS     PC              ;;RETURN
                                7$:
                                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
                                        BEQ     6$              ;;BR IF NO
                                        TYPE    ,8$             ;;TYPE TWO(2) SPACES
                                        BR      7$              ;;LOOP
                                8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                                        .EVEN

                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

015012  012646  000000          $TYpOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
015016  116637  000001  015235          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
015024  112637  015237                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
015030  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
015034  000406                          BR      $TYpON
015036  112737  000001  015235  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
015044  112737  000006  015237          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
015052  112737  000005  015234  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
015060  010346                          MOV     R3,-(SP)        ;;SAVE R3
015062  010446                          MOV     R4,-(SP)        ;;SAVE R4
015064  010546                          MOV     R5,-(SP)        ;;SAVE R5
015066  113704  015237                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
015072  005404                          NEG     R4
015074  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
015100  110437  015236                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
015104  113704  015235                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
015110  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
015114  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
```

```
2235  015116  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2236  015120  000404                          BR      3$              ;;GO DO MSB
2237  015122  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2238  015124  006105                          ROL     R5
2239  015126  006105                          ROL     R5
2240  015130  010503                          MOV     R5,R3
2241  015132  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2242  015134  105337  015236                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2243  015140  100016                          BPL     7$              ;;BR IF NO
2244  015142  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2245  015146  001002                          BNE     4$              ;;TEST FOR 0
2246  015150  005704                          TST     R4              ;;SUPPRESS THIS 0?
2247  015152  001403                          BEQ     5$              ;;BR IF YES
2248  015154  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2249  015156  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2250  015162  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2251  015166  110337  015232                  MOVB    R3,8$           ;;SAVE FOR TYPING
2252  015172  104401  015232                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2253  015176  105337  015234          7$:     DECB    $OCNT           ;;COUNT BY 1
2254  015202  003347                          BGT     2$              ;;BR IF MORE TO DO
2255  015204  002402                          BLT     6$              ;;BR IF DONE
2256  015206  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2257  015210  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2258  015212  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2259  015214  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2260  015216  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2261  015220  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2262  015226  012616                          MOV     (SP)+,(SP)
2263  015230  000002                          RTI                     ;;RETURN
2264  015232     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2265  015233     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2266  015234     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2267  015235     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2268  015236  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2269
2270                                  .SBTTL  POWER DOWN AND UP ROUTINES
2271
2272                                  ;;**************************************************************
2273                                  ;POWER DOWN ROUTINE
2274  015240  012737  015404  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2275  015246  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2276  015254  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2277  015256  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2278  015260  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2279  015262  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2280  015264  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2281  015266  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2282  015270  017746  163644                  MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
2283  015274  010637  015410                  MOV     SP,$SAVR6       ;;SAVE SP
2284  015300  012737  015312  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2285  015306  000000                          HALT
2286  015310  000776                          BR      .-2             ;;HANG UP
2287
2288                                  ;;**************************************************************
```

```
                                        ;POWER UP ROUTINE
015312  012737  015404  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWn
015320  013706  015410                  MOV     $SAVR6,SP        ;;GET SP
015324  005037  015410                  CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
015330  005237  015410          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
015334  001375                          BNE     1$               ;;OF  WORD
015336  012677  163576                  MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
015342  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
015344  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
015346  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
015350  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
015352  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
015354  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
015356  012737  015240  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
015364  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
015372  104401                          TYPE                     ;;REPORT THE POWER FAILURE
015374  015412                  $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
015376  012716                          MOV     (PC)+,(SP)       ;;RESTART AT IOTEST
015400  002662                  $PWRAD: .WORD   IOTEST           ;;RESTART ADDRESS
015402  000002                          RTI
015404  000000                  $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
015406  000776                          BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
015410  000000                  $SAVR6: 0                        ;;PUT THE SP HERE
015412  005015  042522  052123 PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
015420  051101  044524  043516
015426  040440  052106  051105
015434  040440  050040  053517
015442  051105  043040  044501
015450  052514  042522  005015
015456     000
        015460                          .EVEN

                                .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
                                ;*

015460  105737  001157  $TYpE:  TSTB    $TPFLG           ;;IS THERE A TERMINAL?
015464  100002                  BPL     1$               ;;BR IF YES
015466  000000                  HALT                     ;;HALT HERE IF NO TERMINAL
015470  000430                  BR      3$               ;;LEAVE
015472  010046          1$:     MOV     R0,-(SP)         ;;SAVE R0
```

```
2343  015474  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2344  015500  122737  000001  001216      CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
2345  015506  001011                      BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
2346  015510  132737  000100  001217      BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
2347  015516  001405                      BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
2348  015520  010037  015530              MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
2349  015524  004737  015750              JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
2350  015530  000000            61$:      .WORD   0               ;;MESSAGE ADDRESS
2351  015532  132737  000040  001217 62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
2352  015540  001003                      BNE     60$             ;;YES,SKIP TYPE OUT
2353  015542  112046                2$:   MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2354  015544  001005                      BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2355  015546  005726                      TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2356  015550  012600                60$:  MOV     (SP)+,R0        ;;RESTORE R0
2357  015552  062716  000002        3$:   ADD     #2,(SP)         ;;ADJUST RETURN PC
2358  015556  000002                      RTI                     ;;RETURN
2359  015560  122716  000011        4$:   CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2360  015564  001430                      BEQ     8$
2361  015566  122716  000200              CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2362  015572  001006                      BNE     5$
2363  015574  005726                      TST     (SP)+           ;;POP  <CR><LF> EQUIV
2364  015576  104401                      TYPE                    ;;TYPE A CR AND LF
2365  015600  001173                      $CRLF
2366  015602  105037  015736              CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2367  015606  000755                      BR      2$              ;;GET NEXT CHARACTER
2368  015610  004737  015672        5$:   JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2369  015614  123726  001156        6$:   CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2370  015620  001350                      BNE     2$              ;;IF NO GO GET NEXT CHAR.
2371  015622  013746  001154              MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2372                                                              ;;AND THE NULL CHAR.
2373  015626  105366  000001        7$:   DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2374  015632  002770                      BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2375  015634  004737  015672              JSR     PC,$TYPEC       ;;GO TYPE A NULL
2376  015640  105337  015736              DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2377  015644  000770                      BR      7$              ;;LOOP
2378
2379                                ;HORIZONTAL TAB PROCESSOR
2380
2381  015646  112716  000040        8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2382  015652  004737  015672        9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
2383  015656  132737  000007  015736      BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2384  015664  001372                      BNE     9$              ;;TAB STOP
2385  015666  005726                      TST     (SP)+           ;;POP SPACE OFF STACK
2386  015670  000724                      BR      2$              ;;GET NEXT CHARACTER
2387  015672  105777  163252        $TYPEC: TSTB  @$TPS           ;;WAIT UNTIL PRINTER IS READY
2388  015676  100375                      BPL     $TYPEC
2389  015700  116677  000002  163244      MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2390  015706  122766  000015  000002      CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2391  015714  001003                      BNE     1$              ;;BRANCH IF NO
2392  015716  105037  015736              CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2393  015722  000406                      BR      $TYPEX          ;;EXIT
2394  015724  122766  000012  000002 1$:  CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2395  015732  001402                      BEQ     $TYPEX          ;;BRANCH IF YES
2396  015734  105227                      INCB    (PC)+           ;;COUNT THE CHARACTER
```

```
2397  015736  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2398  015740  000207                  $TYPEX: RTS     PC
2399
2400
2401                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2402
2403                                  ;;**************************************************************************
2404  015742  112737  000001  016206  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2405  015750  112737  000001  016204  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2406  015756  000403                          BR      $ATYC
2407  015760  112737  000001  016206  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2408  015766                          $ATYC:
2409  015766  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2410  015770  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2411  015772  105737  016204                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2412  015776  001450                          BEQ     5$              ;;IF NOT:  BR
2413  016000  122737  000001  001216          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2414  016006  001031                          BNE     3$              ;;IF NOT:  BR
2415  016010  132737  000100  001217          BITB    #APTSPOOL,$ENVm ;;SHOULD SPOOL MESSAGES?
2416  016016  001425                          BEQ     3$              ;;IF NOT:  BR
2417  016020  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2418  016024  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2419  016032  005737  001176          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2420  016036  001375                          BNE     1$              ;;IF NOT:  WAIT
2421  016040  010037  001212                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2422  016044  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2423  016046  001376                          BNE     2$
2424  016050  163700  001212                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2425  016054  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2426  016056  010037  001214                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2427  016062  012737  000004  001176          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2428  016070  000413                          BR      5$
2429  016072  017637  000004  016116  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2430  016100  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2431  016106  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2432  016112  004737  015460                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
2433  016116  000000                  4$:     .WORD   0
2434  016120                          5$:
2435  016120  105737  016206          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2436  016124  001416                          BEQ     12$             ;;IF NOT:  BR
2437  016126  005737  001216                  TST     $ENv            ;;RUNNING UNDER APT?
2438  016132  001413                          BEQ     12$             ;;IF NOT:  BR
2439  016134  005737  001176          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2440  016140  001375                          BNE     11$             ;;IF NOT:  WAIT
2441  016142  017637  000004  001200          MOV     @4(SP),$FATAL   ;;GET ERROR #
2442  016150  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2443  016156  005237  001176                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2444  016162  105037  016206          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2445  016166  105037  016205                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2446  016172  105037  016204                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2447  016176  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2448  016200  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2449  016202  000207                          RTS     PC              ;;RETURN
2450  016204     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
```

```
2451  016205  000                     $LFLG:  .BYTE   0               ;;LOG FLAG
2452  016206  000                     $FFLG:  .BYTE   0               ;;FATAL FLAG
2453          016210                          .EVEN
2454          000200                  APTSIZE=200
2455          000001                  APTENV=001
2456          000100                  APTSPOOL=100
2457          000040                  APTCSUP=040
2458
2459                                  .SBTTL  TRAP DECODER
2460
2461                                  ;;**************************************************************
2462                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2463                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2464                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2465                                  ;*GO TO THAT ROUTINE.
2466
2467  016210  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2468  016212  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
2469  016216  005740                          TST     -(R0)           ;;BACKUP BY 2
2470  016220  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2471  016222  006300                          ASL     R0              ;;POSITION FOR INDEXING
2472  016224  016000  016244                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2473  016230  000200                          RTS     R0              ;;GO TO ROUTINE
2474
2475
2476                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2477
2478  016232  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2479  016234  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWn
2480  016242  000002                          RTI                     ;;RESTORE THE PSW
2481
2482                                  .SBTTL  TRAP TABLE
2483
2484                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2485                                  ;*BY THE "TRAP" INSTRUCTION.
2486
2487                                  ;       ROUTINE
2488                                  ;       -------
2489  016244  016232                  $TRPAD: .WORD   $TRAP2
2490  016246  015460                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2491  016250  015036                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2492  016252  015012                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2493  016254  015052                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2494
2495  016256  014152                  $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
2496
2497  016260  014102                  $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2498  016262  014364                  $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2499  016264  014504                  $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2500  016266  014000                  $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2501
2502                                  ;*
2503                                  ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
2504                                  ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
```

```
                                        ;*NEXT WE WILL INIT BOTH UPROCESSORS
                                        ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
                                        ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
                                        ;*
                                        ;*      CALL=   JSR     R5,$LPAI
                                        ;*              .WORD   0               ;ADDR. OF DEVICE ADDRESS.
                                        ;* ROUTINES REQUIRED:   .LOADLP
                                        ;* PROGRAMS REQUIRED:   DRLPX2
                                        ;*
                                        ;*              ;RETURNS WITH $AERR=1 IF SLAVE
                                        ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
                                        ;*
016270                          $LPAI:
016270  013746  000004                  MOV     4,-(SP)

016274  000413                          BR      31$             ;FIELD DOES NOT HAVE A BUS SWITCH TO
                                                                ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
                                                                ;BRANCH ARROUD THE NEXT CODE THAT
                                                                ;WORKS BASED ON A BUS SWITCH.
                                                                ;CODE LEFT IN HERE FOR IN HOUSE
                                                                ;PERSONAL WHO MAY PATCH THIS BRANCH
                                                                ;INSTRUCTION TO A <NOP> OCTAL <240>
                                                                ;IN ORDER TO RUN PROGRAM WITH A SWITCH.

                                                                ;NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
                                                                ;TEST EQUIPMENT ONLY IT CONNECTS
                                                                ;THE UNIBUS TO THE I/O BUS FOR
                                                                ;CERTAIN TESTING.
016276  012737  016322  000004          MOV     #30$,4
016304  005237  170000                  INC     170000
016310  104401  016316                  TYPE    ,65$            ;;TYPE ASCIZ STRING
016314  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <7>##
016320                          64$:
016320  000401                          BR      31$
016322  022626                  30$:    CMP     (SP)+,(SP)+
016324  012637  000004          31$:    MOV     (SP)+,4         ;ALL THIS JUNK MUST BE REMOVED!!
016330  005037  017146                  CLR     $AERR
016334  004537  017150                  JSR     R5,$LOAD        ;LOAD MICRO-CODE.
016340  000000G                         .WORD   DRLPX2          ;FILE "DRLPX2.OBJ"

016342  052777  040000  163066          BIS     #BIT14,@KMADO   ;ISSUE KMC+DMC INIT.

016350                          1$:
                                                                ;"HANGS" HERE THEN KMC-11 ERROR.
016350  010146                          MOV     R1,-(SP)
016352  005001                          CLR     R1
016354  005201                  2$:     INC     R1              ;STALL FOR DMC-UP
016356  001376                          BNE     2$
016360  012777  104000  163050          MOV     #BIT15!BIT11,@KMADO     ;SET RUN, AND ENABLE ARBITRATION.
016366  105201                  25$:    INCB    R1
016370  001376                          BNE     25$
```

```
2559  016372  032777  000040  163036          BIT     #BIT5,@KMAD0    ;SLAVE READY? (READING IPBM SR)
2560  016400  001401                          BEQ     3$
2561                                                                  ;FATAL LPA-11 ERROR SLAVE NOT READY.
2562  016402  104000                          ERROR
2563
2564  016404  012777  000004  163030  3$:     MOV     #4,@KMAD2       ;READ FAST PATH
2565  016412                          4$:
2566  016412  004537  020060                  JSR     R5, $TOUt       ;-TOUT-CHECK FOR TIMEOUT
2567
2568  016416  104000                          ERROR                   ;/TIME-OUT ERROR
2569                                                                  ;/WE FAILED TO COMPLETE
2570                                                                  ;/CURRENT OPERATION.
2571                                                                  ;/CONTINUES IN THIS LOOP
2572                                                                  ;/WOULD MAKE US "HANG" HERE
2573
2574  016420  000774                          BR              4$
2575
2576                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
2577  016422  122777  000377  163012          CMPB    #377,@KMAD2     ;WAIT TILL KMC DONE COMMAND.
2578  016430  001370                          BNE     4$
2579  016432  122777  000377  163006          CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
2580  016440  001001                          BNE     35$
2581  016442  104000                          ERROR                   ;IPBM ERROR (SLAVE SIDE)
2582                                                                  ;YOU MUST RUN IPBM DIAGNOSTIC.
2583
2584  016444  122777  000004  162774  35$:     CMPB    #4,@KMAD4       ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
2585  016452  001543                          BEQ     5$              ;YES-CONTINUE.
2586  016454  005227  177777                  INC     #-1
2587  016460  001140                          BNE     5$
2588  016462  005227  177777                  INC     #-1
2589  016466  001135                          BNE     5$
2590  016470  104401  016476                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2591  016474  000440                          BR      66$             ;;GET OVER THE ASCIZ
2592                                  ;;67$:   .ASCIZ  <200>"W A R N I N G  THIS PROGRAM WAS DESIGNED TO rUN WITH VERSION 4"
2593  016576                          66$:
2594  016576  104401  016604                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2595  016602  000430                          BR      68$             ;;GET OVER THE ASCIZ
2596                                  ;;69$:   .ASCIZ  <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
2597  016664                          68$:
2598  016664  104401  016672                  TYPE    ,71$            ;;TYPE ASCIZ STRING
2599  016670  000434                          BR      70$             ;;GET OVER THE ASCIZ
2600                                  ;;71$:   .ASCIZ  <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
2601  016762                          70$:
2602
2603  016762  112737  177777  017114  5$:     MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
2604  016770  012501                          MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
2605  016772  021127  000000          6$:     CMP     (R1),#0         ;TERM REACHED?
2606  016776  001444                          BEQ     10$
2607  017000  105237  017114                  INCB    11$
2608  017004  113777  017114  162434          MOVB    11$,@KMAD4      ;FIFO DATA
2609  017012  004737  017116                  JSR     PC,20$          ;ISSUE SEND
2610  017016  112177  162424                  MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
2611  017022  004737  017116                  JSR     PC,20$          ;ISSUE SEND
2612  017026  112177  162414                  MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO sLAVE.
```

```
2613  017032  004737  017116                  JSR     PC,20$
2614
2615  017036  032777  000002  162372  7$:     BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
2616  017044  001374                          BNE     7$              ;=1 NO DATA. =0 DATA.
2617  017046  112777  000002  162366          MOVB    #2,@KMAD2       ;READ FIFO.
2618
2619  017054                          8$:
2620  017054  004537  020060                  JSR     R5, $TOUt       ;-TOUT-CHECK FOR TIMEOUT
2621
2622  017060  104000                          ERROR                   ;/TIME-OUT ERROR
2623                                                                  ;/WE FAILED TO COMPLETE
2624                                                                  ;/CURRENT OPERATION.
2625                                                                  ;/CONTINUES IN THIS LOOP
2626                                                                  ;/WOULD MAKE US "HANG" HERE
2627
2628  017062  000774                          BR              8$
2629
2630                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
2631  017064  122777  000377  162350          CMPB    #377,@KMAD2     ;WAIT FOR READ.
2632  017072  001370                          BNE     8$
2633  017074  105777  162346                  TSTB    @KMAD4          ;WAS A ZERO RETURNED?
2634  017100  001734                          BEQ     6$              ;YES GET NEXT ADDR.
2635                                                                  ;SLAVE WILL RETURN CODE 0 IF
2636  017102  005237  017146                  INC     $AERR           ;DEV PRESENT.   ELSE
2637                                                                  ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
2638  017106  005041                          CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
2639  017110  012601                  10$:    MOV     (SP)+,R1
2640  017112  000205                          RTS     R5              ;RETURN ALL ADDR. CHECKED.
2641
2642  017114  000000                  11$:    .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
2643                                                                  ;TO SLAVES ADDR. TABLE.
2644
2645  017116  112777  000003  162316  20$:    MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
2646  017124                          21$:
2647  017124  004537  020060                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2648
2649  017130  104000                          ERROR                   ;/TIME-OUT ERROR
2650                                                                  ;/WE FAILED TO COMPLETE
2651                                                                  ;/CURRENT OPERATION.
2652                                                                  ;/CONTINUES IN THIS LOOP
2653                                                                  ;/WOULD MAKE US "HANG" HERE
2654
2655  017132  000774                          BR              21$
2656
2657                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
2658  017134  122777  000377  162300          CMPB    #377,@KMAD2     ;KMC CODE WILL RETURN A "377"
2659  017142  001370                          BNE     21$             ;WHEN DONE COMMAND.
2660  017144  000207                          RTS     PC
2661
2662  017146  000000                  $AERR:  .WORD   0               ;=0 IF ADDR. LIST OK,=1 IF BAD.
2663
2664                                          ;*
2665                                          ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
2666                                          ;*      CALL =  JSR     R5,$LOAD
```

```
                                        ;*              .WORD   XX              ;ADDR. OF MICRO CODE.
                                        ;*              ;RETURNS HERE
                                        ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
                                        ;*
017150  010446                  $LOAD:  MOV     R4,-(SP)        ;SAVE R4.
017152  010046                          MOV     R0,-(SP)        ;SAVE R0.
017154  012500                  1$:     MOV     (5)+,R0         ;GET PROG. ADDR.
017156  005077  162254                  CLR     @KMADD          ;CLEAR CSR
017162  005077  162260                  CLR     @KMAD4          ;CLEAR CRAM ADDR.
017166  052777  002000  162242  2$:     BIS     #2000,@KMADD    ;SELECT CRAM.
017174  012077  162252                  MOV     (0)+,@KMAD6     ;WRITE DATA.
017200  052777  020000  162230          BIS     #20000,@KMADD   ;SET CRAM WRITE
017206  005077  162224                  CLR     @KMADD          ;DISABLE CRAM.
017212  005277  162230                  INC     @KMAD4          ;UPDATE CRAM ADDR.
017216  021027  177777                  CMP     (0),#-1         ;ALL DONE?
017222  001361                          BNE     2$              ;NO LOOP.
017224  005077  162216                  CLR     @KMAD4          ;CLEAR CRAM ADDR.
017230  016500  177776                  MOV     -2(5),R0        ;GET MICRO CODE ADDR.

017234  052777  002000  162174  3$:     BIS     #2000,@KMADD    ;SELECT CRAM
017242  022077  162204                  CMP     (R0)+,@KMAD6    ;DATA OK?
017246  001013                          BNE     5$              ;NO - REPORT AN ERROR.
017250  021027  177777                  CMP     (0),#-1         ;ALL DONE?
017254  001405                          BEQ     4$              ;YES - EXIT
017256  005077  162154                  CLR     @KMADD          ;NO - DESELECT CRAM.
017262  005277  162160                  INC     @KMAD4          ;UPDATE CRAM ADDR.
017266  000762                          BR      3$

017270  012600                  4$:     MOV     (SP)+,R0        ;RESTORE R0
017272  012604                          MOV     (SP)+,R4        ;RESTORE R4
017274  000205                          RTS     R5              ;EXIT

017276                          5$:                             ;COME HERE ON LOAD ERROR
017276  005745                          TST     -(5)
017300  105204                          INCB    R4              ;UPDATE ERROR COUNTER.
017302  100324                          BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
017304  000000                          HALT                    ;MICRO CODE LOAD ERROR.
                                                                ;KMC-11 FAULT. YOU COULD TRY
017306  000722                          BR      1$              ;TO PRESS CONTINUE TO GIVE IT
                                                                ;ANOTHER CHANCE, BUT I DOUBT
                                                                ;THAT THAT WOULD WORK. SINCE I'VE
                                                                ;ALREADY GIVEN IT 177 (OCTAL) CHANCES.
                                                                ;TRY RUNNING THE KMC-11 DIAGNOSTIC.


                                        ;
                                        ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
                                        ;*
                                        ;*      CALL =  JSR     R5,$TLKW
                                        ;*              .WORD   0               ;OFFSET OF DEVICE ADDR.
                                        ;*              .WORD   0               ;DATA TO BE WRITTEN
                                        ;*
017310  010046                  $TLKW:  MOV     R0,-(SP)        ;SAVE R0
```

```
2721  017312  012500                          MOV     (5)+,R0         ;GET DEVICE OFFSET
2722  017314  052700  000340                  BIS     #340,R0         ;ADD WRITE CODE.
2723  017320  004737  017572                  JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
2724  017324  010037  017416                  MOV     R0,W1
2725  017330  010077  162112                  MOV     R0,@KMAD4
2726  017334  112777  000005  162100          MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
2727  017342  004737  017572                  JSR     PC,$LPW         ;WAIT FOR RDY
2728  017346  011537  017420                  MOV     (5),W2
2729  017352  112577  162070                  MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
2730
2731  017356  112777  000005  162056          MOVB    #5,@KMAD2       ;FP WRITE
2732  017364  004737  017572                  JSR     PC,$LPW
2733  017370  111537  017422                  MOVB    (5),W3
2734  017374  112577  162046                  MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
2735  017400  112777  000005  162034          MOVB    #5,@KMAD2
2736  017406  004737  017572                  JSR     PC,$LPW
2737  017412  012600                          MOV     (SP)+,R0
2738  017414  000205                          RTS     R5              ;EXIT DONE.
2739  017416  000000                  W1:     0
2740  017420  000000                  W2:     0
2741  017422  000000                  W3:     0
2742
2743                                          ;*
2744                                          ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
2745                                          ;*
2746                                          ;*      CALL =  JSR     R5,$TLKR
2747                                          ;*              .WORD   0               ;OFFSET OF DEVICE
2748                                          ;*              .RETURNS HERE
2749                                          ;*DATA IN WORD $DATR
2750                                          ;*
2751
2752  017424  010046          $TLKR:  MOV     R0,-(SP)        ;SAVE R0
2753  017426  012500                  MOV     (5)+,R0         ;GET OFFSET
2754  017430  052700  000300          BIS     #300,R0         ;ADD READ CODE
2755  017434  004737  017572          JSR     PC,$LPW         ;WAIT TILL READY
2756  017440  110077  162002          MOVB    R0,@KMAD4
2757  017444  112777  000005  161770  MOVB    #5,@KMAD2       ;ISSUE WRITE FP
2758  017452  004737  017572          JSR     PC,$LPW
2759  017456  010037  017566          MOV     R0,RD1
2760  017462                  1$:
2761  017462  004537  020060          JSR     R5, $TOUt       ;-TOUT-CHECK FOR TIMEOUT
2762
2763  017466  104000                  ERROR                   ;/TIME-OUT ERROR
2764                                          ;/WE FAILED TO COMPLETE
2765                                          ;/CURRENT OPERATION.
2766                                          ;/CONTINUES IN THIS LOOP
2767                                          ;/WOULD MAKE US "HANG" HERE
2768
2769  017470  000774                  BR              1$
2770
2771                                          ;/RETURNS HERE-FROM-TIMED OUT.
2772  017472  032777  000040  161736  BIT     #BIT5,@KMAD0    ;FAST PATH GOT DATA?
2773  017500  001370                  BNE     1$
2774  017502  112777  000004  161732  MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
```

```
2775  017510  004737  017572                      JSR     PC,$LPW
2776  017514  117737  161726  017570              MOVB    @KMAD4,$DATR    ;GET LOW BYTE
2777  017522                          2$:
2778  017522  004537  020060                      JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2779
2780  017526  104000                              ERROR                   ;/TIME-OUT ERROR
2781                                                                      ;/WE FAILED TO COMPLETE
2782                                                                      ;/CURRENT OPERATION.
2783                                                                      ;/CONTINUES IN THIS LOOP
2784                                                                      ;/WOULD MAKE US "HANG" HERE
2785
2786  017530  000774                              BR              2$
2787
2788                                                                      ;/RETURNS HERE-FROM-TIMED OUT.
2789  017532  032777  000040  161676              BIT     #BIT5,@KMAD0    ;FAST PATH READY?
2790  017540  001370                              BNE     2$
2791  017542  112777  000004  161672              MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
2792  017550  004737  017572                      JSR     PC,$LPW
2793  017554  117737  161666  017571              MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
2794  017562  012600                              MOV     (SP)+,R0
2795  017564  000205                              RTS     R5
2796  017566  000000                  RD1:        0
2797  017570  000000                  $DATR:      .WORD   0
2798                                              ;
2799                                              ;THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
2800                                              ;AS FAST PATH TO BE READ.
2801                                              ;
2802                                              ;       CALL =  JSR     PC,$LPW
2803                                              ;
2804                                              ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
2805                                              ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
2806                                              ;
2807
2808  017572  010146                  $LPW:       MOV     R1,-(SP)        ;SAVE R1
2809  017574  005001                              CLR     R1
2810  017576  122777  000377  161636  1$:         CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
2811  017604  001403                              BEQ     2$
2812  017606  005201                              INC     R1              ;TIME OUT?
2813  017610  001372                              BNE     1$
2814  017612  000411                              BR      10$
2815
2816  017614  032777  000020  161614  2$:         BIT     #BIT4,@KMAD0    ;FAST PATH READ?
2817  017622  001403                              BEQ     3$
2818  017624  005201                              INC     R1              ;NO - TIME OUT?
2819  017626  001372                              BNE     2$
2820  017630  000402                              BR      10$             ;YES - REPORT AN ERROR
2821
2822  017632  012601                  3$:         MOV     (SP)+,R1        ;RESTORE R1
2823  017634  000207                              RTS     PC              ;EXIT
2824
2825  017636                          10$:
2826  017636  104401  017644                      TYPE    ,65$            ;;TYPE ASCIZ STRING
2827  017642  000407                              BR      64$             ;;GET OVER THE ASCIZ
2828                                  ;;65$:      .ASCIZ  <200>#LPA-11 FAULT#
```

```
2829  017662                          64$:
2830
2831  017662  000000                  11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
2832  017664  000776                          BR      11$             ;DIAGNOSTICS.
2833
2834
2835
2836                                          ;*
2837                                          ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
2838                                          ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
2839                                          ;*
2840                                          ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
2841                                          ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
2842                                          ;* THAT ADDRESS.
2843                                          ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
2844                                          ;* $TLKW
2845                                          ;*
2846
2847  017666  010046                  $OUtLP: MOV     R0,-(SP)        ;SAVE R0
2848  017670  010146                          MOV     R1,-(SP)        ;SAVE R1
2849
2850  017672  012700  001464                  MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
2851  017676  005001                          CLR     R1
2852  017700  005710                  1$:     TST     (0)             ;TERMINATOR REACHED?
2853  017702  001421                          BEQ     10$             ;YES NEXT STEP.
2854  017704  027520  000000                  CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
2855  017710  001402                          BEQ     2$
2856  017712  005201                          INC     R1
2857  017714  000771                          BR      1$
2858
2859  017716  010137  017734          2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
2860  017722  005725                          TST     (5)+
2861  017724  013537  017736                  MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
2862  017730  004537  017310                  JSR     R5,$TLKW        ;DO WRITE
2863  017734  000000                  3$:     .WORD   0               ;DEVICE OFFSET
2864  017736  000000                  4$:     .WORD   0               ;DATA TO BE WRITTEN.
2865  017740  012601                          MOV     (SP)+,R1
2866  017742  012600                          MOV     (SP)+,R0
2867  017744  000205                          RTS     R5
2868  017746  017520  000000          10$:    MOV     @(5),(0)+       ;SAVE ADDR.
2869  017752  005010                          CLR     (0)
2870  017754  004537  016270                  JSR     R5,$LPAI
2871  017760  001464                          .WORD   .DVLS
2872  017762  000755                          BR      2$
2873
2874                                          ;*
2875                                          ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
2876                                          ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
2877                                          ;*
2878                                          ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
2879                                          ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
2880                                          ;*WITH THE NEW ADDR.
2881                                          ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
2882                                          ;*$TLKr
```

```
                                        ;*      CALL THROUGH    MOVEI   DATA,ADDR.
                                        ;*              WHICH EQUALS:
                                        ;*                      JSR     R5,$INLP
                                        ;*                      .WORD   XX      ADDR OF DEVICE
                                        ;*                      .WORD   YY      ADDR TO STORE READ DATA.
017764  010046                  $INLP:  MOV     R0,-(SP)        ;SAVE R0
017766  010146                          MOV     R1,-(SP)        ;SAVE R1
017770  012700  001464                  MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
017774  005001                          CLR     R1
017776  005710                  1$:     TST     (0)             ;EOL REACHED?
020000  001420                          BEQ     10$             ;YES - DEFINE NEW ADDR.
020002  027520  000000                  CMP     @(5),(0)+       ;ADDR. MATCH?
020006  001402                          BEQ     2$
020010  005201                          INC     R1
020012  000771                          BR      1$
020014  010137  020026          2$:     MOV     R1,3$           ;SAVE LIST OFFSET
020020  005725                          TST     (5)+
020022  004537  017424                  JSR     R5,$TLKr        ;GO READ DEVICE
                020026          $OFS=.
020026  000000                  3$:     .WORD   0               ;OFFSET OF DEVICE
020030  013735  017570                  MOV     $DATR,@(5)+     ;STORE DATA.
020034  012601                          MOV     (SP)+,R1        ;RESTORE R1
020036  012600                          MOV     (SP)+,R0        ;RESTORE R2
020040  000205                          RTS     R5              ;EXIT
020042  017520  000000          10$:    MOV     @(5),(0)+
020046  005010                          CLR     (0)
020050  004537  016270                  JSR     R5,$LPAI
020054  001464                          .WORD   .DVLS
020056  000756                          BR      2$
                                        ;*
                                        ;*$TOUT ROUTINE USED TO WATCH IF
                                        ;*      WE'RE IN A LOOP TOO-LONG
                                        ;*      CALL=   JSR R5, $TOUT
                                        ;*              ERROR X     ;RETURNS HERE ON TIMEOUT
                                        ;*              BR
                                        ;*              ;RETURNS HERE NO ERROR
                                        ;*
020060  020537  020114          $TOUt:  CMP     R5,$SAD         ;SAME ADDR?
020064  001405                          BEQ     1$
020066  010537  020114                  MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
020072  005037  020116                  CLR     $CNT            ;CLR CNT AT ADDR.
020076  000403                          BR      2$
020100  005237  020116          1$:     INC     $CNT            ;OVERFLOW?
020104  100402                          BMI     3$              ;YES-ERROR RETURN
020106  062705  000004          2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
020112  000205                  3$:     RTS     R5              ;RETURN.
```

```
2937  020114  000000                  $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
2938  020116  000000                  $CNT:   .WORD   0               ;# OF TIMES AT ADDR.
2939
2940                                          ;*
2941                                          ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
2942                                          ;*USE FOR A RESET.  FIRST,WE DO A RESET INSTRUCTION.
2943                                          ;*THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
2944                                          ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
2945                                          ;*
2946                                          ;*      CALL=JSR        PC,$RESET       ;REPLACES "RESET INSTRUCTION
2947                                          ;*              ;RETURNS HERE.
2948                                          ;*
2949  020120  000005                  $RESET: RESET                   ;RESET THE WORLD.
2950
2951                                  ;*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
2952  020132  005737  017146                  TST     $AERR           ;IF NO ERROR,LOOP
2953  020136  001004                          BNE     10$             ;THERE WAS AN ERROR.
2954  020140  062737  000002  020154          ADD     #2,2$           ;UPDATE DEVICE ADDR.
2955                                                                  ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
2956                                                                  ;IF 2$ CONTAINED A VALID ADDR,WE
2957                                                                  ;MUST KEEP TRYING UNTIL WE GENERATE
2958                                                                  ;AN INVALID ADDR.
2959  020146  000764                          BR      $RESET
2960  020150                          10$:
2961  020150  000207                          RTS     PC
2962  020152  000000                  1$:     .WORD   0               ;JUNK LOC.
2963  020154  160000                  2$:     .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
2964
2965
2966                                          ;
2967                                          ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
2968                                          ;        IS NOT TIME DEPENDENT CODE SENCE
2969                                          ;        NOT USED TO gET SPECIFIC TIME BUT
2970                                          ;        JUST A LITTLE DELAY.
2971                                          ;
2972                                          ;       THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
2973                                          ;       THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
2974                                          ;
2975                                          ;
2976                                          ;       CALL=   JSR PC,  SDELAY
2977                                          ;
2978  020156                          SDELAY:
2979  020156  005737  020240                  TST     RTCCSR          ;CLOCK PRESENT?
2980  020162  100016                          BPL     10$
2981  020164  012737  000002  020230          MOV     #2,TIME
2982  020172  052777  000115  000040          BIS     #115,@RTCCSR    ;START CLOCK
2983  020200  005037  177776                  CLR     PS
2984  020204  005737  020230          1$:     TST     TIME
2985  020210  001375                          BNE     1$
2986  020212  005077  000022                  CLR     @RTCCSR         ;STOP CLOCK
2987
2988  020216  000207                          RTS PC
2989  020220  105237  020230          10$:    INCB    TIME
2990  020224  001375                          BNE     10$
```

```
2991   020226  000207                          RTS     PC
2992
2993   020230  000000                  TIME:   .WORD   0
2994
2995   020232  005337  020230          CLKINT: DEC     TIME
2996   020236  000002                          RTI
2997   020240  000000                  RTCCSR: .WORD   0               ;CLOCK CSR IF USED.
2998                                           ;*
2999                                           ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
3000                                           ;*ANY DEVICE ON THE I/O BUS
3001                                           ;*USER MUST START AT THIS ADDR.
3002                                           ;*HE MUST SAY EITHER "E" FOR EXAMINE, OR "D" FOR DEPOSIT.
3003                                           ;*"E" IS DEFAULT.
3004                                           ;*NEXT, HE MUST SUPPLY AN ADDR.
3005                                           ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
3006                                           ;*WILL OCCUR.
3007
3008   020242                          $UTK:
3009   020242  005037  001464                  CLR     .DVLS
3010   020246                          21$:
3011   020246  104401  020254                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3012   020252  000405                          BR      64$             ;;GET OVER THE ASCIZ
3013                                  ;;65$:   .ASCIZ  <200>#E OR D?#
3014   020266                          64$:
3015   020266  105777  160652          1$:     TSTB    @$TKS
3016   020272  100375                          BPL     1$
3017   020274  117737  160646  020416          MOVB    @$TKB,20$       ;GET INPUT
3018   020302  104401  020416                  TYPE,   20$             ;ECHO, NEXT MESSAGE.
3019   020306  142737  000240  020416          BICB    #240,20$        ;STRIP PARITY, LC
3020   020314  104411                          RDOCT                   ;GET ADDR.
3021   020316  012637  020414                  MOV     (SP)+,14$
3022   020322  123727  020416  000104          CMPB    20$,#'D         ;DEPOSIT?
3023   020330  001411                          BEQ     10$
3024
3025   020332  004537  017764                  JSR     R5,$INLP        ;GET DATA
3026   020336  020414                  2$:     .WORD   14$
3027   020340  020352                          .WORD   5$
3028
3029   020342  013746  020352                  MOV     5$,-(SP)        ;;SAVE 5$ FOR TYPEOUT
3030   020346  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3031   020350  000736                          BR      21$             ;LOOP.
3032   020352  000000                  5$:     .WORD   0
3033
3034   020354                          10$:
3035   020354  104401  020362                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3036   020360  000404                          BR      66$             ;;GET OVER THE ASCIZ
3037                                  ;;67$:   .ASCIZ  <200>#DATA= #
3038   020372                          66$:
3039   020372  104411                          RDOCT
3040   020374  012637  020412                  MOV     (SP)+,13$
3041
3042   020400  004537  017666          11$:    JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
3043   020404  020414                  12$:    .WORD   14$             ;DEVICE ADDR.
3044
```

```
3045  020406  020412                          .WORD   13$             ;DATA
3046  020410  000716                          BR      21$
3047
3048  020412  000000                  13$:    .WORD   0
3049  020414  000000                  14$:    .WORD   0
3050  020416  100001  042504  044526  20$:    .ASCIZ  <1><200>#DEVICE ADDR= #
3051  020424  042503  040440  042104
3052  020432  036522  000040
3053                                          .EVEN
3054
3055
3056
3057                                          ;
3058                                          ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
3059                                          ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
3060                                          ;TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
3061                                          ;SAMPLE TAKEING PURPOSES.
3062                                          ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
3063                                          ;       A/D CSR IN BSEL 4,AND 5.
3064                                          ;       (2) HE MUST CALL THIS ROUTINE:
3065                                          ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
3066                                          ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
3067                                          ;               ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
3068                                          ;                               ;(UNTILL ONE DOES A RESET)
3069                                          ;
3070                                          ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
3071                                          ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR.!
3072                                          ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONe)
3073                                          ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
3074                                          ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
3075                                          ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
3076                                          ;
3077  020436  012537  020446          $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
3078  020442  004537  017764                  JSR     R5,$INLP
3079  020446  000000                  1$:     .WORD   0
3080  020450  020544                          .WORD   10$
3081  020452  113777  020026  160772          MOVB    $OFS,@KMAD6
3082  020460  113777  020026  160766          MOVB    $OFS,@KMAD7
3083  020466  013737  020446  020506          MOV     1$,2$
3084  020474  062737  000002  020506          ADD     #2,2$
3085  020502  004537  017764                  JSR     R5,$INLP
3086  020506  000000                  2$:     .WORD   0
3087  020510  020544                          .WORD   10$
3088  020512  113777  020026  160724          MOVB    $OFS,@KMAD3
3089  020520  152777  000340  160724          BISB    #340,@KMAD6
3090  020526  152777  000300  160720          BISB    #300,@KMAD7
3091  020534  152777  000300  160702          BISB    #300,@KMAD3
3092  020542  000205                          RTS     R5
3093  020544  000000                  10$:    .WORD   0
3094
3095          000001                          .END
```

```
ABASE = 170410            168#    280     321
ACDW1 = 000000            280     323
ACDW2 = 000000            280     324
ACPUOP= 000000            280     295
ADDW0 = 000000            280     325
ADDW1 = 000000            280     326
ADDW10= 000000            280     335
ADDW11= 000000            280     336
ADDW12= 000000            280     337
ADDW13= 000000            280     338
ADDW14= 000000            280     339
ADDW15= 000000            280     340
ADDW2 = 000000            280     327
ADDW3 = 000000            280     328
ADDW4 = 000000            280     329
ADDW5 = 000000            280     330
ADDW6 = 000000            280     331
ADDW7 = 000000            280     332
ADDW8 = 000000            280     333
ADDW9 = 000000            280     334
ADEVCT= 000000            280     286
ADEVM = 000000            280     322
AENV  = 000000            280     291
AENVM = 000000            280     292
AFATAL= 000000            280     283
AMADR1= 000000            280     308
AMADR2= 000000            280     312
AMADR3= 000000            280     315
AMADR4= 000000            280     318
AMAMS1= 000000            280     302
AMAMS2= 000000            280     310
AMAMS3= 000000            280     313
AMAMS4= 000000            280     316
AMSGAD= 000000            280     288
AMSGLG= 000000            280     289
AMSGTY= 000000            280     282
AMTYP1= 000000            280     303
AMTYP2= 000000            280     311
AMTYP3= 000000            280     314
AMTYP4= 000000            280     317
APASS = 000000            280     285
APRIOR= 000200            170#    280
APTCSU= 000040           2351    2457#
APTENV= 000001           1943    2344    2413    2455#
APTSIZ= 000200            516    2454#
APTSPO= 000100           2346    2415    2456#
ASWREG= 000000            280     293
ATESTN= 000000            280     284
AUNIT = 000000            280     287
AUSWR = 000000            280     294
AVECT1= 000350            169#    280     319     445     446
AVECT2= 000000            280     320
BEGIN   001564            193     472#
BEGIN1  001570            194     475#
```

```
BIT0   = 000001            153#    648     669     826     916    1015    1058    1091    1545    1604    1665
BIT00  = 000001            143#    153
BIT01  = 000002            142#    152
BIT02  = 000004            141#    151
BIT03  = 000010            140#    150
BIT04  = 000020            139#    149
BIT05  = 000040            138#    148
BIT06  = 000100            137#    147
BIT07  = 000200            136#    146
BIT08  = 000400            135#    145    1882
BIT09  = 001000            134#    144    1890    1954
BIT1   = 000002            152#   2615
BIT10  = 002000            133#    930    1541    1600
BIT11  = 004000            132#   1897    2555
BIT12  = 010000            131#
BIT13  = 020000            130#   1938
BIT14  = 040000            129#    869     916    1868    2547
BIT15  = 100000            128#    853     916     922    1518    2555
BIT2   = 000004            151#
BIT3   = 000010            150#
BIT4   = 000020            149#   2816
BIT5   = 000040            148#   2559    2772    2789
BIT6   = 000100            147#    899     916
BIT7   = 000200            146#    883     916    1499    1547
BIT8   = 000400            145#    840     916    1669
BIT9   = 001000            144#
BPTVEC= 000014             160#
BRLEV1  001554             464#
BRLEV2  001556             465#   1015*   1018    1044    1049*
BRLEV3  001560             466#   1016*   1017*   1026    1042    1091*   1093    1106    1119*   1545*   1549    1562    1576
                          1581    1594*   1604*   1608    1619    1635*   1826
CKSWR = 104406            1866    1867    1930    1953    2497#
CLKINT  020232            2995#
CR     = 000015             68#   2390    2400
CRLF   = 000200             69#   2361    2400
DDISP  = 177570             75#    258     504
DF1     013230             365     371     377     383     389     395     401    1828#
DF10    013234             407    1830#
DH1     013016             363    1799#
DH10    013137             405     411    1816#
DH2     013047             369    1804#
DH3     013100             375    1809#
DH4     013131             381     387     393     399    1814#
DIOBRL  001552             463#
DISPLA  001142             258#    504*    512*   1912*   1933*
DISPRE  000174             190#    512
DONOT   002046             520#
DRADD   001430             414#    520*    568     569     571     573
DRBRL   001434             416#
DRIV    001432             415#
DRLPX2= ******   G          14#   2545
DRT21   011176             932    1536#
DSWR   = 177570             74#    257     503
DT1     013202             364     370     376    1823#
```

```
DT10    013216              406     412     1826#
DT4     013212              382     388      394     400     1825#
EMTVEC= 000030              163#    487*     488*
EM1     012376              362    1747#
EM10    012656              404    1781#
EM11    012741              410    1790#
EM2     012427              368    1752#
EM3     012457              374    1757#
EM4     012510              380    1762#
EM5     012542              386    1767#
EM6     012575              392    1772#
EM7     012622              398    1776#
ERRVEC= 000004              156#    501     502*    513*    1873    1874*   1876*   1879*
EXTCNT  012246             1675*   1697*   1702#
EXTTMP  012244             1685    1687    1695    1697    1701#
EXTTST  011764              195    1656#   1673
FLIP    012076             1668    1672    1675#
FULNUL  012356             1737    1741#
GETSWR  013254              554     561     567     1837#
GNS   = ****** U            189     544     548     552     558     565     1841    2490    2491    2492    2493    2495    2497
                            2498    2499    2500    2538    2592    2596    2600    2828    3014    3038
GRBHIO  001534              456#    573*    574*
GRDAI   001530              454#    569*    570*    586     826     940     946     960     965     980     985     1000    1005
                            1026    1042    1069    1077    1102    1115    1137    1146    1149    1152    1161    1164    1167
                            1176    1179    1182    1191    1194    1197    1206    1209    1212    1221    1224    1227    1236
                            1239    1242    1251    1254    1257    1266    1269    1272    1281    1284    1287    1296    1299
                            1302    1323    1332    1335    1338    1347    1350    1353    1362    1365    1368    1377    1380
                            1383    1392    1395    1398    1407    1410    1413    1422    1425    1428    1437    1440    1443
                            1452    1455    1458    1467    1470    1473    1482    1485    1488    1527    1557    1581    1584
                            1616    1665    1685    1687    1695    1697
GRDIO   001532              455#    571*    572*    588     596     598     611     614     624     626     637     639     653
                             656     658     674     677     679     694     696     699     701     704     706     709     711
                             714     716     719     721     724     726     729     731     734     736     739     741     744
                             748     759     761     764     766     769     771     774     776     779     781     784     786
                             789     791     794     796     799     801     804     806     809     812     823     937     944
                             957     963     977     983     997    1003    1024    1037    1040    1066    1072    1075    1099
                            1109    1112    1134    1141    1144    1156    1159    1171    1174    1186    1189    1201    1204
                            1216    1219    1231    1234    1246    1249    1261    1264    1276    1279    1291    1294    1320
                            1327    1330    1342    1345    1357    1360    1372    1375    1387    1390    1402    1405    1417
                            1420    1432    1435    1447    1450    1462    1465    1477    1480    1505    1524    1554    1562
                            1576    1579    1613    1622    1663    1680    1683    1690    1693
GRIVA   001542              459#    1644*
GRIVB   001546              461#    1646*
GRIVSA  001544              460#    1644    1645*
GRIVSB  001550              462#    1646    1647*
GRSTAT  001526              453#     568*    584     829     831     843     845     856     858     869     872     874     886
                             888     899     902     904     916     919     922    1503    1508    1522    1530    1560    1564
                            1567    1588    1590    1619    1625    1628    1661    1668    1672
GTSWR = 104405              2495#
HT    = 000011               66#    2359    2400
IOTEST  002662              575#    1744    2307
IOTVEC= 000020              161#     485*    486*
KMADO   001436              428#     527    2547*   2555*   2559    2615    2675*   2677*   2679*   2680*   2687*   2692*   2772
                            2789    2816
```

```
KMAD1   001440            431#   528
KMAD2   001442            433#  2564*  2577   2617*  2631   2645*  2658   2726*  2731*  2735*  2757*  2774*  2791*
                         2810
KMAD3   001444            435#  3088*  3091*
KMAD4   001446            437#  2579   2584   2608*  2610*  2612*  2633   2676*  2681*  2684*  2693*  2725*  2729*
                         2734*  2756*  2776   2793
KMAD5   001450            439#
KMAD6   001452            441#  2678*  2688   3081*  3089*
KMAD7   001454            443#   532   3082*  3090*
LF    = 000012             67#  2394   2400
LPADH   001450            438#
LPADL   001446            436#
LPCI    001436            427#
LPCO    001442            432#
LPMR    001440            430#
LPMS1   001452            440#
LPMS2   001454            442#
LPSO    001444            434#
MSGSWR  013242           1832#  1846
NOTINT  001540            458#   562*  1549   1608
NOTLCH  001536            457#   555*  1016   1019   1060   1093   1103   1129   1315
PC    =%000007             87#   554*   561*   567*   611*   919*  1643*  1668*  1672*  1699*  1721*  1724*  1731*
                         1736   1844*  1849*  1940*  1946*  2058*  2181*  2306   2349*  2368*  2375*  2382*  2396*
                         2398*  2432*  2449*  2609*  2611*  2613*  2660*  2723*  2727*  2732*  2736*  2755*  2758*
                         2775*  2792*  2823*  2961*  2988*  2991*
PIRQ  = 177772             73#
PIRQVE= 000240            167#
PR0   = 000000             90#
PR1   = 000040             91#
PR2   = 000100             92#
PR3   = 000140             93#
PR4   = 000200             94#
PR5   = 000240             95#
PR6   = 000300             96#
PR7   = 000340             97#
PS    = 177776             70#    71   2983*
PSW   = 177776             71#  1648*
PWRMSG  015412            2305   2312#
PWRVEC= 000024             162#   491*   492*  2274*  2275*  2284*  2290*  2302*  2303*
RBEG    001574             474    476#
RDCHR = 104407            2117   2498#
RDLIN = 104410            1981   2499#
RDOCT = 104411            1847   2500#  3021   3040
RD1     017566            2759*  2796#
RESVEC= 000010             157#
RTCCSR  020240            2979   2982*  2986*  2997#
R0    =%000000             78#   473*   475*   525    527*   530*   531    536*   537   1130*  1138   1152*  1153
                         1167*  1168   1182*  1183   1197*  1198   1212*  1213   1227*  1228   1242*  1243   1257*
                         1258   1272*  1273   1287*  1288   1302*  1316*  1324   1338*  1339   1353*  1354   1368*
                         1369   1383*  1384   1398*  1399   1413*  1414   1428*  1429   1443*  1444   1458*  1459
                         1473*  1474   1488*  1527*  1728*  1731   1978   1982*  1985   2001*  2153   2154*  2155*
                         2162*  2163*  2164*  2165*  2166*  2167   2172   2177*  2179*  2183   2185   2276   2301*
                         2342   2343*  2348   2353   2356*  2409   2417*  2421   2422   2424*  2425*  2426   2448*
                         2467   2468*  2469   2470*  2471*  2472*  2473*  2673   2674*  2685*  2688   2696*  2720
```

```
                    2721*  2722*  2724   2725   2737*  2752   2753*  2754*  2756   2759   2794*  2847   2850*
                    2866*  2889   2892*  2910*
R1      =%000001      79#    526    528*   532    535*  1146*  1149   1161*  1164   1176*  1179   1191*  1194
                    1206*  1209   1221*  1224   1236*  1239   1251*  1254   1266*  1269   1281*  1284   1296*
                    1299   1304   1332*  1335   1347*  1350   1362*  1365   1377*  1380   1392*  1395   1407*
                    1410   1422*  1425   1437*  1440   1452*  1455   1467*  1470   1482*  1485   1490   1979
                    1983*  1987*  1989*  1991*  1994*  1997   2000*  2277   2300*  2410   2447*  2551   2552*
                    2553*  2556*  2604*  2605   2610   2612   2639*  2808   2809*  2812*  2818*  2822*  2848
                    2851*  2856*  2859   2865*  2890   2893*  2899*  2902   2909*
R2      =%000002      80#   1980   1984*  1988*  1990*  1992*  1998   1999*  2278   2299*
R3      =%000003      81#   2113   2114*  2115   2118*  2119   2123   2125   2127*  2129*  2225   2234*  2240*
                    2241*  2244*  2249*  2250*  2251   2260*  2279   2298*
R4      =%000004      82#   2226   2228*  2229*  2230*  2231   2232*  2246   2248*  2256*  2259*  2280   2297*
                    2672   2697*  2702*
R5      =%000005      83#    584*   586*   588*   596*   598*   611*   614*   624*   626*   637*   639*   653*
                     656*   658*   674*   677*   679*   694*   696*   699*   701*   704*   706*   709*   711*
                     714*   716*   719*   721*   724*   726*   729*   731*   734*   736*   739*   741*   744*
                     748*   759*   761*   764*   766*   769*   771*   774*   776*   779*   781*   784*   786*
                     789*   791*   794*   796*   799*   801*   804*   806*   809*   812*   823*   826*   829*
                     831*   843*   845*   856*   858*   869*   872*   874*   886*   888*   899*   902*   904*
                     916*   919*   922*   937*   940*   944*   946*   957*   960*   963*   965*   977*   980*
                     983*   985*   997*  1000*  1003*  1005*  1024*  1026*  1037*  1040*  1042*  1066*  1069*
                    1072*  1075*  1077*  1099*  1102*  1109*  1112*  1115*  1134*  1137*  1141*  1144*  1146*
                    1149*  1152*  1156*  1159*  1161*  1164*  1167*  1171*  1174*  1176*  1179*  1182*  1186*
                    1189*  1191*  1194*  1197*  1201*  1204*  1206*  1209*  1212*  1216*  1219*  1221*  1224*
                    1227*  1231*  1234*  1236*  1239*  1242*  1246*  1249*  1251*  1254*  1257*  1261*  1264*
                    1266*  1269*  1272*  1276*  1279*  1281*  1284*  1287*  1291*  1294*  1296*  1299*  1302*
                    1320*  1323*  1327*  1330*  1332*  1335*  1338*  1342*  1345*  1347*  1350*  1353*  1357*
                    1360*  1362*  1365*  1368*  1372*  1375*  1377*  1380*  1383*  1387*  1390*  1392*  1395*
                    1398*  1402*  1405*  1407*  1410*  1413*  1417*  1420*  1422*  1425*  1428*  1432*  1435*
                    1437*  1440*  1443*  1447*  1450*  1452*  1455*  1458*  1462*  1465*  1467*  1470*  1473*
                    1477*  1480*  1482*  1485*  1488*  1503*  1505*  1508*  1522*  1524*  1527*  1530*  1554*
                    1557*  1560*  1562*  1564*  1567*  1576*  1579*  1581*  1584*  1588*  1590*  1613*  1616*
                    1619*  1622*  1625*  1628*  1661*  1663*  1665*  1668*  1672*  1680*  1683*  1685*  1687*
                    1690*  1693*  1695*  1697*  2227   2233*  2235*  2237*  2238*  2239*  2240   2258*  2281
                    2296*  2544*  2566*  2620*  2640*  2647*  2698*  2738*  2761*  2778*  2795*  2862*  2867*
                    2870*  2904*  2911*  2915*  2927   2929   2934*  2935*  2952*  3026*  3043*  3078*  3085*
                    3092*
R6      =%000006      84#    479*   480*   481
R7      =%000007      85#
SDELAY   020156     2978#
SP      =%000006      86#    483*   501*   509*   513    525*   526*   535    536    576*  1656*  1848   1873*
                    1876   1878   1879   1908   1909   1913*  1935   1956*  1959*  1976*  1977*  1978*  1979*
                    1980*  1982   1985*  1993*  1994   1996   1997*  1999   2000   2001   2018*  2019*  2020
                    2027*  2030*  2031*  2035*  2036*  2040   2043*  2047   2049   2051   2052*  2059   2061
                    2063*  2064   2066*  2067*  2068*  2069*  2070*  2085*  2086*  2089*  2090*  2091   2095*
                    2096*  2097   2100   2102   2104*  2113*  2118   2129   2130*  2131*  2132*  2153*  2158*
                    2179   2183*  2217*  2218   2219   2220*  2225*  2226*  2227*  2233   2258   2259   2260
                    2261*  2262*  2276*  2277*  2278*  2279*  2280*  2281*  2282*  2283   2291*  2295   2296
                    2297   2298   2299   2300   2301   2306*  2342*  2343   2353*  2355   2356   2357*  2359
                    2361   2363   2369   2371*  2373*  2381*  2385   2389   2390   2394   2409*  2410*  2417
                    2418*  2429   2430*  2431*  2441   2442*  2447   2448   2467*  2468   2478*  2479*  2519*
                    2541   2542   2551*  2639   2672*  2673*  2696   2697   2720*  2737   2752*  2794   2808*
                    2822   2847*  2848*  2865   2866   2889*  2890*  2909   2910   3022   3030*  3041
```

```
STACK = 001100            61#    483     576     1656
STKLMT= 177774            72#
SWR     001140           257#    481     503#    505     511#    518#    555     562     930     1541    1600    1837    1848#
                        1868    1882    1884    1890    1897    1938    1950    1954    2014    2051#   2282    2295#
SWREG   000176           191#    511     1837    2014    2027
SW0   = 000001           125#
SW00  = 000001           115#    125
SW01  = 000002           114#    124
SW02  = 000004           113#    123
SW03  = 000010           112#    122
SW04  = 000020           111#    121
SW05  = 000040           110#    120
SW06  = 000100           109#    119
SW07  = 000200           108#    118
SW08  = 000400           107#    117
SW09  = 001000           106#    116
SW1   = 000002           124#
SW10  = 002000           105#
SW11  = 004000           104#
SW12  = 010000           103#
SW13  = 020000           102#
SW14  = 040000           101#
SW15  = 100000           100#
SW2   = 000004           123#
SW3   = 000010           122#
SW4   = 000020           121#
SW5   = 000040           120#
SW6   = 000100           119#
SW7   = 000200           118#
SW8   = 000400           117#
SW9   = 001000           116#
TBITVE= 000014           158#
TIME    020230          2981#   2984    2989#   2993#   2995#
TKVEC = 000060           165#
TPVEC = 000064           166#
TRAPVE= 000034           164#    489#    490#
TRTVEC= 000014           159#
TST1    002670           580#
TST10   003364           690#
TST11   003742           749     755#
TST12   004316           813     819#
TST13   004414           832     839#
TST14   004456           846     852#
TST15   004520           859     865#
TST16   004600           875     882#
TST17   004642           889     895#
TST2    002730           592#
TST20   004722           905     911#
TST21   005016           924     929#
TST22   005134           947     953#
TST23   005232           966     973#
TST24   005330           986     993#
TST25   005426          1006    1012#
TST26   005624          1056#
```

```
TST27   005764            1089#
TST3    002772             599     605#
TST30   006152            1126#
TST31   007472            1306    1312#
TST32   011012            1492    1498#
TST33   011100            1509    1517#
TST34   011176            1531    1540#
TST35   011506            1542    1599#
TST36   011712            1601    1641#
TST37   011762            1655#
TST4    003050             615     620#
TST5    003112             627     633#
TST6    003154             640     646#
TST7    003256             667#
TYPE  = 104401             542     546     550     556     563    1727    1839    1845    1941    2025    2026    2029    2042
                          2053    2072    2121    2124    2128    2152    2169    2171    2174    2176    2180    2187    2252
                          2304    2364    2490#   2536    2590    2594    2598    2826    3012    3019    3036
TYPOC = 104402            2028    2160    2184    2491#   3031
TYPON = 104404            2493#
TYPOS = 104403            2492#
VECTOR  001456             445#
VECTPS  001460             446#
VERSN   001462             448#
W1      017416            2724#   2739#
W2      017420            2728#   2740#
W3      017422            2733#   2741#
$AERR   017146            2543#   2636#   2662#   2952
$APTHD  001000             217     223#
$ASTAT= ****** U          2435    2450
$ATYC   015766            2406    2408#
$ATY1   015742            2404#
$ATY3   015750            2349    2405#
$ATY4   015760            1946    2407#
$AUTOB  001134             254#    2022    2143
$BASE   001252             321#     520
$BDADR  001122             249#
$BDDAT  001126             251#     598     614     626     639     658     679#    680     748     812     831     845     858
                           874     888     904     922#    923     946     965     985    1005    1026#   1027    1042#   1044
                          1077    1115    1304#   1305    1490#   1491    1508    1530    1546#   1590#   1605#   1823    1826
$CDW1   001256             323#
$CDW2   001260             324#
$CHARC  015736            2366#   2376#   2383    2392#   2397#
$CKSWR  014102            2014#   2497
$CMTAG  001100             237#    478     479     487     493     494     495
$CM1  = 000002             269#    270#    271#
$CM2  = 000004             269#    270#    271#
$CM3  = 000002             267#    269
$CNT    020116            2930#   2932#   2938#
$CNTLG  014627            2025    2138#
$CNTLU  014622            2042    2137#
$CPUOP  001224             295#
$CRLF   001173             274#    1941    1966    2053    2137    2152    2171    2176    2180    2365    2400
$DATR   017570            2776#   2793#   2797#   2908
$DDWD   001262             325#
```

```
$DDW1   001264            326#
$DDW10  001306            335#
$DDW11  001310            336#
$DDW12  001312            337#
$DDW13  001314            338#
$DDW14  001316            339#
$DDW15  001320            340#
$DDW2   001266            327#
$DDW3   001270            328#
$DDW4   001272            329#
$DDW5   001274            330#
$DDW6   001276            331#
$DDW7   001300            332#
$DDW8   001302            333#
$DDW9   001304            334#
$DEVCT  001206            286#
$DEVm   001254            322#
$DOAGN  012334           1723    1729    1735#
$ENDAD  012324            203    1731#   1961
$ENDCT  012304            493    1725#
$ENDMG  012343           1727    1739#
$ENuLL  012340           1738#
$ENV    001216            291#    1943    2344    2413    2437
$ENVM   001217            292#     516    2346    2351    2415
$EOP    012250           1649    1715#
$EOPCT  012276            493*   1722#   1726
$ERFLG  001103            240#   1856    1886    1888    1894*   1916    1931*   1966
$ERMAX  001115            246#    496*   1888    1911*   1916
$ERROR  013614            487    1929#
$ERRPC  001116            247#   1823    1825    1826    1935*   1936*   1937    1966    2158
$ERRTB  001322            359#   2166
$ERRTY  014656           1940    2151#
$ERTTL  001112            244#   1934*   1966
$ESCAP  001170            272#    495*   1910*   1957    1959    1966
$ETABL  001216            290#
$ETEND  001322            229     343#
$FATAL  001200            283#   2441*
$FFLG   016206           2404*   2407*   2435    2444*   2452#
$FILLC  001156            265#   2369    2400
$FILLS  001155            264#   2400
$GDADR  001120            248#
$GDDAT  001124            250#    593*    596     598     607*    621*    624     626     634*    637     639     648*    653
                          658     662*    669*    674     680     684*    691*    694     748     756*    759     812     826*
                          829     831     840*    843     845     853*    856     858     869*    872     874     883*    886
                          888     899*    902     904     913*    916     940*    946     960*    963     965     980*    983
                          985    1000*   1003    1005    1018*   1019    1024    1027    1037    1043*   1058*   1060    1072
                         1077    1083*   1102*   1103*   1106*   1109    1115    1128*   1129*   1130    1305    1314*   1315*
                         1316    1491    1499*   1508    1518*   1547*   1584*   1606*   1658*   1661    1663    1665    1823
                         1826
$GET42  012314           1728#
$GTSWR  014152           2026#   2495
$HD   = 000000             40
$HIBTS  001000            224#
$HIOCT  014100           1998*   2003#
```

```
$ICNT   001104          241#  1901*  1902   1904*  1915
$ILLUP  015404         2274   2290   2309#
$INLP   017764          598    614    626    639    658    679    696    701    706    711    716    721    726
                        731    736    741    748    761    766    771    776    781    786    791    796    801
                        806    812    831    845    858    874    888    904    922    946    965    985   1005
                       1026   1037   1042   1077   1115   1146   1149   1161   1164   1176   1179   1191   1194
                       1206   1209   1221   1224   1236   1239   1251   1254   1266   1269   1281   1284   1296
                       1299   1332   1335   1347   1350   1362   1365   1377   1380   1392   1395   1407   1410
                       1422   1425   1437   1440   1452   1455   1467   1470   1482   1485   1508   1527   1530
                       1567   1576   1581   1590   1628   1685   1695  2889#  2952   3026   3078   3085
$INTAG  001135          255#  2054   2143
$ITEMB  001114          245#  1937*  1945   1966   2155
$LF     001174          275#  1966   2128   2137   2400
$LFLG   016205         2445*  2451#
$LOAD   017150         2544   2672#
$LPADR  001106          242#   497*  1892*  1908*  1913   1915
$LPAI   016270         2518#  2870   2915
$LPERR  001110          243#   498*   647*   668*  1013*  1057*  1090*  1544*  1603*  1892   1909*  1915   1956
$LPW    017572         2723   2727   2732   2736   2755   2758   2775   2792  2808#
$MADR1  001230          308#
$MADR2  001234          312#
$MADR3  001240          315#
$MADR4  001244          318#
$MAIL   001176          225    229    281#   515   1907   1943   2344
$MAMS1  001226          302#
$MAMS2  001232          310#
$MAMS3  001236          313#
$MAMS4  001242          316#
$MBADR  001002          225#
$MFLG   016204         2405*  2411   2446*  2450#
$MNEW   014645         2029   2141#
$MSGAD  001212          288#  2421*  2424
$MSGLG  001214          289#  2426*
$MSGTY  001176          282#  2419   2427*  2439   2443*
$MSWR   014634         2026   2139#
$MTYP1  001227          303#
$MTYP2  001233          311#
$MTYP3  001237          314#
$MTYP4  001243          317#
$MXCNT  013612         1905   1915#
$NULL   001154          263#  2371   2400
$NWTST= 000001          577#   589#   602#   617#   630#   643#   664#   687#   752#   816#   836#   849#   862#
                        879#   892#   908#   926#   950#   970#   990#  1009#  1053#  1086#  1123#  1309#  1495#
                       1514#  1537#  1596#  1638#  1652#
$OCNT   015234         2224*  2253*  2266#
$OFS  = 020026         2905#  3081   3082   3088
$OMODE  015236         2219*  2223*  2228   2231*  2242*  2268#
$OUTLP  017666          584    586    588    596    611    624    637    653    656    674    677    694    699
                        704    709    714    719    724    729    734    739    744    759    764    769    774
                        779    784    789    794    799    804    809    823    826    829    843    856    869
                        872    886    899    902    916    919    937    940    944    957    960    963    977
                        980    983    997   1000   1003   1024   1040   1066   1069   1072   1075   1099   1102
                       1109   1112   1134   1137   1141   1144   1152   1156   1159   1167   1171   1174   1182
                       1186   1189   1197   1201   1204   1212   1216   1219   1227   1231   1234   1242   1246
```

```
                    1249    1257    1261    1264    1272    1276    1279    1287    1291    1294    1302    1320    1323
                    1327    1330    1338    1342    1345    1353    1357    1360    1368    1372    1375    1383    1387
                    1390    1398    1402    1405    1413    1417    1420    1428    1432    1435    1443    1447    1450
                    1458    1462    1465    1473    1477    1480    1488    1503    1505    1522    1524    1554    1557
                    1560    1562    1564    1579    1584    1588    1613    1616    1619    1622    1625    1661    1663
                    1665    1668    1672    1680    1683    1687    1690    1693    1697    2847#   3043
                    1869    1885    1893    1903    1912#
$OVER   013576       285#    515*   1719*   1720*   1738    1899    1916
$PASS   001204       227#
$PASTM  001006      3077#
$PUTS   020436      2307#
$PWRAD  015400       491    2274#   2302
$PWRDN  015240      2305#
$PWRMG  015374      2284    2290#
$PWRUP  015312       273#   1966    2072    2121    2137    2400
$QUES   001172      2085#   2498
$RDCHR  014364      2501
$RDDEC= ****** U    2113#   2499
$RDLIN  014504      1976#   2500
$RDOCT  014000      2106#
$RDSZ = 000010       267#
$REGAD  001160       269#
$REGO   001162       270#
$REGI   001164       611     919    1643    2949#   2959
$RESET  020120      1737#
$RTNAD  012336      2501
$R2A  = ****** U    2927    2929*   2937#
$SAD    020114      2501
$SAVRE= ****** U    2283*   2291    2292*   2293*   2311#
$SAVR6  015410       485    1865#
$SCOPE  013332       397#    484     485     487     489     491     493     494     495     497    1717    1866    1930
$SETUP= 000137      1953    1961    2009    2143
                     397#
$STUP = 177777      1877    1906#
$SVLAD  013542       201#    206
$SVPC.= 000214        30#     40     175     176     177     178     179     180     181     271     272     273     494
$SWR  = 165400       495     497     498     581     593     606     621     634     647     668     691     756     820
                     840     853     866     883     896     912     930     954     974     994    1013    1057    1090
                    1127    1313    1499    1518    1541    1600    1642    1656    1710    1718    1730    1736    1738
                    1857    1858    1859    1860    1861    1868    1880    1882    1883    1886    1887    1888    1895
                    1896    1897    1909    1912    1915    1922    1923    1924    1925    1934    1938    1950    1954
                    1966    2308
$SWREG  001220       293#    518
$SWRMK= 000000       181     182    1861    1862    1884
$TESTN  001202       284#   1907*
$TIMES  001166       271#    494*    606*    912*   1642*   1718*   1895*   1902    1905*   1915
$TKB    001146       260#   2007    2018    2035    2089    2095    3018
$TKS    001144       259#   2007    2016    2032    2056*   2087    2093    3016
$TLKR   017424      2752#   2904
$TLKW   017310      2720#   2862
$TMDAT  001562       467#    581*    584     586     588     608*    611     650*    653*    656     671*    674*    677
                     696*    699     701*    704     706*    709     711*    714     716*    719     721*    724     726*
                     729     731*    734     736*    739     741*    744     761*    764     766*    769     771*    774
                     776*    779     781*    784     786*    789     791*    794     796*    799     801*    804     806*
```

```
                    809     820*    823*    826     866*    869     896*    899     916*    919     934*    937*    940
                    941*    944     954*    957*    960     974*    977*    980     994*    997*    1000    1037*   1040
                   1063*   1066*   1069    1072*   1075    1096*   1099*   1102    1109*   1112    1131*   1134*   1137
                   1138*   1141*   1144    1146    1149*   1152    1153*   1156*   1159    1161    1164*   1167    1168*
                   1171*   1174    1176    1179*   1182    1183*   1186*   1189    1191    1194*   1197    1198*   1201*
                   1204    1206    1209*   1212    1213*   1216*   1219    1221    1224*   1227    1228*   1231*   1234
                   1236    1239*   1242    1243*   1246*   1249    1251    1254*   1257    1258*   1261*   1264    1266
                   1269*   1272    1273*   1276*   1279    1281    1284*   1287    1288*   1291*   1294    1296    1299*
                   1302    1317*   1320*   1323    1324*   1327*   1330    1332    1335*   1338    1339*   1342*   1345
                   1347    1350*   1353    1354*   1357*   1360    1362    1365*   1368    1369*   1372*   1375    1377
                   1380*   1383    1384*   1387*   1390    1392    1395*   1398    1399*   1402*   1405    1407    1410*
                   1413    1414*   1417*   1420    1422    1425*   1428    1429*   1432*   1435    1437    1440*   1443
                   1444*   1447*   1450    1452    1455*   1458    1459*   1462*   1465    1467    1470*   1473    1474*
                   1477*   1480    1482    1485*   1488    1500*   1503    1505    1519*   1522    1524    1527    1551*
                   1554*   1557*   1560    1564    1567    1576*   1579    1581*   1584    1585*   1588    1590    1610*
                   1613*   1616*   1619*   1622*   1625    1628    1665*   1668    1669*   1672    1677*   1680*   1683
                   1687*   1690*   1693
$TN     = 000040     30#      40     577     581#    589     593#    599     602     606#    615     617     621#    627
                    630     634#    640     643     647#    664     668#    687     691#    749     752     756#    813
                    816     820#    832     836     840#    846     849     853#    859     862     866#    875     879
                    883#    889     892     896#    905     908     912#    924     926     930#    947     950     954#
                    966     970     974#    986     990     994#   1006    1009    1013#   1053    1057#   1086    1090#
                   1123    1127#   1306    1309    1313#   1492    1495    1499#   1509    1514    1518#   1531    1537
                   1541#   1542    1596    1600#   1601    1638    1642#   1652    1656#
$TOUT   020060     2566    2620    2647    2761    2778    2927#
$TP8    001152      262#   2389*   2400
$TPFLG  001157      266#   2338    2400
$TPS    001150      261#   2387    2400
$TRAP   016210      489    2467#
$TRAP2  016232     2478#   2489
$TRP    = 000012   2482#   2491#   2492#   2493#   2494#   2495    2496#   2497    2498#   2499#   2500#   2501#
$TRPAD  016244     2472    2489#
$TSTM   001004      226#
$TSTNM  001102      239#   1717*   1856    1884    1906*   1907    1912    1916    1933    1966
$TTYIN  014612     2114    2115    2132    2136#
$TYPBN= ****** U   2494
$TYPOS= ****** U   2494
$TYPE   015460     2338#   2432    2482    2490
$TYPEC  015672     2058    2368    2375    2382    2387#   2388
$TYPEX  015740     2393    2395    2398#
$TYPOC  015036     2222#   2491
$TYPON  015052     2221    2224#   2493
$TYPOS  015012     2217#   2492
$UNIT   001210      287#
$UNITM  001010      228#
$USWR   001222      294#
$UTK    020242     3009#
$VECT1  001246      319#
$VECT2  001250      320#
$XTSTR  013346     1871#
$$GET4= 000000     1730#
$OFILL  015235     2218*   2222*   2232    2267#
$40CAT= ****** U   1868    1940
.       = 020546    185#    189#    201     202#    204#    206#    207#    213     214#    216#    218#    236#    276
```

```
                    451#    482     497     498     545#    549#    566#    1738    1741    1822#   1834#   1915    1916
                   1966     2007    2136#   2137    2143    2190#   2286    2310    2319#   2400    2453#   2905    3015#
.DVLS   001464      450#     534*    1657*   2850    2871    2892    2916    3010*
.$ASTA= ****** U   2405     2408
.$X    = 001000     213#     218
```

```
CLERIN      926#    937     957     977     997    1066    1099    1134    1320    1554    1613
CLEROT      926#    933     954     974     994    1063    1072    1096    1109    1131    1141    1156    1171    1186    1201
           1216    1231    1246    1261    1276    1291    1317    1327    1342    1357    1372    1387    1402    1417    1432
           1447    1462    1477    1551    1610    1676    1687
CLRVCT      469#   1644
COMMEN      168#
ENDCOM      168#
ERROR        62#     600     616     628     641     660     682     750     814     833     847     860     876     890     906
            925     948     967     987    1007    1029    1046    1079    1117    1307    1493    1510    1532    1569    1592
           1630    2562    2568    2581    2622    2649    2763    2780
ESCAPE      168#
GETPRI      168#
GETSWR      168#
MOVEI        20#    596     612     624     637     656     677     694     699     704     709     714     719     724     729
            734     739     746     759     764     769     774     779     784     789     794     799     804     810     829
            843     856     872     886     902     920     944     963     983    1003    1024    1034    1040    1075    1113
           1144    1147    1159    1162    1174    1177    1189    1192    1204    1207    1219    1222    1234    1237    1249
           1252    1264    1267    1279    1282    1294    1297    1330    1333    1345    1348    1360    1363    1375    1378
           1390    1393    1405    1408    1420    1423    1435    1438    1450    1453    1465    1468    1480    1483    1506
           1525    1528    1565    1573    1579    1588    1626    1683    1693    2950
MOVEM        19#    582     584     586     594     609     622     635     651     654     672     675     692     697     702
            707     712     717     722     727     732     737     742     757     762     767     772     777     782     787
            792     797     802     807     821     824     827     841     854     867     870     884     897     900     914
            917     935     938     942     955     958     961     975     978     981     995     998    1001    1022    1038
           1064    1067    1070    1073    1097    1100    1107    1110    1132    1135    1139    1142    1150    1154    1157
           1165    1169    1172    1180    1184    1187    1195    1199    1202    1210    1214    1217    1225    1229    1232
           1240    1244    1247    1255    1259    1262    1270    1274    1277    1285    1289    1292    1300    1318    1321
           1325    1328    1336    1340    1343    1351    1355    1358    1366    1370    1373    1381    1385    1388    1396
           1400    1403    1411    1415    1418    1426    1430    1433    1441    1445    1448    1456    1460    1463    1471
           1475    1478    1486    1501    1503    1520    1522    1552    1555    1558    1560    1562    1577    1582    1586
           1611    1614    1617    1620    1623    1659    1661    1663    1666    1670    1678    1681    1685    1688    1691
           1695
MULT        168#
NEWTST      168#    577     589     602     617     630     643     664     687     752     816     836     849     862     879
            892     908     926     950     970     990    1009    1053    1086    1123    1309    1495    1514    1536    1596
           1638    1652
POP         168#   1999    2295    2296    2447    2448
PUSH        168#   1978    2276    2282    2408    2410    2431
REPORT      168#
SCOPE        63#    580     592     605     620     633     646     667     690     755     819     839     852     865     882
            895     911     929     953     973     993    1012    1056    1089    1126    1312    1498    1517    1540    1599
           1641    1655    1716
SETPRI      168#
SETTRA     2482#   2491    2492    2493    2495    2497    2498    2499    2500
SETUP       168#    477
SKIP        168#    599     615     627     640     749     813     832     846     859     875     889     905     924     947
            966     986    1006    1028    1045    1078    1116    1306    1492    1509    1531    1542    1550    1568    1591
           1601    1609    1629
SLASH       168#
SPACE       168#
STARS       168#    199     210     212     219     232     276     279     469     471     577     579     589     591     602
            604     617     619     630     632     643     645     664     666     687     689     752     754     816     818
            836     838     849     851     862     864     879     881     892     894     908     910     926     928     950
            952     970     972     990     992    1009    1011    1053    1055    1086    1088    1123    1125    1309    1311
```

```
                1495    1497    1514    1516    1537    1539    1596    1598    1638    1640    1652    1654    1706    1853    1918
                1968    2006    2009    2077    2106    2146    2194    2272    2288    2323    2403    2461
SWRSU            168#    499#
TOUT             453#    2565    2619    2646    2760    2777
TRMTRP          2482#
TYPBIN           168#
TYPDEC           168#
TYPNAM           168#
TYPNUM           168#
TYPOCS           168#
TYPOCT           168#    2027    2158    2182    3030
TYPTXT           168#     542     546     550     556     563    1839    2536    2590    2594    2598    2825    3011    3036
$CAL.             30#    2501
$DMAST            30#
$DMDT             30#
$MMAST            30#
$SCMRE           230#     269     270
$SCmTM           230#
$SESCA           168#
$SNEWT           168#     577     589     602     617     630     643     664     687     752     816     836     843     862     879
                 892      908     926     950     970     990    1009    1053    1086    1123    1309    1495    1514    1537    1596
                1638    1652
$SSET           2482#    2491    2492    2493    2495    2497    2498    2499    2500
$SSETM           515#
$SSKIP           168#     599     615     627     640     749     813     832     846     859     875     889     905     924     947
                 966      986    1006    1306    1492    1509    1531    1542    1601
.EQUAT            30#      58
.HEADE            30#
.KMADR            17#     419
.KSIS             22#     521
.LOADL            25#    2664
.LPAIN            22#    2502
.PUTCS            24#    3057
.RESET            22#    2939
.SETUP            30#     397
.SWRHI            30#     171
.SWRLO           182#
.UTK              29#    2999
.$ACT1            30#     197
.$APTB           277#
.$APTH            30#     208
.$APTY            30#    2401
.$CATC            30#     183
.$CMTA            30#     230
.$EOp             30#    1704
.$ERRO            30#    1916
.$ERRT            30#    2144
.$INLP            28#    2874
.$MMAC            18#
.$OUTL            27#    2835
.$PARM            30#
.$POWE            30#    2270
.$RDOC            30#    1966
.$READ            30#    2004
```

```
.$SAVE      30#
.$SCOP      30#   1851
.$SPAC      30#
.$SWDO      30#
.$TLKW      26#   2713
.$TOUT     453#   2918
.$TRAP      30#   2459
.$TYPE      30#   2321
.$TYPO      30#   2192
```

```
ADD      570   572   574  1994  2043  2052  2166  2220  2230  2357  2418  2430  2442  2934  2954
        3084
ASL      662   684  1049  1083  1119  1594  1635  1987  1989  1991  2066  2067  2068  2163  2164
        2165  2471
ASR     2425
BEQ      517   538   599   615   627   640   659   681   749   813   889   905   924   931   947
         966   986  1006  1020  1028  1045  1078  1116  1306  1492  1509  1609  1729  1838  1883
        1885  1887  1891  1900  1932  1955  1958  1986  2023  2050  2065  2168  2173  2186  2247
        2347  2360  2395  2412  2416  2436  2438  2560  2585  2606  2634  2691  2811  2817  2853
        2855  2895  2898  2928  3024
BGE     1903
BGT     1723  2062  2103  2254
BHI     1889
BIC      674   922  1026  1037  1042  1103  1106  1129  1315  1576  1720  1993  2019  2036  2063
        2090  2096  2104  2244
bICB    3020
BIS      653  1149  1164  1179  1194  1209  1224  1239  1254  1269  1284  1299  1335  1350  1365
        1380  1395  1410  1425  1440  1455  1470  1485  1581  2070  2249  2250  2547  2677  2679
        2687  2722  2754  2982
BISB    2155  3089  3090  3091
BIT      831   845   858   874   930  1019  1044  1060  1093  1541  1549  1600  1608  1868  1882
        1890  1897  1938  1954  2559  2615  2772  2789  2816
BITB     516  2346  2351  2383  2415
BLOS    2116
BLT     2060  2101  2255  2374
BMI     1531  1568  2933
BNE      482   506   533   541   663   685   832   846   859   875  1050  1061  1084  1094  1120
        1542  1550  1595  1601  1636  1698  1743  1869  1898  1939  1944  1962  2015  2021  2041
        2048  2055  2092  2098  2120  2126  2156  2178  2245  2294  2345  2352  2354  2362  2370
        2384  2391  2414  2420  2423  2440  2554  2557  2578  2580  2587  2589  2616  2632  2659
        2683  2689  2773  2790  2813  2819  2953  2985  2990
BPL     1591  1629  1951  2017  2033  2088  2094  2243  2339  2388  2703  2980  3017
BR       474   508   543   547   551   557   564  1673  1840  1871  1877  1880  1893  1896  1949
        1995  2044  2071  2073  2099  2122  2161  2188  2221  2236  2257  2286  2310  2341  2367
        2377  2386  2393  2406  2428  2521  2537  2540  2574  2591  2595  2599  2628  2655  2694
        2706  2769  2786  2814  2820  2827  2832  2857  2872  2900  2917  2931  2959  3013  3032
        3037  3046
CLR      473   480   494   495   515   534   607   913   940  1043  1519  1546  1584  1606  1645
        1647  1648  1657  1658  1717  1718  1741  1895  1910  1983  1984  2030  2031  2154  2234
        2292  2543  2552  2638  2675  2676  2680  2684  2692  2809  2851  2869  2893  2914  2930
        2983  2986  3010
CLRB    1894  2127  2366  2392  2444  2445  2446
CMP      481   505   532   598   626   639   658   680   748   812   888   904   946   965   985
        1005  1027  1077  1115  1305  1491  1505  1508  1524  1837  1878  1902  1961  2014  2020
        2040  2047  2059  2061  2091  2097  2100  2102  2115  2541  2605  2682  2688  2690  2854
        2897  2927
CMPB    1884  1888  1943  2022  2054  2119  2125  2344  2359  2361  2369  2390  2394  2413  2577
        2579  2584  2631  2658  2810  3023
COM      679   696   701   706   711   716   721   726   731   736   741   761   766   771   776
         781   786   791   796   801   806  1017  1152  1167  1182  1197  1212  1227  1242  1257
        1272  1287  1302  1338  1353  1368  1383  1398  1413  1428  1443  1458  1473  1488
DEC     1697  1721  1742  2162  2995
DECb    2242  2253  2373  2376
EMT       62
```

```
        2791    2793    3018    3081    3082    3088
NEG     2229
NOP      476     575     744     745     809    1104    1105    1732    1733    1734
RESET   1730    2949
ROL     1988    1990    1992    2235    2237    2238    2239    2241
RTI      510    1914    1965    2002    2057    2105    2133    2263    2308    2358    2480    2996
RTS     1699    1844    1849    2181    2398    2449    2473    2640    2660    2698    2738    2795    2823    2867    2911
        2935    2961    2988    2991    3092
SUB     1936    2424
tRAP    2482    2491    2492    2493    2495    2497    2498    2499    2500
TST      537     540     614     923    1530    1567    1875    1899    1950    1957    1996    2049    2064    2185    2246
        2355    2363    2385    2419    2437    2439    2469    2701    2852    2860    2894    2903    2952    2979    2984
TSTB    1628    1886    2016    2032    2087    2093    2338    2387    2411    2422    2435    2633    3016
.ASCII   273     274
.ASCIZ   275     545     549     553     559     566    1739    1747    1752    1757    1762    1767    1772    1776    1781
        1790    1799    1804    1809    1814    1816    1832    1842    2137    2138    2139    2141    2189    2312    2539
        2593    2597    2601    2829    3015    3039    3050
.ASECT    14
.BLKB   2136
.BLKW    451
.BYTE    239     240     245     246     254     255     263     264     265     266     291     292     302     303     310
         311     313     314     316     317    1738    1828    1830    1947    1948    2134    2135    2264    2265    2266
        2267    2450    2451    2452
.DSABL  2074
.ENABL    30    2007
.END    3095
.ENDC     35      62     154     168     178     180     181     182     194     200     204     206     211     213     220
         233     237     239     267     271     272     273     277     280     302     310     313     316     319     320
         321     322     323     324     325     326     327     328     329     330     331     332     333     334     335
         336     337     338     339     340     341     345     397     448     470     472     483     484     487     489
         491     493     494     495     497     499     520     537     545     549     553     559     566     578     579
         580     581     590     591     592     593     600     603     604     605     606     607     616     618     619
         620     621     628     631     632     633     634     641     644     645     646     647     665     666     667
         668     688     689     690     691     750     753     754     755     756     814     817     818     819     820
         833     837     838     839     840     847     850     851     852     853     860     863     864     865     866
         876     880     881     882     883     890     893     894     895     896     906     909     910     911     912
         913     925     927     928     929     930     948     951     952     953     954     967     971     972     973
         974     987     991     992     993     994    1007    1010    1011    1012    1013    1029    1046    1054    1055
        1056    1057    1079    1087    1088    1089    1090    1117    1124    1125    1126    1127    1307    1310    1311
        1312    1313    1493    1496    1497    1498    1499    1510    1515    1516    1517    1518    1532    1538    1539
        1540    1541    1543    1551    1569    1592    1597    1598    1599    1600    1602    1610    1630    1639    1640
        1641    1642    1643    1653    1654    1655    1656    1707    1709    1710    1712    1714    1717    1723    1726
        1727    1728    1730    1736    1738    1741    1842    1854    1857    1862    1868    1870    1881    1884    1885
        1886    1888    1890    1897    1901    1906    1908    1912    1915    1916    1919    1922    1931    1935    1940
        1941    1942    1950    1961    1965    1966    1969    1971    2004    2007    2008    2010    2038    2074    2078
        2106    2107    2114    2116    2119    2121    2137    2143    2147    2162    2191    2195    2273    2282    2283
        2289    2295    2296    2306    2308    2312    2324    2353    2404    2405    2408    2435    2450    2462    2468
        2471    2490    2491    2492    2493    2494    2495    2496    2497    2498    2499    2500    2501    2539    2569
        2576    2593    2597    2601    2623    2630    2650    2657    2764    2771    2781    2788    2829    2999    3015
        3039
.EQUIV    62      63      71     116     117     118     119     120     121     122     123     124     125     144     145
         146     147     148     149     150     151     152     153
.EVEN     280     545     549     553     559     566    1822    1834    1842    2190    2319    2453    2539    2593    2597
        2601    2829    3015    3039    3053
```

```
.GLOBL  14
.IF     31    60   126   154   178   179   180   181   182   192   199   202   204   210   212
       219   232   236   238   267   271   272   273   276   277   279   302   310   313   316
       319   320   321   322   323   324   325   326   327   328   329   330   331   332   333
       334   335   336   337   338   339   340   341   345   397   445   469   471   478   483
       485   487   489   491   493   494   495   497   515   537   544   548   552   558   565
       577   579   581   589   591   593   599   602   604   606   607   615   617   619   621
       627   630   632   634   640   643   645   647   664   666   668   687   689   691   749
       752   754   756   813   816   818   820   832   836   838   840   846   849   851   853
       859   862   864   866   875   879   881   883   889   892   894   896   905   908   910
       912   913   924   926   928   930   947   950   952   954   966   970   972   974   986
       990   992   994  1006  1009  1011  1013  1028  1045  1053  1055  1057  1078  1086  1088
      1090  1116  1123  1125  1127  1306  1309  1311  1313  1492  1495  1497  1499  1509  1514
      1516  1518  1531  1537  1539  1541  1542  1550  1568  1591  1596  1598  1600  1601  1609
      1629  1638  1640  1642  1643  1652  1654  1656  1706  1707  1708  1709  1710  1711  1712
      1716  1722  1725  1727  1728  1730  1736  1738  1739  1741  1841  1853  1856  1861  1867
      1868  1880  1882  1883  1884  1886  1887  1888  1897  1899  1907  1909  1914  1915  1916
      1918  1921  1931  1934  1938  1940  1941  1943  1950  1954  1961  1965  1966  1968  1971
      1983  2006  2008  2009  2010  2038  2077  2078  2106  2114  2115  2119  2120  2136  2137
      2143  2146  2161  2177  2194  2272  2282  2283  2288  2295  2296  2304  2306  2308  2312
      2323  2344  2403  2405  2408  2435  2450  2461  2467  2471  2482  2491  2492  2493  2494
      2495  2497  2498  2499  2500  2501  2538  2568  2574  2592  2596  2600  2622  2628  2649
      2655  2763  2769  2780  2786  2828  2999  3014  3038
.IFF      60   178   179   181   182   200   204   206   211   213   220   233   236   239   267
       277   280   470   472   483   578   579   580   581   590   591   592   593   600   603
       604   605   606   616   618   619   620   621   628   631   632   633   634   641   644
       645   646   647   665   666   667   668   688   689   690   691   750   753   754   755
       756   814   817   818   819   820   833   837   838   839   840   847   850   851   852
       853   860   863   864   865   866   876   880   881   882   883   890   893   894   895
       896   906   909   910   911   912   925   927   928   929   930   948   951   952   953
       954   967   971   972   973   974   987   991   992   993   994  1007  1010  1011  1012
      1013  1028  1045  1054  1055  1056  1057  1078  1087  1088  1089  1090  1116  1124  1125
      1126  1127  1307  1310  1311  1312  1313  1493  1496  1497  1498  1499  1510  1515  1516
      1517  1518  1532  1538  1539  1540  1541  1543  1550  1568  1591  1597  1598  1599  1600
      1602  1609  1629  1639  1640  1641  1642  1643  1653  1654  1655  1656  1707  1711  1717
      1722  1725  1738  1854  1881  1884  1885  1888  1915  1919  1921  1935  1961  1966  1969
      2007  2010  2078  2080  2085  2106  2107  2116  2120  2137  2147  2162  2191  2195  2273
      2289  2304  2324  2404  2462  2468  2568  2574  2622  2628  2649  2655  2763  2769  2780
      2786
.IFT    545   549   553   559   566  1842  1896  1941  1987  2003  2004  2080  2085  2539  2593
      2597  2601  2829  3015  3039
.IFTF   545   549   553   559   566  1842  1894  1940  1983  1987  2003  2025  2078  2081  2539
      2593  2597  2601  2829  3015  3039
.IIF     30    35    40   175   176   177   178   181   182   189   276   280   484   487   493
       494   495   497   498  1709  1710  1717  1718  1738  1741  1857  1858  1859  1860  1861
      1862  1866  1895  1896  1912  1915  1916  1922  1923  1924  1925  1930  1953  1961  1966
      2007  2028  2129  2137  2143  2159  2184  2400  2490  2491  2492  2493  2495  2497  2498
      2499  2500  3031
.IRP     397   577   589   602   617   630   643   664   687   752   816   836   849   862   879
       892   908   926   950   970   990  1009  1053  1086  1123  1309  1495  1514  1537  1596
      1638  1652  1867  1978  1999  2276  2282  2295  2296  2409  2410  2431  2447  2448
.LIST    14    30   168   181   189   267   269   270   271   277   280   397   499   545   549
       553   559   566   577   581   584   586   588   589   593   596   598   602   606   611
       614   617   621   624   626   630   634   637   639   643   647   653   656   658   664
```

```
                668   674   677   679   687   691   694   696   699   701   704   706   709   711   714
                716   719   721   724   726   729   731   734   736   739   741   744   748   752   756
                759   761   764   766   769   771   774   776   779   781   784   786   789   791   794
                796   799   801   804   806   809   812   816   820   823   826   829   831   836   840
                843   845   849   853   856   858   862   866   869   872   874   879   883   886   888
                892   896   899   902   904   908   912   916   919   922   926   930   937   940   944
                946   950   954   957   960   963   965   970   974   977   980   983   985   990   994
                997  1000  1003  1005  1009  1013  1024  1026  1037  1040  1042  1053  1057  1066  1069
               1072  1075  1077  1086  1090  1099  1102  1109  1112  1115  1123  1127  1134  1137  1141
               1144  1146  1149  1152  1156  1159  1161  1164  1167  1171  1174  1176  1179  1182  1186
               1189  1191  1194  1197  1201  1204  1206  1209  1212  1216  1219  1221  1224  1227  1231
               1234  1236  1239  1242  1246  1249  1251  1254  1257  1261  1264  1266  1269  1272  1276
               1279  1281  1284  1287  1291  1294  1296  1299  1302  1309  1313  1320  1323  1327  1330
               1332  1335  1338  1342  1345  1347  1350  1353  1357  1360  1362  1365  1368  1372  1375
               1377  1380  1383  1387  1390  1392  1395  1398  1402  1405  1407  1410  1413  1417  1420
               1422  1425  1428  1432  1435  1437  1440  1443  1447  1450  1452  1455  1458  1462  1465
               1467  1470  1473  1477  1480  1482  1485  1488  1495  1499  1503  1505  1508  1514  1518
               1522  1524  1527  1530  1537  1541  1554  1557  1560  1562  1564  1567  1576  1579  1581
               1584  1588  1590  1596  1600  1613  1616  1619  1622  1625  1628  1638  1642  1652  1656
               1661  1663  1665  1668  1672  1680  1683  1685  1687  1690  1693  1695  1697  1717  1730
               1842  1861  1961  2106  2482  2490  2491  2492  2493  2494  2495  2496  2497  2498  2499
               2500  2501  2539  2593  2597  2601  2829  2952  3015  3039
.MACRO           17    18    19    20    22    24    25    26    27    28    29    30   182   230   453
                469   515   926  2482
.MCALL           30   168   277   499
.NLIST           14    30   168   181   189   267   269   270   271   277   280   397   499   545   549
                553   559   566   577   581   584   586   588   589   593   596   598   602   605   611
                614   617   621   624   626   630   634   637   639   643   647   653   656   658   664
                668   674   677   679   687   691   694   696   699   701   704   706   709   711   714
                716   719   721   724   726   729   731   734   736   739   741   744   748   752   756
                759   761   764   766   769   771   774   776   779   781   784   786   789   791   794
                796   799   801   804   806   809   812   816   820   823   826   829   831   836   840
                843   845   849   853   856   858   862   866   869   872   874   879   883   886   888
                892   896   899   902   904   908   912   916   919   922   926   930   937   940   944
                946   950   954   957   960   963   965   970   974   977   980   983   985   990   994
                997  1000  1003  1005  1009  1013  1024  1026  1037  1040  1042  1053  1057  1066  1069
               1072  1075  1077  1086  1090  1099  1102  1109  1112  1115  1123  1127  1134  1137  1141
               1144  1146  1149  1152  1156  1159  1161  1164  1167  1171  1174  1176  1179  1182  1186
               1189  1191  1194  1197  1201  1204  1206  1209  1212  1216  1219  1221  1224  1227  1231
               1234  1236  1239  1242  1246  1249  1251  1254  1257  1261  1264  1266  1269  1272  1276
               1279  1281  1284  1287  1291  1294  1296  1299  1302  1309  1313  1320  1323  1327  1330
               1332  1335  1338  1342  1345  1347  1350  1353  1357  1360  1362  1365  1368  1372  1375
               1377  1380  1383  1387  1390  1392  1395  1398  1402  1405  1407  1410  1413  1417  1420
               1422  1425  1428  1432  1435  1437  1440  1443  1447  1450  1452  1455  1458  1462  1465
               1467  1470  1473  1477  1480  1482  1485  1488  1495  1499  1503  1505  1508  1514  1518
               1522  1524  1527  1530  1537  1541  1554  1557  1560  1562  1564  1567  1576  1579  1581
               1584  1588  1590  1596  1600  1613  1616  1619  1622  1625  1628  1638  1642  1652  1656
               1661  1663  1665  1668  1672  1680  1683  1685  1687  1690  1693  1695  1697  1717  1730
               1842  1861  1961  2106  2482  2490  2491  2492  2493  2494  2495  2496  2497  2498  2499
               2500  2501  2539  2593  2597  2601  2829  2952  3015  3039
.PAGE           171   230   345
.PSECT           14
.REM              1    14    40
.REPT           189   269   694   759  1138  1324
```

```
.SBTTL      58   171   183   192   197   208   230   277   345   477   577   589   602   617   630
           643   664   687   752   816   836   849   862   879   892   908   926   950   970   990
          1009  1053  1086  1123  1309  1495  1514  1537  1596  1638  1652  1704  1835  1851  1916
          1966  2004  2144  2192  2270  2321  2401  2459  2482
.TITLE      30
.WORD      189   190   191   205   224   225   226   227   228   229   238   241   242   243   244
           247   248   249   250   251   252   253   256   257   258   267   269   270   282   283
           284   285   286   287   288   289   293   294   295   308   312   315   318   319   320
           321   322   323   324   325   326   327   328   329   330   331   332   333   334   335
           336   337   338   339   340   428   431   433   435   437   439   441   443   445   446
           448   450   584   586   588   596   598   611   614   624   626   637   639   653   656
           658   674   677   679   694   696   699   701   704   706   709   711   714   716   719
           721   724   726   729   731   734   736   739   741   744   748   759   761   764   766
           769   771   774   776   779   781   784   786   789   791   794   796   799   801   804
           806   809   812   823   826   829   831   843   845   856   858   869   872   874   886
           888   899   902   904   916   919   922   937   940   944   946   957   960   963   965
           977   980   983   985   997  1000  1003  1005  1024  1026  1037  1040  1042  1066  1069
          1072  1075  1077  1099  1102  1109  1112  1115  1134  1137  1141  1144  1146  1149  1152
          1156  1159  1161  1164  1167  1171  1174  1176  1179  1182  1186  1189  1191  1194  1197
          1201  1204  1206  1209  1212  1216  1219  1221  1224  1227  1231  1234  1236  1239  1242
          1246  1249  1251  1254  1257  1261  1264  1266  1269  1272  1276  1279  1281  1284  1287
          1291  1294  1296  1299  1302  1320  1323  1327  1330  1332  1335  1338  1342  1345  1347
          1350  1353  1357  1360  1362  1365  1368  1372  1375  1377  1380  1383  1387  1390  1392
          1395  1398  1402  1405  1407  1410  1413  1417  1420  1422  1425  1428  1432  1435  1437
          1440  1443  1447  1450  1452  1455  1458  1462  1465  1467  1470  1473  1477  1480  1482
          1485  1488  1503  1505  1508  1522  1524  1527  1530  1554  1557  1560  1562  1564  1567
          1576  1579  1581  1584  1588  1590  1613  1616  1619  1622  1625  1628  1661  1663  1665
          1668  1672  1680  1683  1685  1687  1690  1693  1695  1697  1722  1725  1737  2003  2170
          2175  2268  2305  2307  2350  2397  2433  2489  2545  2642  2662  2797  2863  2864  2871
          2906  2916  2937  2938  2952  2962  2963  2993  2997  3027  3028  3033  3044  3045  3048
          3049  3079  3080  3086  3087  3093

        000000

ERRORS DETECTED:  0
```

```
*DRLPJ,DRLPJ/SOL/CRF=DRLPA.MAC,DRLPJ
RUN-TIME:  28 16 2 SECONDS
CORE USED:  39K
```