

# 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

This section contains a grid of 12 columns and 12 rows of small, illegible diagnostic test data tables. Each table appears to be a structured data set, possibly representing test results for various components or systems. The text within these tables is too small to be read.

11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100



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

-----

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 CHARACTERISTICS. 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 RE-CABLING IS NEEDED. SOME TESTS 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 EQUIPMENT

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 TYPINGS                |
| SW 11 = 1 | INHIBIT INTERACTIONS                 |
| 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

-----

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 REPORTING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE"



D01

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. MISCELLANEOUS

## 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 "\$VECT1" 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:
- 2) THE DIALOG TO EXAMINE A LOCATION IS AS FOLLOWS:

```
E OR D      "E"
DEVICE ADDR= "OCTAL ADDR"
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.

SEQ 0004



## 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. MB254 AND MB200-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 MB254 FALLS INTO THE GROUP "B" CATEGORY.



# F01

THE LPA11-KX DIAGNOSTIC KIT WILL INCLUDE:

SEQ 0006

| <u>OPTION</u> | <u>GROUP</u> | <u>DIAG. #</u> | <u>DIAG. TITLE</u>                       |
|---------------|--------------|----------------|--|
| 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<br>MICRO-CPU R/W TEST |
|               | B            | MD-11-DZLPM    | LPA/M8200-YC<br>JMP+ROM READ TEST        |

|      |  |
|------|--|
| 59   | BASIC DEFINITIONS  |
| 172  | OPERATIONAL SWITCH SETTINGS  |
| 184  | TRAP CATCHER   |
| 193  | STARTING ADDRESS(ES)   |
| 198  | ACT11 HOOKS  |
| 209  | APT PARAMETER BLOCK  |
| 231  | COMMON TAGS  |
| 278  | APT MAILBOX-ETABLE   |
| 346  | ERROR POINTER TABLE  |
| 478  | INITIALIZE THE COMMON TAGS   |
| 578  | T1 TEST FOR NO BUS ERRORS  |
| 591  | T2 TEST THAT OUTPUT REG. CAN HOLD #-1                              |
| 605  | T3 TEST THAT RESET CLEARS OUTPUT REG.                              |
| 621  | T4 TEST THAT OUTPUT REG. CAN HOLD #52525                           |
| 635  | T5 TEST THAT OUTPUT REG. CAN HOLD #125252                          |
| 649  | T6 FLOAT A 1 ACROSS THE OUTPUT REGISTER                            |
| 671  | T7 FLOAT A 0 ACROSS THE OUTPUT REGISTER                            |
| 695  | T10 TEST FOR SLOW OUTPUT GATES WITH #125252                        |
| 761  | T11 TEST FOR SLOW OUTPUT GATES WITH #52525                         |
| 826  | T12 TEST RELAY #1 STATUS BIT                                       |
| 847  | T13 TEST RELAY #2 STATUS BIT                                       |
| 861  | T14 TEST OUTPUT DATA ACCEPT FLAG                                   |
| 875  | T15 TEST OUTPUT INTERRUPT ENABLE                                   |
| 893  | T16 TEST INPUT DATA READY FLAG                                     |
| 907  | T17 TEST INPUT INTERRUPT ENABLE                                    |
| 924  | T20 TEST THAT RESET CLEARS THE DIGITAL STATUS REG.                 |
| 943  | T21 TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED              |
| 968  | T22 TEST INPUT WITH #-1  |
| 989  | T23 TEST INPUT WITH #52525   |
| 1010 | T24 TEST INPUT WITH #125252  |
| 1030 | T25 FLOAT A 1 ACROSS NON-LATCHING INPUT BITS                       |
| 1075 | T26 FLOAT A 1 ACROSS LATCHING INPUT BITS                           |
| 1109 | T27 FLOAT A 0 ACROSS LATCHING INPUT BITS                           |
| 1147 | T30 TEST FOR SLOW INPUT GATES WITH #125252                         |
| 1334 | T31 TEST FOR SLOW INPUT GATES WITH #52525                          |
| 1521 | T32 TEST THAT WHEN OUTPUTTING THE INPUT DATA READY FLAG SETS       |
| 1541 | T33 TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET |
| 1565 | T34 TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG            |
| 1625 | T35 TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG |
| 1668 | T36 END OF THE PROGRAM   |
| 1683 | T37 MISC. EXTERNAL LOGIC TEST                                      |
| 1736 | END OF PASS ROUTINE  |
| 1867 | GETSWR SUBROUTINE  |
| 1883 | SCOPE HANDLER ROUTINE  |
| 1949 | ERROR HANDLER ROUTINE  |
| 1999 | READ AN OCTAL NUMBER FROM THE TTY                                  |
| 2037 | TTY INPUT ROUTINE  |
| 2177 | ERROR MESSAGE TYPEOUT ROUTINE                                      |
| 2225 | BINARY TO OCTAL (ASCII) AND TYPE                                   |
| 2303 | POWER DOWN AND UP ROUTINES   |
| 2354 | TYPE ROUTINE   |
| 2434 | APT COMMUNICATIONS ROUTINE   |
| 2492 | TRAP DECODER   |
| 2515 | TRAP TABLE   |



.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 AVAILABLE FOR TESTING THE LPA SYSTEM.

GOOD LUCK !

[  
.GLOBL DRLPX2

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30

30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83

```
.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. FROM 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
177774 .EQUIV PS,PSW
177772 STKLMT= 177774 ;;STACK LIMIT REGISTER
177570 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



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 BASIC DEFINITIONS

```

84      000006      R6=      %6      ;; GENERAL REGISTER
85      000007      R7=      %7      ;; GENERAL REGISTER
86      000006      SP=      %6      ;; STACK POINTER
87      000007      PC=      %7      ;; PROGRAM COUNTER
88
89      ;*PRIORITY LEVEL DEFINITIONS
90      000000      PR0=      0      ;; PRIORITY LEVEL 0
91      000040      PR1=      40     ;; PRIORITY LEVEL 1
92      000100      PR2=      100    ;; PRIORITY LEVEL 2
93      000140      PR3=      140    ;; PRIORITY LEVEL 3
94      000200      PR4=      200    ;; PRIORITY LEVEL 4
95      000240      PR5=      240    ;; PRIORITY LEVEL 5
96      000300      PR6=      300    ;; PRIORITY LEVEL 6
97      000340      PR7=      340    ;; PRIORITY LEVEL 7
98
99      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100     100000      SW15=     100000
101     040000      SW14=     40000
102     020000      SW13=     20000
103     010000      SW12=     10000
104     004000      SW11=     4000
105     002000      SW10=     2000
106     001000      SW09=     1000
107     000400      SW08=     400
108     000200      SW07=     200
109     000100      SW06=     100
110     000040      SW05=     40
111     000020      SW04=     20
112     000010      SW03=     10
113     000004      SW02=     4
114     000002      SW01=     2
115     000001      SW00=     1
116     .EQUIV      SW09,SW9
117     .EQUIV      SW08,SW8
118     .EQUIV      SW07,SW7
119     .EQUIV      SW06,SW6
120     .EQUIV      SW05,SW5
121     .EQUIV      SW04,SW4
122     .EQUIV      SW03,SW3
123     .EQUIV      SW02,SW2
124     .EQUIV      SW01,SW1
125     .EQUIV      SW00,SW0
126
127     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
128     100000      BIT15=    100000
129     040000      BIT14=    40000
130     020000      BIT13=    20000
131     010000      BIT12=    10000
132     004000      BIT11=    4000
133     002000      BIT10=    2000
134     001000      BIT09=    1000
135     000400      BIT08=    400
136     000200      BIT07=    200
137     000100      BIT06=    100

```

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 BASIC DEFINITIONS

```

138      000040      BIT05= 40
139      000020      BIT04= 20
140      000010      BIT03= 10
141      000004      BIT02= 4
142      000002      BIT01= 2
143      000001      BIT00= 1
144      .EQUIV      BIT09,BIT9
145      .EQUIV      BIT08,BIT8
146      .EQUIV      BIT07,BIT7
147      .EQUIV      BIT06,BIT6
148      .EQUIV      BIT05,BIT5
149      .EQUIV      BIT04,BIT4
150      .EQUIV      BIT03,BIT3
151      .EQUIV      BIT02,BIT2
152      .EQUIV      BIT01,BIT1
153      .EQUIV      BIT00,BIT0
154
155      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
156      000004      ERRVEC= 4      ;: TIME OUT AND OTHER ERRORS
157      000010      RESVEC= 10     ;: RESERVED AND ILLEGAL INSTRUCTIONS
158      000014      TBITVEC=14     ;: "T" BIT
159      000014      TRTVEC= 14     ;: TRACE TRAP
160      000014      BPTVEC= 14     ;: BREAKPOINT TRAP (BPT)
161      000020      IOTVEC= 20     ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
162      000024      PWRVEC= 24     ;: POWER FAIL
163      000030      EMTVEC= 30     ;: EMULATOR TRAP (EMT) **ERROR**
164      000034      TRAPVEC=34     ;: "TRAP" TRAP
165      000060      TKVEC= 60      ;: TTY KEYBOARD VECTOR
166      000064      TPVEC= 64      ;: TTY PRINTER VECTOR
167      000240      PIRQVEC=240    ;: PROGRAM INTERRUPT REQUEST VECTOR
168      170410      ABASE=170410
169      000350      AVECT1=350
170      000200      APRIOR=200

```



171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224

```

.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
.=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
.=174
DISPREG: .WORD 0 ;; SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;; SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP @#BEGIN ;; JUMP TO STARTING ADDRESS OF PROGRAM
JMP @#BEGIN1 ;; JUMP TO THE RESTART ADDRESS
JMP @#EXTTST ;; JUMP TO THE MISC. TEST

.SBTTL ACT11 HOOKS
;*****
;HOOKS REQUIRED BY ACT11
$SVPC=. ;SAVE PC
.=46
$ENDAD ;; 1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
.=52
.WORD 0 ;; 2)SET LOC.52 TO ZERO
.$SVPC
.=1000
;; RESTORE PC

.SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.$X=. ;; SAVE CURRENT LOCATION
.=24 ;; SET POWER FAIL TO POINT TO START OF PROGRAM
200 ;; FOR APT START UP
.=44 ;; POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR ;; POINT TO APT HEADER BLOCK
.=.$X ;; RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.
$APTHD:
$HIBTS: .WORD 0 ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.

```

000000

000174 000174 000000  
000176 000000

000200 000137 001564  
000204 000137 001570  
000210 000137 011764

000214 000214  
000046 012324 000046  
000052 000052 000052  
000214 000214  
001000 001000

001000 001000  
000024 000200 000024  
000044 000044 000044  
001000 001000 001000

001000 001000 000000

MO1

MACY11 27(654) 15-DEC-77 08:37 PAGE 6

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 APT PARAMETER BLOCK

SEQ 0013

|     |        |        |                |                    |  |
|-----|--------|--------|----------------|--------------------|--|
| 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          | SETEND-\$MAIL/2 ;; | LENGTH MAILBOX-ETABLE(WORDS)                                     |



230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283

.SBTTL COMMON TAGS

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

001100

SCMTAG: . =1100

;; START OF COMMON TAGS

000000  
000  
000  
000000  
000000  
000000  
000000  
000000  
000  
001  
000000  
000000  
000000  
000000  
000000  
000000  
000000  
000000  
000  
000  
000000  
177570  
177570  
177560  
177562  
177564  
177566  
000  
002  
012  
000  
000000  
000000  
000000  
000000  
077  
015  
000012

.WORD 0  
STSNM: .BYTE 0  
SERFLG: .BYTE 0  
SICNT: .WORD 0  
SLPADR: .WORD 0  
SLPERR: .WORD 0  
SERTTL: .WORD 0  
SITEMB: .BYTE 0  
SERMAX: .BYTE 1  
SERRPC: .WORD 0  
SGDADR: .WORD 0  
SBDADR: .WORD 0  
SGDDAT: .WORD 0  
SBDDAT: .WORD 0  
SAUTOB: .BYTE 0  
SINTAG: .BYTE 0  
SWR: .WORD DSWR  
DISPLAY: .WORD DDI:P  
STKS: 177560  
STKB: 177562  
STPS: 177564  
STPB: 177566  
SNULL: .BYTE 0  
SFILLS: .BYTE 2  
SFILLC: .BYTE 12  
STPFLG: .BYTE 0  
SREGAD: .WORD 0  
SREGO: .WORD 0  
SREG1: .WORD 0  
STIMES: 0  
SESCAPE: 0  
SQUES: .ASCII /?/  
SCRLF: .ASCII <15>  
SLF: .ASCIZ <12>

;; CONTAINS THE TEST NUMBER  
;; CONTAINS ERROR FLAG  
;; CONTAINS SUBTEST ITERATION COUNT  
;; CONTAINS SCOPE LOOP ADDRESS  
;; CONTAINS SCOPE RETURN FOR ERRORS  
;; CONTAINS TOTAL ERRORS DETECTED  
;; CONTAINS ITEM CONTROL BYTE  
;; CONTAINS MAX. ERRORS PER TEST  
;; CONTAINS PC OF LAST ERROR INSTRUCTION  
;; CONTAINS ADDRESS OF 'GOOD' DATA  
;; CONTAINS ADDRESS OF 'BAD' DATA  
;; CONTAINS 'GOOD' DATA  
;; CONTAINS 'BAD' DATA  
;; RESERVED--NOT TO BE USED  
;; AUTOMATIC MODE INDICATOR  
;; INTERRUPT MODE INDICATOR  
;; ADDRESS OF SWITCH REGISTER  
;; ADDRESS OF DISPLAY REGISTER  
;; TTY KBD STATUS  
;; TTY KBD BUFFER  
;; TTY PRINTER STATUS REG. ADDRESS  
;; TTY PRINTER BUFFER REG. ADDRESS  
;; CONTAINS NULL CHARACTER FOR FILLS  
;; CONTAINS # OF FILLER CHARACTERS REQUIRED  
;; INSERT FILL CHARS. AFTER A "LINE FEED"  
;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)  
;; CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED  
;; CONTAINS ((\$REGAD)+0)  
;; CONTAINS ((\$REGAD)+2)  
;; MAX. NUMBER OF ITERATIONS  
;; ESCAPE ON ERROR ADDRESS  
;; QUESTION MARK  
;; CARRIAGE RETURN  
;; LINE FEED

.SBTTL APT MAILBOX-ETABLE

\*\*\*\*\*  
; EVEN  
\$MAIL: ; APT MAILBOX  
\$MSGTY: .WORD AMSGTY ; MESSAGE TYPE CODE  
\$FATAL: .WORD AFATAL ; FATAL ERROR NUMBER





C02

MACY11 27(654) 15-DEC-77 08:37 PAGE 9

SEQ 0016

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 APT MAILBOX-ETABLE

338 001314 000000  
339 001316 000000  
340 001320 000000  
341  
342  
343 001322  
344

\$DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13  
\$DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14  
\$DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15

SETEND:

.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

359 001322

\$ERRTB:

361 001322 012376  
362 001324 013016  
363 001324 013016  
364 001326 013202  
365 001330 013230

;ITEM 1  
EM1 ;:STATUS REGISTER IN ERROR  
DH1 ;:ERRPC STATUS EXPECTED  
DT1 ;:SERRPC \$BDDAT \$GDDAT  
DF1

368 001332 012427  
369 001334 013047  
370 001336 013202  
371 001340 013230

;ITEM 2  
EM2 ;:INPUT REGISTER IN ERROR  
DH2 ;:ERRPC INPUT EXPECTED  
DT1 ;:SERRPC \$BDDAT \$GDDAT  
DF1

374 001342 012457  
375 001344 013100  
376 001346 013202  
377 001350 013230

;ITEM 3  
EM3 ;:OUTPUT REGISTER IN ERROR  
DH3 ;:ERRPC OUTPUT EXPECTED  
DT1 ;:SERRPC \$BDDAT \$GDDAT  
DF1

380 001352 012510  
381 001354 013131  
382 001356 013212  
383 001360 013230

;ITEM 4  
EM4 ;:INPUT FAILED TO INTERRUPT  
DH4 ;:ERRPC  
DT4 ;:SERRPC  
DF1

386 001362 012542  
387 001364 013131  
388 001366 013212  
389 001370 013230

;ITEM 5  
EM5 ;:OUTPUT FAILED TO INTERRUPT  
DH4 ;:ERRPC  
DT4 ;:SERRPC  
DF1

392 001372 012575  
393 001374 013131  
394 001376 013212  
395 001400 013230

;ITEM 6  
EM6 ;:UNEXPECTED INTERRUPT  
DH4 ;:ERRPC  
DT4 ;:SERRPC  
DF1

398 001402 012622

;ITEM 7  
EM7 ;:OPERATOR INTERVENTION ERROR



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 ERROR POINTER TABLE

```

399 001404 013131          DH4          :ERRPC
400 001406 013212          DT4          :$ERRPC
401 001410 013230          DF1
402
403          ;ITEM      10
404 001412 012656          EM10       ; INTERRUPT INPUT BIT FAILED TO SET INPUT READY
405 001414 013137          DH10       ;ERRPC      STATUS      EXPECTED      INPUT BIT
406 001416 013216          DT10       ;$ERRPC    $BDDAT    $GDDAT    BRLEV3
407 001420 013234          DF10
408
409          ;ITEM      11
410 001422 012741          EM11       ;NON-INTERRUPTING INPUT BIT SET INPUT READY
411 001424 013137          DH10       ;ERRPC      STATUS      EXPECTED      INPUT BIT
412 001426 013216          DT10       ;$ERRPC    $BDDAT    $GDDAT    BRLEV3
413
414 001430 170410          DRADD:     170410      ;DRA STARTING ADDRESS
415 001432 000350          DRIV:      350        ;DRA STARTING INTERRUPT VECTOR
416 001434 000200          DRBRL:     200        ;DRA BR LEVEL
417
418
419
420          ;
421          ; ADDRESS OF KMC-11 OF LPA-11      THE ADDR FOR KMADO MAY BE
422          ;                               ; CHANGED BY THE USER TO REFLECT
423          ;                               ; A DIFFERENT KMC-11 ADDR. THE
424          ;                               ; REST OF THE ADDRESSES WILL
425          ;                               ; BE CHANGED BY THE PROGRAM.
426          ;
427          ;
428          ;
429          ;
430          ;
431          ;
432          ;
433          ;
434          ;
435          ;
436          ;
437          ;
438          ;
439          ;
440          ;
441          ;
442          ;
443          ;
444          ;
445 001436 170460          LPCI:
446 001436 170460          KMADO:     .WORD     170460      ;BASE KMC ADDR. MAY BE PATCHED BY USER.
447
448 001440 170461          LPMR:
449 001440 170461          KMAD1:     .WORD     170460+1      ;>DO NOT      <;KMC-CSR ADDR
450 001442 170462          LPCO:
451 001442 170462          KMAD2:     .WORD     170460+2      ;>PATCH      <;
452 001444 170463          LPSO:
453 001444 170463          KMAD3:     .WORD     170460+3      ;>THIS AREA   <
454 001446 170464          LPADL:
455 001446 170464          KMAD4:     .WORD     170460+4      ;
456 001450 170465          LPADH:
457 001450 170465          KMAD5:     .WORD     170460+5      ;>DO NOT      <
458 001452 170466          LPMS1:
459 001452 170466          KMAD6:     .WORD     170460+6      ;>PATCH      <
460 001454 170467          LPMS2:
461 001454 170467          KMAD7:     .WORD     170460+7      ;>THIS AREA   <
462
463 001456 000350          VECTOR:    .WORD     AVECT1&777      ;BASE VECTOR OF KMC
464 001460 000354          VECTPS:    .WORD     4+AVECT1&777    ;VECTR ADDR.+2
465
466 001462 000004          VERSN:     .WORD     4              ;CURRENT VERSION NUMBER OF MICROCODE.
467
468 001464 000000          .DVLs:     .WORD     0              ;/DEVICE LIST OF I/O ADDR. DEFINED
469 001466 000020          .BLKW:     .BLKW    16.           ;/BY INIT.
470
471
472

```

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 ERROR POINTER TABLE

|     |        |        |        |        |
|-----|--------|--------|--------|--------|
| 453 | 001526 | 170410 |        |        |
| 454 | 001530 | 170412 |        |        |
| 455 | 001532 | 170414 |        |        |
| 456 | 001534 | 170415 |        |        |
| 457 | 001536 | 000000 |        |        |
| 458 | 001540 | 000000 |        |        |
| 459 | 001542 | 000310 |        |        |
| 460 | 001544 | 000312 |        |        |
| 461 | 001546 | 000314 |        |        |
| 462 | 001550 | 000316 |        |        |
| 463 | 001552 | 000200 |        |        |
| 464 | 001554 | 000000 |        |        |
| 465 | 001556 | 000000 |        |        |
| 466 | 001560 | 000000 |        |        |
| 467 | 001562 | 000000 |        |        |
| 468 |        |        |        |        |
| 469 |        |        |        |        |
| 470 |        |        |        |        |
| 471 |        |        |        |        |
| 472 | 001564 |        |        |        |
| 473 | 001564 | 005000 |        |        |
| 474 | 001566 | 000402 |        |        |
| 475 | 001570 | 012700 | 177777 |        |
| 476 | 001574 | 000240 |        |        |
| 477 |        |        |        |        |
| 478 |        |        |        |        |
| 479 | 001576 | 012706 | 001100 |        |
| 480 | 001602 | 005026 |        |        |
| 481 | 001604 | 022706 | 001140 |        |
| 482 | 001610 | 001374 |        |        |
| 483 | 001612 | 012706 | 001100 |        |
| 484 |        |        |        |        |
| 485 | 001616 | 012737 | 013332 | 000020 |
| 486 | 001624 | 012737 | 000340 | 000022 |
| 487 | 001632 | 012737 | 013614 | 000030 |
| 488 | 001640 | 012737 | 000340 | 000032 |
| 489 | 001646 | 012737 | 016210 | 000034 |
| 490 | 001654 | 012737 | 000340 | 000036 |
| 491 | 001662 | 012737 | 015240 | 000024 |
| 492 | 001670 | 012737 | 000340 | 000026 |
| 493 | 001676 | 013737 | 012304 | 012276 |
| 494 | 001704 | 005037 | 001166 |        |
| 495 | 001710 | 005037 | 001170 |        |
| 496 | 001714 | 112737 | 000001 | 001115 |
| 497 | 001722 | 012737 | 001722 | 001106 |
| 498 | 001730 | 012737 | 001730 | 001110 |
| 499 |        |        |        |        |
| 500 |        |        |        |        |
| 501 | 001736 | 013746 | 000004 |        |
| 502 | 001742 | 012737 | 001776 | 000004 |
| 503 | 001750 | 012737 | 177570 | 001140 |
| 504 | 001756 | 012737 | 177570 | 001142 |
| 505 | 001764 | 022777 | 177777 | 177146 |
| 506 | 001772 | 001012 |        |        |

```

GRSTAT: 170410 ;DR STATUS
GRDAI: 170412 ;INPUT REG.
GRDIO: 170414 ;OUTPUT REG.
GRBHI0: 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
STMDAT: 0

;*****
;DIGITAL I-O LOGIC TEST
;*****
BEGIN:
CLR RO
BR RBEG ;
BEGIN1: MOV #-1,RO
RBEG: NOP
.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS (SCMTAG) AREA
MOV #SCMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;:CLEAR MEMORY LOCATION
CMP #SWR,R6 ;:DONE?
BNE -6 ;:LOOP BACK IF NO
MOV #STACK,SP ;:SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV #SSCOPE,#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
MOV #340,#IOTVEC+2 ;:LEVEL 7
MOV #SEAROR,#EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
MOV #340,#EMTVEC+2 ;:LEVEL 7
MOV #STRAP,#TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
MOV #340,#TRAPVEC+2 ;:LEVEL 7
MOV #SPWRDN,#PWRVEC ;:POWER FAILURE VECTOR
MOV #340,#PWRVEC+2 ;:LEVEL 7
MOV SENDCT,SEOPCT ;:SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
MOV #1,$SERMAX ;:ALLOW ONE ERROR PER TEST
MOV #,$SLPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$SLPERR ;: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.
MOV #ERRVEC, -(SP) ;:SAVE ERROR VECTOR
MOV #64,$ERRVEC ;:SET UP ERROR VECTOR
MOV #DSWR,$SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP,$DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
CMP #-1,$SWR ;:TRY TO REFERENCE HARDWARE SWR
BNE 66$ ;:BRANCH IF NO TIMEOUT TRAP OCCURRED

```



```

507                                     ;; AND THE HARDWARE SWR IS NOT = -1
508 001774 000403 BR 65$ ;; BRANCH IF NO TIMEOUT
509 001776 012716 002004 64$: MOV #65$, (SP) ;; SET UP FOR TRAP RETURN
510 002002 000002 RTI
511 002004 012737 000176 001140 65$: MOV #SWREG, SWR ;; POINT TO SOFTWARE SWR
512 002012 012737 000174 001142 MOV #DISPREG, DISPLAY
513 002020 012637 000004 66$: MOV (SP)+, #ERRVEC ;; RESTORE ERROR VECTOR
514
515 002024 005037 001204 CLR $PASS ;; CLEAR PASS COUNT
516 002030 132737 000200 001217 BITB #APTSIZE, $ENVM ;; TEST USER SIZE UNDER APT
517 002036 001403 BEQ 67$ ;; YES, USE NON-APT SWITCH
518 002040 012737 001220 001140 MOV #SSWREG, SWR ;; NO, USE APT SWITCH REGISTER
519 002046
520 002046 013737 001252 001430 67$: DONOT: MOV $BASE, DRADD
521
522 ;; THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
523 ;;
524
525 002054 010046 MOV RO, -(SP)
526 002056 010146 MOV R1, -(SP)
527 002060 013700 001436 MOV KMADD, RO ;; GET KMC-11 ADDRESS.
528 002064 012701 001440 MOV #KMA1, R1 ;; GET ADDR. OF ADDR. LIST.
529
530 002070 005200 64$: INC RO ;; UPDATE ADDR.
531 002072 010021 MOV RO, (1)+ ;; WRITE ADDR.
532 002074 020127 001456 CMP R1, #KMA7+2 ;; DONE ALL ADDRESSES?
533 002100 001373 BNE 64$ ;; NO - DO NEXT ADDR.
534 002102 005037 001464 CLR .DVLS ;; CLR ADDR. LIST.
535 002106 012601 MOV (SP)+, R1
536 002110 012600 MOV (SP)+, RO
537 002112 005700 TST RO ;; TEST RO
538 002114 001402 BEQ 2$ ;; BR IF CLEARED
539 002116 000137 002610 1$: JMP 3$ ;; JUMP IF SET
540 002122 005737 000042 2$: TST #42 ;; TEST IF IN "CHAIN MODE"
541 002126 001373 BNE 1$ ;; BR IF YES
542 002130 104401 002136 TYPE ,66$ ;; TYPE ASCIZ STRING
543 002134 000432 BR 65$ ;; GET OVER THE ASCIZ
544
545 002222 66$: .ASCIZ <15><12><15><12> LPA/LPS-11-DRA DIGITAL INPUT OUTPUT LOGIC TEST#
546 002222 104401 002230 65$: TYPE ,68$ ;; TYPE ASCIZ STRING
547 002226 000413 BR 67$ ;; GET OVER THE ASCIZ
548
549 002256 68$: .ASCIZ <15><12>/MAINDEC-11-DRLPJ-A/
550 002256 104401 002264 67$: TYPE ,70$ ;; TYPE ASCIZ STRING
551 002262 000435 BR 69$ ;; GET OVER THE ASCIZ
552
553 002356 69$: .ASCIZ <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-LATCHING BITS/
554 002356 004737 013254 JSR PC, GETSWR ;; GET THE SWITCH REGISTER VALUE
555 002362 017737 176552 001536 MOV @SWR, NOTLCH ;; SAVE SWITCHES
556 002370 104401 002376 TYPE ,72$ ;; TYPE ASCIZ STRING
557 002374 000437 BR 71$ ;; GET OVER THE ASCIZ
558
559 002474 72$: .ASCIZ <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-INTERRUPTING BITS/
560 71$:

```

```

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$: .ASCIZ <15><12>/SET SWITCH REGISTER WITH THE DESIRED PROGRAM OPTIONS/
566                                     73$:
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                                     ;*      MOV    $TMDAT,@GRDAI  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
585                                     ;*      MOV    $TMDAT,@GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
586
587                                     ;;*****
588                                     ;*TEST 2          TEST THAT OUTPUT REG. CAN HOLD #-1
589                                     ;;*****
590
591                                     TST2:  SCOPE
592 002730 000004          MOV    #-1,$GDDAT        ;LOAD EXPECTED
593 002732 012737 177777 001124    ;*      MOV    $GDDAT,@GRDIO ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
594
595                                     ;*      MOV    @GRDIO,$BDDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
596                                     ;*      CMP    $GDDAT,$BDDAT ;COMPARE
597                                     BEQ    TST3              ;;BR IF EQUAL
598 002760 023737 001124 001126    ERROR  3                ;REG WILL NOT HOLD ONES
599 002766 001401
600 002770 104003
601
602                                     ;;*****
603                                     ;*TEST 3          TEST THAT RESET CLEARS OUTPUT REG.
604                                     ;;*****
605
606 002772 000004          TST3:  SCOPE
607 002774 012737 000004 001166    MOV    #4,$TIMES        ;;DO 4 ITERATIONS
608 003002 005037 001124          CLR    $GDDAT
609 003006 012737 177777 001562    MOV    #-1,$TMDAT
610
611                                     ;*      MOV    $TMDAT,@GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
612                                     JSR    PC,$RESET
613
614                                     ;*      MOV    @GRDIO,$BDDAT ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
615                                     TST    $BDDAT

```



```

615 003044 001401          BEQ      TST4          ;;BR IF EQUAL
616 003046 104003          ERROR     3
617                                     ;:*****
618                                     ;:*TEST 4          TEST THAT OUTPUT REG. CAN HOLD #52525
619                                     ;:*****
620 003050 000004          TST4:    SCOPE
621 003052 012737 052525 001124      MOV      #52525,$GDDAT          ;LOAD EXPECTED VALUE
622                                     ;*
623                                     ;*      MOV      $GDDAT,@GRDIO      ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
624                                     ;*
625                                     ;*      MOV      @GRDIO,$BDDAT      ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
626 003100 023737 001124 001126      CMP      $GDDAT,$BDDAT          ;COMPARE
627 003106 001401          BEQ      TST5          ;;BR IF EQUAL
628 003110 104003          ERROR     3          ;DATA NOT=52525
629
630                                     ;:*****
631                                     ;:*TEST 5          TEST THAT OUTPUT REG. CAN HOLD #125252
632                                     ;:*****
633 003112 000004          TST5:    SCOPE
634 003114 012737 125252 001124      MOV      #125252,$GDDAT        ;LOAD EXPECTED VALUE
635                                     ;*
636                                     ;*      MOV      $GDDAT,@GRDIO      ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
637                                     ;*
638                                     ;*      MOV      @GRDIO,$BDDAT      ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
639 003142 023737 001124 001126      CMP      $GDDAT,$BDDAT          ;COMPARE
640 003150 001401          BEQ      TST6          ;;BR IF EQUAL
641 003152 104003          ERROR     3          ;DATA NOT=125252
642
643                                     ;:*****
644                                     ;:*TEST 6          FLOAT A 1 ACROSS THE OUTPUT REGISTER
645                                     ;:*****
646 003154 000004          TST6:    SCOPE
647 003156 012737 003172 001110      MOV      #1$, $LPERR          ;LOAD SCOPE ERROR RETURN
648 003164 012737 000001 001124      MOV      #BIT0,$GDDAT          ;LOAD EXPECTED VALUE
649                                     ;*
650 003172 012737 000000 001562 1$:    MOV      #0,$TMDAT          ;CLEAR OUTPUT
651                                     ;*
652                                     ;*      MOV      $TMDAT,@GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
653 003210 053737 001124 001562      BIS      $GDDAT,$TMDAT          ;READ OUTPUT REG.
654                                     ;*
655                                     ;*      MOV      $TMDAT,@GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
656                                     ;*
657                                     ;*      MOV      @GRDIO,$BDDAT      ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
658 003236 023737 001124 001126      CMP      $GDDAT,$BDDAT          ;TEST RESULTS
659 003244 001401          BEQ      2$          ;BR IF EQUAL ?
660 003246 104003          ERROR     3
661
662 003250 006337 001124          2$:     ASL      $GDDAT          ;SHIFT EXPECTED DATA
663 003254 001346          BNE      1$          ;BR UNTIL DONE
664
665                                     ;:*****
666                                     ;:*TEST 7          FLOAT A 0 ACROSS THE OUTPUT REGISTER
667                                     ;:*****
668 003256 000004          TST7:    SCOPE
669 003260 012737 003274 001110      MOV      #1$, $LPERR          ;LOAD SCOPE ERROR RETURN

```

```

669 003266 012737 000001 001124      MOV      #BIT0,$GDDAT      ;LOAD EXPECTED VALUE
670
671 003274 012737 177777 001562 1$:  MOV      #-1,$TMDAT      ;LOAD OUTPUT TO A ONE
672
673      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
674 003312 043737 001124 001562      BIC      $GDDAT,$TMDAT    ;CLEAR A BIT
675
676      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
677
678      ;*      MOV      $GRDIO,$BDDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $BDDAT.
679 003340 005137 001126      COM      $BDDAT          ;COMPLEMENT IT
680 003344 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;EQUAL ?
681 003352 001401      BEQ      2$             ;BR IF EQUAL
682 003354 104003      ERROR    3
683
684 003356 006337 001124      2$:  ASL      $GDDAT        ;SHIFT LEFT
685 003362 001344      BNE      1$             ;BRANCH UNTIL DONE
686
687      ;:*****
688      ;:TEST 10      TEST FOR SLOW OUTPUT GATES WITH #125252
689      ;:*****
690 003364 000004      TST10: SCOPE
691 003366 012737 125252 001124      MOV      #125252,$GDDAT  ;LOAD EXPECTED VALUE
692
693      ;*      MOV      $GDDAT,$GRDIO  ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
694
695      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
696 003414 005137 001562      COM      $TMDAT
697
698      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
699
700      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
701 003440 005137 001562      COM      $TMDAT
702
703      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
704
705      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
706 003464 005137 001562      COM      $TMDAT
707
708      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
709
710      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
711 003510 005137 001562      COM      $TMDAT
712
713      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
714
715      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
716 003534 005137 001562      COM      $TMDAT
717
718      ;*      MOV      $TMDAT,$GRDIO  ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
719
720      ;*      MOV      $GRDIO,$TMDAT  ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
721 003560 005137 001562      COM      $TMDAT
722

```



```

723          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
724          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
725          ;*      COM      $TMDAT
726 003604 005137 001562
727          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
728          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
729          ;*      COM      $TMDAT
730 003630 005137 001562
731          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
732          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
733          ;*      COM      $TMDAT
734 003654 005137 001562
735          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
736          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
737          ;*      COM      $TMDAT
738 003700 005137 001562
739          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
740          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
741          ;*      COM      $TMDAT
742 003714 000240
743 003716 000240
744          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
745          ;*      NOP
746          ;*      NOP
747          ;*      MOV      @GRDIO, $BDDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $BDDAT.
748 003730 023737 001124 001126          ;*      CMP      $GDDAT, $BDDAT      ; TEST REGISTER
749 003736 001401
750 003740 104003          ;*      BEQ      TST11          ; ; BR IF CORRECT
751          ;*      ERROR      3
752          ;*****
753          ;*TEST 11          TEST FOR SLOW OUTPUT GATES WITH #52525
754          ;*****
755 003742 000004          †ST11: SCOPE
756 003744 012737 052525 001124          ;*      MOV      #52525, $GDDAT          ;LOAD EXPECTED VALUE
757          ;*      MOV      $GDDAT, @GRDIO      ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
758          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
759          ;*      COM      $TMDAT
760 003772 005137 001562
761          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
762          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
763          ;*      COM      $TMDAT
764 004016 005137 001562
765          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
766          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
767          ;*      COM      $TMDAT
768 004042 005137 001562
769          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
770          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
771          ;*      COM      $TMDAT
772          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
773          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
774          ;*      COM      $TMDAT
775 004066 005137 001562
776          ;*      MOV      $TMDAT, @GRDIO      ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
777          ;*      MOV      @GRDIO, $TMDAT      ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
778          ;*      COM      $TMDAT

```

```

777
778 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
779
780 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
781 004112 005137 001562 COM $TMDAT
782
783 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
784
785 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
786 004136 005137 001562 COM $TMDAT
787
788 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
789
790 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
791 004162 005137 001562 COM $TMDAT
792
793 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
794
795 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
796 004206 005137 001562 COM $TMDAT
797
798 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
799
800 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
801 004232 005137 001562 COM $TMDAT
802
803 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
804
805 ;* MOV @GRDIO, $TMDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
806 004256 005137 001562 COM $TMDAT
807
808 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
809 004272 000240 NOP
810
811 ;* MOV @GRDIO, $BDDAT ;/ READ DEVICE REG GRDIO, PUT DATA IN $BDDAT.
812 004304 023737 001124 001126 CMP $GDDAT, $BDDAT ; TEST PATTERN
813 004312 001401 BEQ TST12 ; ; BR IF EQUAL
814 004314 104003 ERROR 3 ; ; OUTPUT REGISTER IN ERROR
815
816 ; ; *****
817 ; ; *TEST 12 TEST RELAY #1 STATUS BIT
818 ; ; *****
819 004316 000004 TST12: SCOPE
820 004320 012737 000000 001562 MOV #0, $TMDAT
821
822 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
823 004336 012737 177777 001562 MOV #-1, $TMDAT
824
825 ;* MOV $TMDAT, @GRDAI ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
826 004354 012737 000001 001124 MOV #BIT0, $GDDAT ; ; LOAD EXPECTED
827
828 ;* MOV $GDDAT, @GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
829
830 ;* MOV @GRSTAT, $BDDAT ;/ READ DEVICE REG GRSTAT, PUT DATA IN $BDDAT.

```



```

831 004402 033737 001124 001126 BIT $GDDAT,$BDDAT ;TEST IT
832 004410 001001 BNE TST13 ;;BR IF SET
833 004412 104001 ERROR 1 ;ERROR, RELAY 1 FAILED TO SET
834
835
836 ;*****
837 ;*TEST 13 TEST RELAY #2 STATUS BIT
838 ;*****
839 004414 000004 TST13: SCOPE
840 004416 012737 000400 001124 MOV #BIT8,$GDDAT ;LOAD EXPECT
841
842 ;* MOV $GDDAT,$GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
843
844 ;* MOV $GRSTAT,$BDDAT ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
845 004444 033737 001124 001126 BIT $GDDAT,$BDDAT
846 004452 001001 BNE TST14 ;;BR IF SET
847 004454 104001 ERROR 1 ;ERROR, RELAY 2 FAILED TO SET
848
849 ;*****
850 ;*TEST 14 TEST OUTPUT DATA ACCEPT FLAG
851 ;*****
852 004456 000004 TST14: SCOPE
853 004460 012737 100000 001124 MOV #BIT15,$GDDAT ;LOAD EXPECTED
854
855 ;* MOV $GDDAT,$GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
856
857 ;* MOV $GRSTAT,$BDDAT ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
858 004506 033737 001124 001126 BIT $GDDAT,$BDDAT
859 004514 001001 BNE TST15 ;;BR IF SET
860 004516 104001 ERROR 1 ;ERROR, BIT 15 FAILED TO SET
861
862 ;*****
863 ;*TEST 15 TEST OUTPUT INTERRUPT ENABLE
864 ;*****
865 004520 000004 TST15: SCOPE
866 004522 012737 000000 001562 MOV #0,$TMDAT ;CLEAR STATUS
867
868 ;* MOV $TMDAT,$GRSTAT ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
869 004540 012737 040000 001124 MOV #BIT14,$GDDAT ;LOAD EXPECTED
870
871 ;* MOV $GDDAT,$GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
872
873 ;* MOV $GRSTAT,$BDDAT ;/READ DEVICE REG GRSTAT,PUT DATA IN $BDDAT.
874 004566 033737 001124 001126 BIT $GDDAT,$BDDAT
875 004574 001001 BNE TST16 ;;BR IF SET
876 004576 104001 ERROR 1 ;ERROR BIT 14 FAILED TO SET
877
878 ;*****
879 ;*TEST 16 TEST INPUT DATA READY FLAG
880 ;*****
881
882 004600 000004 TST16: SCOPE
883 004602 012737 000200 001124 MOV #BIT7,$GDDAT ;LOAD EXPECTED
884

```

```

885 ;* MOV $GDDAT, @GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
886
887 ;* MOV @GRSTAT, $BDDAT ;/ READ DEVICE REG GRSTAT, PUT DATA IN $BDDAT.
888 004630 023737 001124 001126 ;* CMP $GDDAT, $BDDAT ; COMPARE
889 004636 001401 ;* BEQ TST17 ; ; BR IF EQUAL
890 004640 104001 ;* ERROR 1 ; ERROR, BIT 7 FAILED TO SET
891
892 ;*****
893 ;*TEST 17 TEST INPUT INTERRUPT ENABLE
894 ;*****
895 004642 000004 ;* TST17: SCOPE
896 004644 012737 000000 001562 ;* MOV #0, $TMDAT ; CLEAR STATUS
897
898 ;* MOV $TMDAT, @GRSTAT ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
899 004662 012737 000100 001124 ;* MOV #BIT6, $GDDAT ; LOAD EXPECTED
900
901 ;* MOV $GDDAT, @GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
902
903 ;* MOV @GRSTAT, $BDDAT ;/ READ DEVICE REG GRSTAT, PUT DATA IN $BDDAT.
904 004710 023737 001124 001126 ;* CMP $GDDAT, $BDDAT ; COMPARE
905 004716 001401 ;* BEQ TST20 ; ; BR IF EQUAL
906 004720 104001 ;* ERROR 1 ; ERROR, BIT 6 FAILED TO SET
907
908 ;*****
909 ;*TEST 20 TEST THAT RESET CLEARS THE DIGITAL STATUS REG.
910 ;*****
911 004722 000004 ;* TST20: SCOPE
912 004724 012737 000040 001166 ;* MOV #40, $TIMES ; ; DO 40 ITERATIONS
913 004732 005037 001124 ;* CLR $GDDAT
914
915 ;* MOV $GDDAT, @GRSTAT ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRSTAT
916 004746 012737 140701 001562 ;* MOV #BIT15!BIT14!BIT8!BIT7!BIT6!BIT0, $TMDAT
917
918 ;* MOV $TMDAT, @GRSTAT ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
919 004764 004737 020120 ;* JSR PC, $RESET
920
921 ;* MOV @GRSTAT, $BDDAT ;/ READ DEVICE REG GRSTAT, PUT DATA IN $BDDAT.
922 005000 042737 100000 001126 ;* BIC #BIT15, $BDDAT
923 005006 005737 001126 ;* TST $BDDAT
924 005012 001401 ;* BEQ TST21 ; ; BR IF EQUAL
925 005014 104001 ;* ERROR 1
926
927 ;*****
928 ;*TEST 21 TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
929 ;*****
930 005016 000004 ;* TST21: SCOPE
931 005020 032777 002000 174112 ;* BIT #BIT10, @SWR ; TEST SWITCH BIT
932 005026 001402 ;* BEQ 1$ ; BRANCH IF DOWN
933 005030 000137 011176 ;* JMP DRT21 ; BYPASS SOME TEST USING THE EXTERNAL CABLE
934 005034 012737 000000 001562 1$: MOV #0, $TMDAT
935
936 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
937 005052 012737 177777 001562 ;* MOV #-1, $TMDAT
938

```



```

939 ;* MOV STMDAT,GRDAI ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI
940 005070 005037 001124 CLR SGDDAT ;CLEAR EXPECTED
941 005074 012737 000000 001562 MOV #0,STMDAT ;LOAD THE OUTPUT
942
943 ;* MOV STMDAT,GRDIO ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO
944
945 ;* MOV GRDAI,SBDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN SBDDAT.
946 005122 023737 001124 001126 CMP SGDDAT,SBDDAT ;COMPARE
947 005130 001401 BEQ TST22 ;;BR IF EQUAL
948 005132 104002 ERROR 2 ;ERROR, INPUT DID NOT EQUAL THE OUTPUT REG.
949
950 ;*****
951 ;*TEST 22 TEST INPUT WITH #-1
952 ;*****
953 ST22: SCOPE
954 005134 000004 MOV #0,STMDAT
955 005136 012737 000000 001562
956 ;* MOV STMDAT,GRDIO ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO
957 005154 012737 177777 001562 MOV #-1,STMDAT
958
959 ;* MOV STMDAT,GRDAI ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI
960 005172 012737 177777 001124 MOV #-1,SGDDAT ;LOAD EXPECTED
961
962 ;* MOV SGDDAT,GRDIO ;/ PUT DATA FROM SGDDAT TO DEVICE REG GRDIO
963
964 ;* MOV GRDAI,SBDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN SBDDAT.
965 005220 023737 001124 001126 CMP SGDDAT,SBDDAT ;COMPARE
966 005226 001401 BEQ TST23 ;;BR IF EQUAL
967 005230 104002 ERROR 2 ;ERROR, INPUT DID NOT EQUAL THE OUTPUT
968 ;IS WRAP-AROUND CABLE CONNECTED ???
969
970 ;*****
971 ;*TEST 23 TEST INPUT WITH #52525
972 ;*****
973 ST23: SCOPE
974 005232 000004 MOV #0,STMDAT
975 005234 012737 000000 001562
976 ;* MOV STMDAT,GRDIO ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO
977 005252 012737 177777 001562 MOV #-1,STMDAT
978
979 ;* MOV STMDAT,GRDAI ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI
980 005270 012737 052525 001124 MOV #52525,SGDDAT ;LOAD EXPECTED
981
982 ;* MOV SGDDAT,GRDIO ;/ PUT DATA FROM SGDDAT TO DEVICE REG GRDIO
983
984 ;* MOV GRDAI,SBDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN SBDDAT.
985 005316 023737 001124 001126 CMP SGDDAT,SBDDAT ;COMPARE
986 005324 001401 BEQ TST24 ;;BR IF EQUAL
987 005326 104002 ERROR 2 ;ERROR, INPUT DID NOT EQUAL OUTPUT
988
989 ;*****
990 ;*TEST 24 TEST INPUT WITH #125252
991 ;*****
992

```

MAINDEC-11-DRLPJ-A  
DRLPJ.P11 T24

LPA/LPS-11-DRA DIAGNOSTIC  
TEST INPUT WITH #125252

```

993 005330 000004 TST24: SCOPE
994 005332 012737 000000 001562 MOV #0,$TMDAT
995
996 ;* MOV $TMDAT,$GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
997 005350 012737 177777 001562 MOV #1,$TMDAT
998
999 ;* MOV $TMDAT,$GRDAI ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1000 005366 012737 125252 001124 MOV #125252,$GDDAT ;LOAD EXPECTED
1001
1002 ;* MOV $GDDAT,$GRDIO ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
1003
1004 ;* MOV $GRDAI,$SDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN $SDDAT.
1005 005414 023737 001124 001126 CMP $GDDAT,$SDDAT ;COMPARE
1006 005422 001401 BEQ TST25 ;;BR IF EQUAL
1007 005424 104002 ERROR 2 ;ERROR, INPUT DID NOT EQUAL OUTPUT
1008
1009 ;*****
1010 ;*TEST 25 FLOAT A 1 ACROSS NON-LATCHING INPUT BITS
1011 ;*****
1012 005426 000004 TST25: SCOPE
1013 005430 012737 005456 001110 MOV #2,$SLPERR ;LOAD ERROR SCOPE RETURN
1014
1015 005436 012737 000001 001556 MOV #BIT0,BRLEV2 ;LOAD EXPECTED
1016 005444 013737 001536 001560 MOV NOTLCH,BRLEV3 ;GET NON-LATCH
1017 005452 005137 001560 COM BRLEV3 ;COMPLEMENT
1018 005456 013737 001556 001124 2$: MOV BRLEV2,$GDDAT ;LOAD GOOD
1019 005464 033737 001124 001536 BIT $GDDAT,NOTLCH ;TEST FOR NON-LATCH
1020 005472 001451 BEQ 1$ ;BR IF LATCHING
1021
1022
1023 ;* MOV $GDDAT,$GRDIO ;/ PUT DATA FROM $GDDAT TO DEVICE REG GRDIO
1024
1025 ;* MOV $GRDAI,$SDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN $SDDAT.
1026 005514 043737 001560 001126 BIC BRLEV3,$SDDAT ;MASK TO LATCH BITS
1027 005522 023737 001124 001126 CMP $GDDAT,$SDDAT ;COMPARE
1028 005530 001401 BEQ 3$ ;;BR IF EQUAL
1029 005532 104002 ERROR 2 ;INPUT REGISTER IN ERROR
; WAS CORRECT LATCH/NON-LATCH SUPPLIED ?
1030
1031 ;SUB-TEST CLEAR THE OUTPUT BIT AND TEST THE INPUT DOES NOT LATCH
1032
1033 3$:
1034 005534
1035
1036 ;* MOV $GRDIO,$TMDAT ;/READ DEVICE REG GRDIO,PUT DATA IN $TMDAT.
1037 005544 043737 001124 001562 BIC $GDDAT,$TMDAT
1038
1039 ;* MOV $TMDAT,$GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1040
1041 ;* MOV $GRDAI,$SDDAT ;/READ DEVICE REG GRDAI,PUT DATA IN $SDDAT.
1042 005572 043737 001560 001126 BIC BRLEV3,$SDDAT ;MASK TO LATCH BITS
1043 005600 005037 001124 CLR $GDDAT ;CLEAR EXPECTED
1044 005604 033737 001556 001126 BIT BRLEV2,$SDDAT ;TEST FOR BIT
1045 005612 001401 BEQ 1$ ;;BR IF CLEARED
1046 005614 104002 ERROR 2 ;INPUT BIT LATCHED IN ERROR

```



; WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

; CHANGE PATTERN  
; BR UNTIL DONE

```

1047
1048
1049 005616 006337 001556
1050 005622 001315
1051
1052
1053
1054
1055
1056 005624 000004
1057 005626 012737 005642 001110
1058 005634 012737 000001 001124
1059
1060 005642 033737 001124 001536
1061 005650 001042
1062
1063 005652 012737 000000 001562
1064
1065
1066 005670 012737 177777 001562
1067
1068
1069
1070
1071
1072 005716 012737 000000 001562
1073
1074
1075
1076
1077 005744 023737 001124 001126
1078 005752 001401
1079 005754 104002
1080
1081
1082
1083 005756 006337 001124
1084 005762 001327
1085
1086
1087
1088
1089 005764 000004
1090 005766 012737 006002 001110
1091 005774 012737 000001 001560
1092
1093 006002 033737 001560 001536
1094 006010 001055
1095
1096 006012 012737 000000 001562
1097
1098
1099 006030 012737 177777 001562
1100

```

```

;*****
; *TEST 26      FLOAT A 1 ACROSS LATCHING INPUT BITS
;*****

```

```

↑ST26:  SCOPE
        MOV     #2$, $LPERR      ;LOAD ERROR SCOPE RETURN
        MOV     #BIT0, $GDDAT    ;LOAD EXPECTED VALUE

```

```

2$:    BIT     $GDDAT, NOTLCH    ;TEST FOR NON-LATCHING
        BNE     1$              ;BR IF NON-LATCH

```

```

        MOV     #0, $TMDAT

```

```

;*     MOV     $TMDAT, @GRDIO    ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
        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
        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.
        CMP     $GDDAT, $BDDAT   ;COMPARE
        BEQ     1$              ;;BR IF EQUAL

```

```

        BEQ     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

```

```

1$:    ASL     $GDDAT            ;CHANGE PATTERN
        BNE     2$              ;BR UNTIL DONE

```

```

;*****
; *TEST 27      FLOAT A 0 ACROSS LATCHING INPUT BITS
;*****

```

```

↑ST27:  SCOPE
        MOV     #2$, $LPERR      ;LOAD ERROR SCOPE RETURN
        MOV     #BIT0, $BRLEV3   ;LOAD EXPECTED

```

```

2$:    BIT     $BRLEV3, NOTLCH   ;TEST FOR LATCHING
        BNE     1$              ;BR IF NOT

```

```

        MOV     #0, $TMDAT

```

```

;*     MOV     $TMDAT, @GRDIO    ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
        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          †ST30:  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 | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1156 | 006340 | 012737 | 000000 | 001562 | MOV    | #0,STMDAT    |   |
| 1157 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1158 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1159 |        |        |        |        | MOV    | STMDAT,R1    |   |
| 1160 | 006366 | 013701 | 001562 |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1161 |        |        |        |        | BIS    | R1,STMDAT    |   |
| 1162 |        |        |        |        | ;* MOV | STMDAT,GRDAI | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI |
| 1163 |        |        |        |        | COM    | RO           |   |
| 1164 | 006402 | 050137 | 001562 |        | MOV    | RO,STMDAT    | ;LOAD OUTPUT                                |
| 1165 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1166 | 006416 | 005100 |        |        | MOV    | #0,STMDAT    |   |
| 1167 | 006420 | 010037 | 001562 |        | MOV    | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1168 |        |        |        |        | MOV    | #0,STMDAT    |   |
| 1169 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1170 | 006434 | 012737 | 000000 | 001562 | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1171 |        |        |        |        | MOV    | STMDAT,R1    |   |
| 1172 |        |        |        |        | ;* MOV | STMDAT,GRDAI | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI |
| 1173 |        |        |        |        | COM    | RO           |   |
| 1174 | 006462 | 013701 | 001562 |        | MOV    | RO,STMDAT    | ;LOAD OUTPUT                                |
| 1175 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1176 | 006476 | 050137 | 001562 |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1177 |        |        |        |        | BIS    | R1,STMDAT    |   |
| 1178 |        |        |        |        | ;* MOV | STMDAT,GRDAI | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI |
| 1179 | 006512 | 005100 |        |        | COM    | RO           |   |
| 1180 | 006514 | 010037 | 001562 |        | MOV    | RO,STMDAT    | ;LOAD OUTPUT                                |
| 1181 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1182 | 006530 | 012737 | 000000 | 001562 | MOV    | #0,STMDAT    |   |
| 1183 |        |        |        |        | MOV    | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1184 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1185 | 006556 | 013701 | 001562 |        | MOV    | STMDAT,R1    |   |
| 1186 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1187 | 006572 | 050137 | 001562 |        | BIS    | R1,STMDAT    |   |
| 1188 |        |        |        |        | ;* MOV | STMDAT,GRDAI | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDAI |
| 1189 | 006606 | 005100 |        |        | COM    | RO           |   |
| 1190 | 006610 | 010037 | 001562 |        | MOV    | RO,STMDAT    | ;LOAD OUTPUT                                |
| 1191 |        |        |        |        | ;* MOV | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1192 | 006624 | 012737 | 000000 | 001562 | MOV    | #0,STMDAT    |   |
| 1193 |        |        |        |        | MOV    | STMDAT,GRDIO | ;/ PUT DATA FROM STMDAT TO DEVICE REG GRDIO |
| 1194 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1195 | 006652 | 013701 | 001562 |        | MOV    | STMDAT,R1    |   |
| 1196 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1197 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1198 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1199 |        |        |        |        | ;* MOV | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1200 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1201 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1202 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1203 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1204 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1205 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1206 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1207 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |
| 1208 |        |        |        |        | MOV    | GRDAI,STMDAT | ;/READ DEVICE REG GRDAI,PUT DATA IN STMDAT. |

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



```

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

```

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



```

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

```

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

```

```

1533
1534 ;TEST THAT THE DIGITAL I/O DOES NOT INTERRUPT
1535 ;SET INPUT-OUTPUT FLAGS BUT DO NOT ENABLE INTERRUPTS
1536 011176 DRT21:
1537 ;*****
1538 ;*TEST 34 TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG
1539 ;*****
1540 011176 000004 TST34: SCOPE
1541 011200 032777 002000 167732 BIT #BIT10, @SWR ;TEST CABLE SWITCH
1542 011206 001137 BNE TST35 ;;BYPASS IF NO I/O CABLE
1543
1544 011210 012737 011224 001110 MOV #1$, $LPERR ;LOAD ERROR SCOPE RETURN
1545 011216 012737 000001 001560 MOV #BIT0, BRLEV3 ;LOAD INTERRUPT BIT
1546 011224 005037 001126 1$: CLR $BDDAT ;CLEAR BAD DATA
1547 011230 012737 000200 001124 MOV #BIT7, $GDDAT ;LOAD GOOD DATA
1548
1549 011236 033737 001560 001540 BIT BRLEV3, NOTINT ;TEST IF SET TO INT
1550 011244 001115 BNE 3$ ;;NO TRY NEXT BIT
1551 011246 012737 000000 001562 MOV #0, $TMDAT
1552
1553 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1554 011264 012737 177777 001562 MOV #-1, $TMDAT
1555
1556 ;* MOV $TMDAT, @GRDAI ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1557 011302 012737 000000 001562 MOV #0, $TMDAT ;CLEAR STATUS
1558
1559 ;* MOV $TMDAT, @GRSTAT ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1560
1561 ;* MOV BRLEV3, @GRDIO ;/ PUT DATA FROM BRLEV3 TO DEVICE REG GRDIO
1562
1563 ;* MOV $TMDAT, @GRSTAT ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRSTAT
1564 ;SHOULD REMAIN SET VIA DIRECT SET SIDE
1565
1566 ;* MOV @GRSTAT, $TMDAT ;/READ DEVICE REG GRSTAT, PUT DATA IN $TMDAT.
1567 011350 005737 001562 TST $TMDAT
1568 011354 100401 BMI 2$ ;;BR IF SET
1569 011356 104010 ERROR 10 ;INPUT INTERRUPT BIT FAILED TO SET INPUT READY
1570 ;?? DID OPERATOR GIVE CORRECT
1571 ;INPUT INTERRUPT BITS ??
1572
1573 011360 2$:
1574
1575 ;* MOV @GRDIO, $TMDAT ;/READ DEVICE REG GRDIO, PUT DATA IN $TMDAT.
1576 011370 043737 001560 001562 BIC BRLEV3, $TMDAT
1577
1578 ;* MOV $TMDAT, @GRDIO ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDIO
1579
1580 ;* MOV @GRDAI, $TMDAT ;/READ DEVICE REG GRDAI, PUT DATA IN $TMDAT.
1581 011416 053737 001560 001562 BIS BRLEV3, $TMDAT
1582
1583 ;* MOV $TMDAT, @GRDAI ;/ PUT DATA FROM $TMDAT TO DEVICE REG GRDAI
1584 011434 005037 001124 CLR $GDDAT ;CLEAR EXPECTED
1585 011440 012737 000000 001562 MOV #0, $TMDAT ;CLEAR STATUS
1586

```



```

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 ;* MOV $TMDAT,$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 †ST35: 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 ;:*****

```

```

1641 011712 000004          TST36: SCOPE
1642 011714 012737 000001 001166 MOV      #1,STIMES      ;;DO 1 ITERATION
1643 011722 004737 020120 JSR      PC,SRESET
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      PSM
1649 011756 000137 012250 JMP      SEOP
1650
1651
1652
1653
1654
1655 011762 000004          ;*****
1656 011764 012706 001100 ;*TEST 37      MISC. EXTERNAL LOGIC TEST
1657 011770 005037 001464 ;*****
1658 011774 005037 001124 TST37: SCOPE
1659
1660
1661
1662
1663
1664
1665 012030 012737 000001 001562 EXTTST: MOV      #STACK,SP      ;SET UP STACK
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694

```



```

1695
1696
1697 012234 005337 012246 ;* MOV EXTTMP, @GRDAI ;/ PUT DATA FROM EXTTMP TO DEVICE REG GRDAI
1698 012240 001321 DEC EXT CNT ;FINISHED COUNT
1699 012242 000207 BNE IS
1700 RTS PC ;EXIT
1701 012244 000000 EXTTMP: 0
1702 012246 000000 EXTCNT: 0
1703
1704 .SBTTL END OF PASS ROUTINE
1705
1706 ;*****
1707 ;INCREMENT THE PASS NUMBER ($PASS)
1708 ;INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
1709 ;TYPE "END PASS"
1710 ;IF THERES A MONITOR GO TO IT
1711 ;IF THERE ISN'T JUMP TO FULNUL
1712 ;IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
1713 ;SENDMG CAN BE CHANGED TO 7.
1714
1715 012250 SEOP:
1716 012250 000004 SCOPE
1717 012252 005037 001102 CLR STSTNM ;: ZERO THE TEST NUMBER
1718 012256 005037 001166 CLR $TIMES ;: ZERO THE NUMBER OF ITERATIONS
1719 012262 005237 001204 INC $PASS ;: INCREMENT THE PASS NUMBER
1720 012266 042737 100000 001204 BIC #100000, $PASS ;: DON'T ALLOW A NEG. NUMBER
1721 012274 005327 DEC (PC)+ ;: LOOP?
1722 012276 000001 SF.OPCT: .WORD 1
1723 012300 003015 BGT SDOAGN ;: YES
1724 012302 012737 MOV (PC)+, @ (PC)+ ;: RESTORE COUNTER
1725 012304 000001 SENDCT: .WORD 1
1726 012306 012276 SEOPCT
1727 012310 104401 012343 TYPE
1728 012314 013700 000042 SGET42: MOV @#42, RO ;: TYPE "END PASS"
1729 012320 001405 BEQ SDOAGN ;: GET MONITOR ADDRESS
1730 012322 000005 RESET ;: BRANCH IF NO MONITOR
1731 012324 004710 SENDAD: JSR PC, (RO) ;: CLEAR THE WORLD
1732 012326 000240 NOP ;: GO TO MONITOR
1733 012330 000240 NOP ;: SAVE ROOM
1734 012332 000240 NOP ;: FOR
1735 012334 SDOAGN: ;: ACT11
1736 012334 000137 JMP @ (PC)+ ;: RETURN
1737 012336 012356 SRTNAD: .WORD FULNUL
1738 012340 377 377 000 SENULL: .BYTE -1, -1, 0 ;: NULL CHARACTER STRING
1739 012343 015 042412 042116 SENDMG: .ASCIZ <15><12>/END PASS/
1740 012350 050040 051501 000123
1741 012356 005037 012374 FULNUL: CLR 10$
1742 012362 005337 012374 IS: DEC 10$
1743 012366 001375 BNE IS
1744 012370 000137 002662 JMP IOTEST
1745 012374 000000 10$: 0
1746
1747 012376 052123 052101 051525 EM1: .ASCIZ /STATUS REGISTER IN ERROR/
1748 012404 051040 043505 051511

```

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
 DRLPJ.P11 END OF PASS ROUTINE

|      |        |        |        |        |       |   |
|------|--------|--------|--------|--------|-------|---|
| 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 |       |   |



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 END OF PASS ROUTINE

```

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 001116 001126 001124 DT1: .EVEN $ERRPC,$BDDAT,$GDDAT,0
1823 013210 000000
1824 013212 001116 000000 DT4: $ERRPC,0
1825 013216 001116 001126 001124 DT10: $ERRPC,$BDDAT,$GDDAT,BRLEV3,0
1826 013224 001560 000000
1827 013230 000 000 000 DF1: .BYTE 0,0,0,0
1828 013233 000
1829 013234 000 000 000 DF10: .BYTE 0,0,0,0,0,0
1830 013237 000 000 000
1831 013242 005015 053523 020122 MSGSWR: .ASCIZ <15><12>/SWR = /
1832 013250 020075 000
1833 013254 000
1834 .EVEN
1835 .SBTTL GETSWR SUBROUTINE
1836
1837 013254 022737 000176 001140 GETSWR: CMP #SWREG,SWR ;TEST IF REAL SWR
1838 013262 001415 BEQ 1$ ;BR IF NOT
1839 013264 104401 013272 TYPE 65$ ;;TYPE ASCIZ STRING
1840 013270 000410 BR 64$ ;;GET OVER THE ASCIZ
1841 ;;65$: .ASCIZ <15><12>/DEPRESS CONT./
1842 64$:
1843 HALT ;WAIT FOR OPERATOR
1844 RTS PC ;EXIT
1845 1$: TYPE
1846 MSGSWR
1847 RDOCT
1848 MOV (SP)+,2SWR ;SAVE VALUE TYPED
1849 RTS PC ;EXIT
1850
1851 .SBTTL SCOPE HANDLER ROUTINE
1852
1853 ;;*****
1854 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1855 ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1856 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>

```

```

1857      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1858      ;*SW14=1      LOOP ON TEST
1859      ;*SW11=1      INHIBIT ITERATIONS
1860      ;*SW09=1      LOOP ON ERROR
1861      ;*SW08=1      LOOP ON TEST IN SWR<7:0>
1862      ;*CALL
1863      ;*          SCOPE          ;;SCOPE=IOT
1864
1865      $SCOPE:
1866      CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
1867      CKSWR
1868      BIT          #BIT14,2SWR      ;;LOOP ON PRESENT TEST?
1869      BNE          $OVER          ;;YES IF SW14=1
1870      ;*****START OF CODE FOR THE XOR TESTER*****
1871      $XTSTR: BR          6$
1872
1873      MOV          2$ERRVEC, -(SP)      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1874      MOV          2$5,2$ERRVEC      ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
1875      TST          2$177060          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1876      MOV          (SP)+,2$ERRVEC      ;;SET FOR TIMEOUT
1877      BR          $SVLAD          ;;TIME OUT ON XOR?
1878      5$: CMP          (SP)+,(SP)+      ;;RESTORE THE ERROR VECTOR
1879      MOV          (SP)+,2$ERRVEC      ;;GO TO THE NEXT TEST
1880      BR          7$          ;;CLEAR THE STACK AFTER A TIME OUT
1881      6$:;*****END OF CODE FOR THE XOR TESTER*****
1882      BIT          #BIT08,2SWR      ;;RESTORE THE ERROR VECTOR
1883      BEQ          2$          ;;LOOP ON SPEC. TEST?
1884      CMPB         2$SWR,$STSINM      ;;ON THE RIGHT TEST? SWR<7:0>
1885      BEQ          $OVER          ;;BR IF YES
1886      2$: TSTB         $ERFLG          ;;HAS AN ERROR OCCURRED?
1887      BEQ          3$          ;;BR IF NO
1888      CMPB         $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
1889      BHI          3$          ;;BR IF NO
1890      BIT          #BIT09,2SWR      ;;LOOP ON ERROR?
1891      BEQ          4$          ;;BR IF NO
1892      7$: MOV          $LPERR,$LPADR      ;;SET LOOP ADDRESS TO 1ST SCOPE
1893      BR          $OVER
1894      4$: CLRB         $ERFLG          ;;ZERO THE ERROR FLAG
1895      CLR          $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1896      BR          1$          ;;ESCAPE TO THE NEXT TEST
1897      3$: BIT          #BIT11,2SWR      ;;INHIBIT ITERATIONS?
1898      BNE          1$          ;;BR IF YES
1899      TST          $PASS          ;;IF FIRST PASS OF PROGRAM
1900      BEQ          1$          ;;INHIBIT ITERATIONS
1901      INC          $ICNT          ;;INCREMENT ITERATION COUNT
1902      CMP          $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
1903      BGE          $OVER          ;;BR IF MORE ITERATION REQUIRED
1904      1$: MOV          #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
1905      MOV          $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
1906      $SVLAD: INCB         $TSTNM      ;;COUNT TEST NUMBERS
1907      MOVB        $TSTNM,$TSTNM      ;;SET TEST NUMBER IN APT MAILBOX
1908      MOV          (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
1909      MOV          (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
1910      CLR          $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS

```



```

1911 013570 112737 000001 001115      MOV#B  #1,SERMAX      ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1912 013576 013777 001102 165336      SOVER:  MOV          $STNM,@DISPLAY  ;; DISPLAY TEST NUMBER
1913 013604 013716 001106      MOV          $LPADR,(SP)  ;; FUDGE RETURN ADDRESS
1914 013610 000002      RTI          ;; FIXES PS
1915 013612 001750      SMXCNT: 1000.          ;; MAX. NUMBER OF ITERATIONS
1916      .SBTTL  ERROR HANDLER ROUTINE
1917
1918      ;; *****
1919      ;; *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1920      ;; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1921      ;; *AND GO TO SERRTYP ON ERROR
1922      ;; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1923      ;; *SW15=1      HALT ON ERROR
1924      ;; *SW13=1      INHIBIT ERROR TYPEOUTS
1925      ;; *SW09=1      LOOP ON ERROR
1926      ;; *CALL
1927      ;; *      ERROR      N      ;; ;ERROR=EMT AND N=ERROR ITEM NUMBER
1928
1929      SERROR:
1930 013614 104406      CKSWR
1931 013616 105237 001103      7$:      INCB          SERFLG      ;; TEST FOR CHANGE IN SOFT-SWR
1932 013622 001775      BEQ          7$          ;; SET THE ERROR FLAG
1933 013624 013777 001102 165310      MOV          $STNM,@DISPLAY  ;; DON'T LET THE FLAG GO TO ZERO
1934 013632 005237 001112      INC          $ERTTL        ;; DISPLAY TEST NUMBER AND ERROR FLAG
1935 013636 011637 001116      MOV          (SP),SERRPC    ;; INC THE ERROR COUNT
1936 013642 162737 000002 001116      SUB          #2,SERRPC     ;; GET ADDRESS OF ERROR INSTRUCTION
1937 013650 117737 165242 001114      MOV#B      @SERRPC,SITEMB  ;; STRIP AND SAVE THE ERROR ITEM CODE
1938 013656 032777 020000 165254      BIT          #BIT13,@SWR   ;; SKIP TYPEOUT IF SET
1939 013664 001004      BNE          20$          ;; SKIP TYPEOUTS
1940 013666 004737 014656      JSR          PC,SERRTYP    ;; GO TO USER ERROR ROUTINE
1941 013672 104401 001173      TYPE          ,SCLF
1942 013676
1943 013676 122737 000001 001216      20$:     CMPB         #APTENV,SENV   ;; RUNNING IN APT MODE
1944 013704 001007      BNE          2$          ;; NO,SKIP APT ERROR REPORT
1945 013706 113737 001114 013720      MOV#B      $ITEMB,21$    ;; SET ITEM NUMBER AS ERROR NUMBER
1946 013714 004737 015760      JSR          PC,SATY4     ;; REPORT FATAL ERROR TO APT
1947 013720      .BYTE      0
1948 013721      .BYTE      0
1949 013722 000777      BR          22$          ;; APT ERROR LOOP
1950 013724 005777 165210      22$:     TST          @SWR        ;; HALT ON ERROR
1951 013730 100002      BPL          3$          ;; SKIP IF CONTINUE
1952 013732 000000      HALT
1953 013734 104406      CKSWR          ;; HALT ON ERROR!
1954 013736 032777 001000 165174      3$:      BIT          #BIT09,@SWR  ;; TEST FOR CHANGE IN SOFT-SWR
1955 013744 001402      BEQ          4$          ;; LOOP ON ERROR SWITCH SET?
1956 013746 013716 001110      MOV          $LPERR,(SP)   ;; BR IF NO
1957 013752 005737 001170      4$:      TST          $ESCAPE     ;; FUDGE RETURN FOR LOOPING
1958 013756 001402      BEQ          5$          ;; CHECK FOR AN ESCAPE ADDRESS
1959 013760 013716 001170      MOV          $ESCAPE,(SP)  ;; BR IF NONE
1960 013764
1961 013764 022737 012324 000042      5$:      CMP          #SENDAD,@#42  ;; FUDGE RETURN ADDRESS FOR ESCAPE
1962 013772 001001      BNE          6$          ;; ACT-11 AUTO-ACCEPT?
1963 013774 000000      HALT          ;; BRANCH IF NO
1964 013776      6$:

```

1965 013776 000002  
 1966  
 1967  
 1968  
 1969  
 1970  
 1971  
 1972  
 1973  
 1974  
 1975  
 1976 014000 011646  
 1977 014002 016666 000004 000002  
 1978 014010 010046  
 1979 014012 010146  
 1980 014014 010246  
 1981 014016 104410  
 1982 014020 012600  
 1983 014022 005001  
 1984 014024 005002  
 1985 014026 112046  
 1986 014030 001412  
 1987 014032 006301  
 1988 014034 006102  
 1989 014036 006301  
 1990 014040 006102  
 1991 014042 006301  
 1992 014044 006102  
 1993 014046 042716 177770  
 1994 014052 062601  
 1995 014054 000764  
 1996 014056 005726  
 1997 014058 010166 000012  
 1998 014054 010237 014100  
 1999 014070 012602  
 2000 014072 012601  
 2001 014074 012600  
 2002 014076 000002  
 2003 014100 000000

```

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 $HIOCT
SRDOCT: MOV      (SP), -(SP)                ;; PROVIDE SPACE FOR THE
MOV      4(SP), 2(SP)                       ;; INPUT NUMBER
MOV      R0, -(SP)                           ;; PUSH R0 ON STACK
MOV      R1, -(SP)                           ;; PUSH R1 ON STACK
MOV      R2, -(SP)                           ;; PUSH R2 ON STACK
1$:      RDLIN                               ;; READ AN ASCII LINE
MOV      (SP)+, R0                           ;; GET ADDRESS OF 1ST CHARACTER
CLR      R1                                  ;; CLEAR DATA WORD
CLR      R2
2$:      MOVB   (R0)+, -(SP)                ;; PICKUP THIS CHARACTER
BEQ      3$                                  ;; IF ZERO GET OUT
ASL      R1                                  ;; *2
ROL      R2                                  ;; *4
ASL      R1                                  ;; *4
ROL      R2                                  ;; *8
ASL      R1                                  ;; *8
ROL      R2
BIC      #1C7, (SP)                          ;; STRIP THE ASCII JUNK
ADD      (SP)+, R1                          ;; ADD IN THIS DIGIT
BR       2$                                  ;; LOOP
3$:      TST   (SP)+                          ;; CLEAN TERMINATOR FROM STACK
MOV      R1, 12(SP)                          ;; SAVE THE RESULT
MOV      R2, $HIOCT
MOV      (SP)+, R2                          ;; POP STACK INTO R2
MOV      (SP)+, R1                          ;; POP STACK INTO R1
MOV      (SP)+, R0                          ;; POP STACK INTO R0
RTI                                           ;; RETURN
$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 TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
;WHEN OPERATING IN TTY FLAG MODE.
$CKSWR: CMP      #SWREG, SWR                 ;; IS THE SOFT-SWR SELECTED?
BNE      1$$                                       ;; BRANCH IF NO
TSTB    @STKS                                       ;; CHAR THERE?
BPL      1$$                                       ;; IF NO, DON'T WAIT AROUND
MOVB    @STKB, -(SP)                               ;; SAVE THE CHAR

```

2014 014102 022737 000176 001140  
 2015 014110 001074  
 2016 014112 105777 165026  
 2017 014116 100071  
 2018 014120 117746 165022



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TTY INPUT ROUTINE

|      |        |        |        |        |               |              |                                     |
|------|--------|--------|--------|--------|---------------|--------------|-------------------------------------|
| 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           | \$WREG,-(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     | \$STKS       | :: CHAR THERE?                      |
| 2033 | 014200 | 100375 |        |        | BPL           | 7\$          | :: IF NOT TRY AGAIN                 |
| 2034 |        |        |        |        |               |              |                                     |
| 2035 | 014202 | 117746 | 164740 |        | MOVB          | \$STKB,-(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, \$STKS | :: 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      205          ;;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          SRDCHR: 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$:      TSTB     @STKS          ;; WAIT FOR
2088 014400 100375          BPL      1$          ;; A CHARACTER
2089 014402 117766 164540 000004  MOVB     @STKB, 4(SP)          ;; READ THE TTY
2090 014410 042766 177600 000004  BIC      #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$:      TSTB     @STKS          ;; WAIT FOR A CHARACTER
2094 014432 100375          BPL      2$          ;; LOOP UNTIL ITS THERE
2095 014434 117746 164506          MOVB     @STKB, -(SP)          ;; GET CHARACTER
2096 014440 042716 177600          BIC      #177, (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$:      RTI          ;; GO BACK TO USER
2106          ;*****
2107          ;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2108          ;CALL:
2109          ;*      RDLIN          ;; INPUT A STRING FROM THE TTY
2110          ;*      RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2111          ;*                  ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
2112          ;
2113 014504 010346          SRDLIN: MOV      R3, -(SP)          ;; SAVE R3
2114 014506 012703 014612 1$:      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 CLR B -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

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

;:*****
;:THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;:ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;:AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

```

```

2151 014656 $ERRTYP: TYPE $CRLF ;: "CARRIAGE RETURN" & "LINE FEED"
2152 014656 104401 001173 MOV RO,-(SP) ;: SAVE RO
2153 014662 010046 CLR RO ;: PICKUP THE ITEM INDEX
2154 014664 005000 BIS B 2*$ITEMB,RO
2155 014666 153700 001114 BNE 1$ ;: IF ITEM NUMBER IS ZERO, JUST
2156 014672 001004 ;: TYPE THE PC OF THE ERROR
2157 ;: SAVE $ERRPC FOR TYPEOUT
2158 014674 013746 001116 MOV $ERRPC,-(SP) ;: ERROR ADDRESS
2159 ;: GO TYPE--OCTAL ASCII(ALL DIGITS)
2160 014700 104402 TYPOC BR 6$ ;: GET OUT
2161 014702 000426 1$: DEC RO ;: ADJUST THE INDEX SO THAT IT WILL
2162 014704 005300 ASL RO ;: WORK FOR THE ERROR TABLE
2163 014706 006300 ASL RO
2164 014710 006300 ASL RO
2165 014712 006300 ADD $ERRTB,RO ;: FORM TABLE POINTER
2166 014714 062700 001322 MOV (RO)+,2$ ;: PICKUP "ERROR MESSAGE" POINTER
2167 014720 012037 014730 BEQ 3$ ;: SKIP TYPEOUT IF NO POINTER
2168 014724 001404 TYPE ;: TYPE THE "ERROR MESSAGE"
2169 014726 104401 ;: "ERROR MESSAGE" POINTER GOES HERE
2170 014730 000000 2$: .WORD 0 ;: "CARRIAGE RETURN" & "LINE FEED"
2171 014732 104401 001173 TYPE $CRLF ;: PICKUP "DATA HEADER" POINTER
2172 014736 012037 014746 3$: MOV (RO)+,4$ ;: SKIP TYPEOUT IF 0
2173 014742 001404 BEQ 5$ ;: TYPE THE "DATA HEADER"
2174 014744 104401 TYPE ;: "DATA HEADER" POINTER GOES HERE
2175 014746 000000 4$: .WORD 0 ;: "CARRIAGE RETURN" & "LINE FEED"
2176 014750 104401 001173 TYPE $CRLF ;: PICKUP "DATA TABLE" POINTER
2177 014754 011000 5$: MOV (RO),RO ;: GO TYPE THE DATA
2178 014756 001004 BNE 7$ ;: RESTORE RO
2179 014760 012600 6$: MOV (SP)+,RO ;: "CARRIAGE RETURN" & "LINE FEED"
2180 014762 104401 001173 TYPE , $CRLF

```

```

2181 014766 000207          RTS      PC          ;;RETURN
2182 014770          7$:      MOV      2(RO)+,-(SP)  ;;SAVE 2(RO)+ FOR TYPEOUT
2183 014770 013046          TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2184 014772 104402          TST     (RO)    ;;IS THERE ANOTHER NUMBER?
2185 014774 005710          BEQ     6$      ;;BR IF NO
2186 014776 001770          TYPE   8$      ;;TYPE TWO(2) SPACES
2187 015000 104401 015006   BR      7$      ;;LOOP
2188 015004 000771          .ASCIZ / /     ;;TWO(2) SPACES
2189 015006 020040 000      .EVEN
2190 015012 015012

```

.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

```

```

2217 015012 017646 000000          $TYPOS: MOV      2(SP),-(SP)  ;;PICKUP THE MODE
2218 015016 116637 000001 015235  MOVB    1(SP),SOFILL  ;;LOAD ZERO FILL SWITCH
2219 015024 112637 015237          MOVB    (SP)+,SOMODE+1  ;;NUMBER OF DIGITS TO TYPE
2220 015030 062716 000002          ADD     #2,(SP)      ;;ADJUST RETURN ADDRESS
2221 015034 000406          BR      $TYPON
2222 015036 112737 000001 015235  $TYPOC: MOVB    #1,SOFILL  ;;SET THE ZERO FILL SWITCH
2223 015044 112737 000006 015237  MOVB    #6,SOMODE+1  ;;SET FOR SIX(6) DIGITS
2224 015052 112737 000005 015234  $TYPON: MOVB    #5,SOCNT  ;;SET THE ITERATION COUNT
2225 015060 010346          MOV     R3,-(SP)    ;;SAVE R3
2226 015062 010446          MOV     R4,-(SP)    ;;SAVE R4
2227 015064 010546          MOV     R5,-(SP)    ;;SAVE R5
2228 015066 113704 015237  MOVB    SOMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
2229 015072 005404          NEG     R4
2230 015074 062704 000006          ADD     #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
2231 015100 110437 015236          MOVB    R4,SOMODE  ;;SAVE IT FOR USE
2232 015104 113704 015235          MOVB    SOFILL,R4  ;;GET THE ZERO FILL SWITCH
2233 015110 016605 000012          MOV     12(SP),R5  ;;PICKUP THE INPUT NUMBER
2234 015114 005003          CLR     R3         ;;CLEAR THE OUTPUT WORD

```



```

2235 015116 006105 1$: ROL R5 ;: ROTATE MSB INTO "C"
2236 015120 000404 BR R3 ;: GO DO MSB
2237 015122 006105 2$: 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 ONE 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 #SILLUP,@#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 $SWR,-(SP) ;: PUSH $SWR ON STACK
2283 015274 010637 015410 MOV SP,$SAVR6 ;: SAVE SP
2284 015300 012737 015312 000024 MOV #SPWRUP,@#PWRVEC ;: SET UP VECTOR
2285 015306 000000 HALT
2286 015310 000776 BR .-2 ;: HANG UP
2287
2288 ;: *****

```

```

2289          :POWER UP ROUTINE
2290 015312 012737 015404 000024 $PWRUP: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST DOWN
2291 015320 013706 015410          MOV $SAVR6, SP ;; GET SP
2292 015324 005037 015410          CLR $SAVR6 ;; WAIT LOOP FOR THE TTY
2293 015330 005237 015410 1$: INC $SAVR6 ;; WAIT FOR THE INC
2294 015334 001375          BNE 1$ ;; OF WORD
2295 015336 012677 163576          MOV (SP)+, @SWR ;; POP STACK INTO @SWR
2296 015342 012605          MOV (SP)+, R5 ;; POP STACK INTO R5
2297 015344 012604          MOV (SP)+, R4 ;; POP STACK INTO R4
2298 015346 012603          MOV (SP)+, R3 ;; POP STACK INTO R3
2299 015350 012602          MOV (SP)+, R2 ;; POP STACK INTO R2
2300 015352 012601          MOV (SP)+, R1 ;; POP STACK INTO R1
2301 015354 012600          MOV (SP)+, R0 ;; POP STACK INTO R0
2302 015356 012737 015240 000024 MOV $SPWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR
2303 015364 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; PRIO:7
2304 015372 104401          TYPE PWRMSG ;; REPORT THE POWER FAILURE
2305 015374 015412 $PWRMG: .WORD PWRMSG ;; POWER FAIL MESSAGE POINTER
2306 015376 012716          MOV (PC)+, (SP) ;; RESTART AT IOTEST
2307 015400 002662 $PWRAD: .WORD IOTEST ;; RESTART ADDRESS
2308 015402 000002          RTI
2309 015404 000000 $SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED
2310 015406 000776          BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
2311 015410 000000 $SAVR6: 0 ;; PUT THE SP HERE
2312 015412 005015 042522 052123 PWRMSG: .ASCIZ <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
2313 015420 051101 044524 043516
2314 015426 040440 052106 051105
2315 015434 040440 050040 053517
2316 015442 051105 043040 044501
2317 015450 052514 042522 005015
2318 015456          000
2319          015460          .EVEN
2320
2321          .SBTTL TYPE ROUTINE
2322
2323          ;*****
2324          ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2325          ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2326          ;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2327          ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2328          ;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2329          ;
2330          ;CALL:
2331          ;*1) USING A TRAP INSTRUCTION
2332          ;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2333          ;*OR
2334          ;* TYPE
2335          ;* MESADR
2336          ;*
2337
2338 015460 105737 001157 $TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
2339 015464 100002          BPL 1$ ;; BR IF YES
2340 015466 000000          HALT ;; HALT HERE IF NO TERMINAL
2341 015470 000430          BR 3$ ;; LEAVE
2342 015472 010046 1$: MOV R0, -(SP) ;; SAVE R0

```



|      |        |        |        |        |               |                |   |
|------|--------|--------|--------|--------|---------------|----------------|---|
| 2343 | 015474 | 017600 | 000002 |        | MOV           | 2(SP),RO       | :: GET ADDRESS OF ASCIZ STRING            |
| 2344 | 015500 | 122737 | 000001 | 001216 | CMPB          | #APTENV,SENV   | :: RUNNING IN APT MODE                    |
| 2345 | 015506 | 001011 |        |        | BNE           | 62\$           | :: NO,GO CHECK FOR APT CONSOLE            |
| 2346 | 015510 | 132737 | 000100 | 001217 | BITB          | #APTPOOL,SENVm | :: SPOOL MESSAGE TO APT                   |
| 2347 | 015516 | 001405 |        |        | BEQ           | 62\$           | :: NO,GO CHECK FOR CONSOLE                |
| 2348 | 015520 | 010037 | 015530 |        | MOV           | RO,61\$        | :: SETUP MESSAGE ADDRESS FOR APT          |
| 2349 | 015524 | 004737 | 015750 |        | JSR           | PC,SATY3       | :: SPOOL MESSAGE TO APT                   |
| 2350 | 015530 | 000000 |        |        |               | .WORD          | :: MESSAGE ADDRESS                        |
| 2351 | 015532 | 132737 | 000040 | 001217 | 61\$: BITB    | #APTCSUP,SENVm | :: APT CONSOLE SUPPRESSED                 |
| 2352 | 015540 | 001003 |        |        | 62\$: BNE     | 60\$           | :: YES,SKIP TYPE OUT                      |
| 2353 | 015542 | 112046 |        |        | 2\$: MOVB     | (RO)+,-(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)+,RO       | :: RESTORE RO                             |
| 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 |        |        | SCRLF         |                |   |
| 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 |        |        |        |        |               |                |   |
| 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 | \$STPS         | :: WAIT UNTIL PRINTER IS READY            |
| 2388 | 015676 | 100375 |        |        | BPL           | \$TYPEC        |   |
| 2389 | 015700 | 116677 | 000002 | 163244 | MOVB          | 2(SP),2\$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 RO,-(SP) ;; PUSH RO 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 #APTPOOL,$ENVm ;; SHOULD SPOOL MESSAGES?
2416 016016 001425 BEQ 3$ ;; IF NOT: BR
2417 016020 017600 000004 MOV #4(SP),RO ;; 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 RO,$MSGAD ;; PUT ADDR IN MAILBOX
2422 016044 105720 2$: TSTB (RO)+ ;; FIND END OF MESSAGE
2423 016046 001376 BNE 2$
2424 016050 163700 001212 SUB $MSGAD,RO ;; SUB START OF MESSAGE
2425 016054 006200 ASR RO ;; GET MESSAGE LGTH IN WORDS
2426 016056 010037 001214 MOV RO,$MSGLG ;; 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)+,RO ;; POP STACK INTO RO
2449 016202 000207 RTS PC ;; RETURN
2450 016204 000 $MFLG: .BYTE 0 ;; MESSG. FLAG

```



2451  
2452  
2453  
2454  
2455  
2456  
2457  
2458  
2459  
2460  
2461  
2462  
2463  
2464  
2465  
2466  
2467  
2468  
2469  
2470  
2471  
2472  
2473  
2474  
2475  
2476  
2477  
2478  
2479  
2480  
2481  
2482  
2483  
2484  
2485  
2486  
2487  
2488  
2489  
2490  
2491  
2492  
2493  
2494  
2495  
2496  
2497  
2498  
2499  
2500  
2501  
2502  
2503  
2504

016205 000  
016206 000  
016210 000200  
000001  
000100  
000040  
  
010046 000002  
016600  
005740  
111000  
006300  
016000 016244  
000200  
  
011646 000004 000002  
016666  
000002  
  
016232 016232  
016246 015460  
016250 015036  
016252 015012  
016254 015052  
  
016256 014152  
  
016260 014102  
016262 014364  
016264 014504  
016266 014000

\$LFLG: .BYTE 0  
\$FFLG: .BYTE 0  
          .EVEN  
APTSIZE=200  
APTENV=001  
APTSPOOL=100  
APTCSUP=040

:: LOG FLAG  
:: FATAL FLAG

.SBTTL TRAP DECODER

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

\$TRAP: MOV RO, -(SP)           :: SAVE RO  
      MOV 2(SP), RO           :: GET TRAP ADDRESS  
      TST -(RO)               :: BACKUP BY 2  
      MOVB (RO), RO           :: GET RIGHT BYTE OF TRAP  
      ASL RO                  :: POSITION FOR INDEXING  
      MOV \$TRAPD(RO), RO      :: INDEX TO TABLE  
      RTS RO                  :: GO TO ROUTINE

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

\$TRAP2: MOV (SP), -(SP)       :: MOVE THE PC DOWN  
      MOV 4(SP), 2(SP)        :: MOVE THE PSW DOWN  
      RTI                      :: RESTORE THE PSW

.SBTTL TRAP TABLE

:: \*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED  
:: \*BY THE "TRAP" INSTRUCTION.

ROUTINE  
-----  
\$TRAPD: WORD \$TRAP2  
          \$TYPE    :: CALL=TYPE    TRAP+1(104401) TTY TYPEOUT ROUTINE  
          \$TYPOC   :: CALL=TYPOC   TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)  
          \$TYPOS    :: CALL=TYPOS   TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)  
          \$TYPON    :: CALL=TYPON   TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)  
  
          \$GTSWR    :: CALL=GTSWR   TRAP+5(104405) GET SOFT-SWR SETTING  
  
          \$CKSWR    :: CALL=CKSWR   TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR  
          \$RDCHR    :: CALL=RDCHR   TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE  
          \$RDLIN    :: CALL=RDLIN   TRAP+10(104410) TTY TYPEIN STRING ROUTINE  
          \$RDOCT    :: CALL=RDOCT   TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY

:: \*  
:: \*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11  
:: \*FIRST WE WILL LOAD MICROCODE INTO KMC-11

E05

2505  
2506  
2507  
2508  
2509  
2510  
2511  
2512  
2513  
2514  
2515  
2516  
2517  
2518  
2519  
2520  
2521  
2522  
2523  
2524  
2525  
2526  
2527  
2528  
2529  
2530  
2531  
2532  
2533  
2534  
2535  
2536  
2537  
2538  
2539  
2540  
2541  
2542  
2543  
2544  
2545  
2546  
2547  
2548  
2549  
2550  
2551  
2552  
2553  
2554  
2555  
2556  
2557  
2558

```

;*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 SAERR=1 IF SLAVE
;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
;*

```

```

016270 013746 000004      $LPAI:
016270 013746 000004      MOV      4,-(SP)
016274 000413            BR       31$

016276 012737 016322 000004      MOV      #30$ 4
016304 005237 170000            INC      170000
016310 104401 016316            TYPE    65$
016314 000401            BR       64$

        ;:65$:
        64$:      .ASCIZ  <?>##

016320 000401            BR       31$
016322 022626            CMP      (SP)+,(SP)+
016324 012637 000004      30$:    MOV      (SP)+,4
016330 005037 017146            31$:    CLR      $AERR
016334 004537 017150            JSR     R5,$LOAD
016340 000000G           .WORD   DRLPX2

016342 052777 040000 163066      BIS     #BIT14,$KMA0
016350                                1$:

016350 010146            MOV      R1,-(SP)
016352 005001            CLR      R1
016354 005201            2$:    INC      R1
016356 001376            BNE     2$
016360 012777 104000 163050      MOV     #BIT15!BIT11,$KMA0
016366 105201            25$:   INCB   R1
016370 001376            BNE     25$

```

```

;FIELD DOES NOT HAVE A BUS SWITCH TO
;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
;BRANCH AROUND 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.

;;TYPE ASCIZ STRING
;;GET OVER THE ASCIZ

;ALL THIS JUNK MUST BE REMOVED!!

;LOAD MICRO-CODE.
;FILE "DRLPX2.OBJ"

;ISSUE KMC+DMC INIT.

;"HANGS" HERE THEN KMC-11 ERROR.

;STALL FOR DMC-UP

;SET RUN, AND ENABLE ARBITRATION.

```



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TRAP TABLE

```

2559 016372 032777 000040 163036 BIT #BITS, @KMADO ;SLAVE READY? (READING IPBM SR)
2560 016400 001401 BEQ 3$ ;FATAL LPA-11 ERROR SLAVE NOT READY.
2561 016402 104000 ERROR ;
2562 016404 012777 000004 163030 3$: MOV #4, @KMAD2 ;READ FAST PATH
2563 016412 004537 020060 4$: JSR R5, $TOU1 ;-TOUT-CHECK FOR TIMEOUT
2564 016416 104000 ERROR ;/TIME-OUT ERROR
2565 ;/WE FAILED TO COMPLETE
2566 ;/CURRENT OPERATION.
2567 ;/CONTINUES IN THIS LOOP
2568 ;/WOULD MAKE US "HANG" HERE
2569
2570
2571
2572
2573
2574 016420 000774 BR 4$ ;
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$: .ASCIZ <200>"W A R N I N G THIS PROGRAM WAS DESIGNED TO RUN WITH VERSION 4"
2593 ;:66$:
2594 016576 104401 016604 TYPE 69$ ;:TYPE ASCIZ STRING
2595 016602 000430 BR 68$ ;:GET OVER THE ASCIZ
2596 ;:69$: .ASCIZ <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
2597 ;:68$:
2598 016664 104401 016672 TYPE 71$ ;:TYPE ASCIZ STRING
2599 016670 000434 BR 70$ ;:GET OVER THE ASCIZ
2600 ;:71$: .ASCIZ <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
2601 ;:70$:
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.
    
```

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TRAP TABLE

```

2613 017032 004737 017116 JSR PC,20$
2614
2615 017036 032777 000002 162372 7$: BIT #BIT1,@KMACG ;WAIT FOR FIFO DATA
2616 017044 001374 BNE 7$ ;=1 NO DATA. =0 DATA.
2617 017046 112777 000002 162366 MOVB #2,@KMAC2 ;READ FIFO.
2618
2619 017054 8$: JSR RS, $TOUT ; -TOUT-CHECK FOR TIMEOUT
2620 017054 004537 020060
2621
2622 017060 104000 ERROR ;/TIME-OUT ERROR
2623 ;/WE FAILED TO COMPLETE
2624 ;/CURRENT OPERATION.
2625 ;/CONTINUES IN THIS LOOP
2626 ;/WOULD MAKE US "HANG" HERE
2627
2628 017062 000774 BR 8$
2629
2630 ;/RETURNS HERE-FROM-TIMED OUT.
2631 017064 122777 000377 162350 CMPB #377,@KMAC2 ;WAIT FOR READ.
2632 017072 001370 BNE 8$
2633 017074 105777 162346 TSTB @KMAC4 ; 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 ; 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,@KMAC2 ; ISSUE FIFO WRITE
2646 017124 21$: JSR RS, $TOUT ; -TOUT-CHECK FOR TIMEOUT
2647 017124 004537 020060
2648
2649 017130 104000 ERROR ;/TIME-OUT ERROR
2650 ;/WE FAILED TO COMPLETE
2651 ;/CURRENT OPERATION.
2652 ;/CONTINUES IN THIS LOOP
2653 ;/WOULD MAKE US "HANG" HERE
2654
2655 017132 000774 BR 21$
2656
2657 ;/RETURNS HERE-FROM-TIMED OUT.
2658 017134 122777 000377 162300 CMPB #377,@KMAC2 ;KMC CODE WILL RETURN A "377"
2659 017142 001370 BNE 21$ ;WHEN DONE COMMAND.
2660 017144 000207 RTS PC
2661
2662 017146 000000 $AERR: .WORD 0 ;=0 IF ADDR. LIST OK,=1 IF BAD.
2663
2664 ;*
2665 ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
2666 ;* CALL = JSR RS,$LOAD

```



```

2667      ;*
2668      ;*
2669      ;*
2670      ;*
2671      ;*
2672      017150 010446      $LOAD:  MOV    R4,-(SP)      ;SAVE R4.
2673      017152 010046      MOV    R0,-(SP)      ;SAVE R0.
2674      017154 012500      1$:   MOV    (5)+,R0      ;GET PROG. ADDR.
2675      017156 005077 162254  CLR    @KMADO        ;CLEAR CSR
2676      017162 005077 162260  CLR    @KMAD4        ;CLEAR CRAM ADDR.
2677      017166 052777 002000 162242 2$:   BIS    #2000,@KMADO  ;SELECT CRAM.
2678      017174 012077 162252  MOV    (0)+,@KMAD6   ;WRITE DATA.
2679      017200 052777 020000 162230  BIS    #20000,@KMADO ;SET CRAM WRITE
2680      017206 005077 162224  CLR    @KMADO        ;DISABLE CRAM.
2681      017212 005277 162230  INC    @KMAD4        ;UPDATE CRAM ADDR.
2682      017216 021027 177777  CMP    (0),#-1      ;ALL DONE?
2683      017222 001361 162216  BNE    2$           ;NO LOOP
2684      017224 005077 162216  CLR    @KMAD4        ;CLEAR CRAM ADDR.
2685      017230 016500 177776  MOV    -2(5),R0     ;GET MICRO CODE ADDR.
2686
2687      017234 052777 002000 162174 3$:   BIS    #2000,@KMADO  ;SELECT CRAM
2688      017242 022077 162204  CMP    (R0)+,@KMAD6  ;DATA OK?
2689      017246 001013 177777  BNE    5$           ;NO - REPORT AN ERROR.
2690      017250 021027 177777  CMP    (0),#-1      ;ALL DONE?
2691      017254 001405 162154  BEQ    4$           ;YES - EXIT
2692      017256 005077 162160  CLR    @KMADO        ;NO - DESELECT CRAM.
2693      017262 005277 162160  INC    @KMAD4        ;UPDATE CRAM ADDR.
2694      017266 000762 162160  BR     3$
2695
2696      017270 012600      4$:   MOV    (SP)+,R0     ;RESTORE R0
2697      017272 012604      MOV    (SP)+,R4     ;RESTORE R4
2698      017274 000205      RTS    R5           ;EXIT
2699
2700      017276      5$:   ;COME HERE ON LOAD ERROR
2701      017276 005745      TST    -(5)
2702      017300 105204      INCB   R4           ;UPDATE ERROR COUNTER.
2703      017302 100324      BPL    1$           ;IF NOT TOO MANY, TRY AGAIN.
2704      017304 000000      HALT
2705
2706      017306 000722      BR     1$           ;MICRO CODE LOAD ERROR.
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720      017310 010046      $TLKW: MOV    R0,-(SP) ;SAVE R0

```

```

;WORD XX ;ADDR. OF MICRO CODE.
;RETURNS HERE
NOTE: MICRO CODE FILE MUST END IN -1 DATA.

```

```

;*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
;*

```

```

2721 017312 012500          MOV      (5)+,RO          ;GET DEVICE OFFSET
2722 017314 052700 000340  BIS      #340,RO          ;ADD WRITE CODE.
2723 017320 004737 017572  JSR      PC,$LPW          ;WAIT FOR FAST PATH READY
2724 017324 010037 017416  MOV      RO,W1
2725 017330 010077 162112  MOV      RO,@KMAD4
2726 017334 112777 000005 162100  MOVB    #5,@KMAD2          ;ISSUE FAST PATH WRITE
2727 017342 004737 017572  JSR      PC,$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)+,RO
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
2745
2746
2747
2748
2749
2750
2751
2752 017424 010046          $TLKR: MOV      RO,-(SP)          ;SAVE RO
2753 017426 012500          MOV      (5)+,RO          ;GET OFFSET
2754 017430 052700 000300  BIS      #300,RO          ;ADD READ CODE
2755 017434 004737 017572  JSR      PC,$LPW          ;WAIT TILL READY
2756 017440 110077 162002  MOVB    RO,@KMAD4
2757 017444 112777 000005 161770  MOVB    #5,@KMAD2          ;ISSUE WRITE FP
2758 017452 004737 017572  JSR      PC,$LPW
2759 017456 010037 017566  MOV      RO,RD1
2760 017462  JSR      R5,$TOUT         ;-TOUT-CHECK FOR TIMEOUT
2761 017462 004537 020060  1$:
2762
2763 017466 104000  ERROR
2764
2765
2766
2767
2768
2769 017470 000774          BR      1$
2770
2771
2772 017472 032777 000040 161736  BIT      #BITS,@KMADO
2773 017500 001370  BNE     1$
2774 017502 112777 000004 161732  MOVB    #4,@KMAD2          ;ISSUE FAST PATH READ

```

```

;*
;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
;*
;*      CALL = JSR      R5,$TLKR
;*              .WORD  0          ;OFFSET OF DEVICE
;*              :RETURNS HERE
;*DATE IN WORD $DATR
;*

```



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
 DRLPJ.P11 TRAP TABLE

```

2775 017510 004737 017572      JSR    PC,$LPW
2776 017514 117737 161726 017570 2$:    MOVB  @KMD4,$DATR ;GET LOW BYTE
2777 017522                                JSR    R5,$TOUT  ; -TOUT-CHECK FOR TIMEOUT
2778 017522 004537 020060                                ERROR      ; /TIME-OUT ERROR
2779 017526 104000                                ; /WE FAILED TO COMPLETE
2780                                ; /CURRENT OPERATION.
2781                                ; /CONTINUES IN THIS LOOP
2782                                ; /WOULD MAKE US "HANG" HERE
2783
2784
2785
2786 017530 000774      BR          2$
2787
2788                                ; /RETURNS HERE-FROM-TIMED OUT.
2789 017532 032777 000040 161676  BIT    #BIT5,@KMD0 ;FAST PATH READY?
2790 017540 001370                                BNE    2$
2791 017542 112777 000004 161672  MOVB  #4,@KMD2    ;ISSUE FAST PATH READ
2792 017550 004737 017572      JSR    PC,$LPW
2793 017554 117737 161666 017571  MOVB  @KMD4,$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,@KMD2  ;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$:    BIT    #BIT4,@KMD0 ;FAST PATH READ?
2817 017622 001403      BEQ   3$
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$:    MOV    (SP)+,R1    ;RESTORE R1
2823 017634 000207      RTS    PC        ;EXIT
2824
2825 017636
2826 017636 104401 017644      10$:  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
2838
2839
2840
2841
2842
2843
2844
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
2876
2877
2878
2879
2880
2881
2882

```



```

2883          ;*      CALL THROUGH      MOVEI      DATA,ADDR.
2884          ;*      WHICH EQUALS:
2885          ;*      JSR          R5,$INLP
2886          ;*      .WORD      XX          ADDR OF DEVICE
2887          ;*      .WORD      YY          ADDR TO STORE READ DATA.
2888
2889 017764 010046          $INLP: MOV      R0,-(SP)          ;SAVE R0
2890 017766 010146          MOV      R1,-(SP)          ;SAVE R1
2891
2892 017770 012700 001464          MOV      #.DVLS,R0          ;PROG DEFINED ADDR. LIST.
2893 017774 005001          CLR      R1
2894 017776 005710          1$:    TST      (0)          ;EOL REACHED?
2895 020000 001420          BEQ      10$              ;YES - DEFINE NEW ADDR.
2896
2897 020002 027520 000000          CMP      @($),(0)+          ;ADDR. MATCH?
2898 020006 001402          BEQ      2$
2899 020010 005201          INC      R1
2900 020012 000771          BR       1$
2901
2902 020014 010137 020026          2$:    MOV      R1,3$          ;SAVE LIST OFFSET
2903 020020 005725          TST      ($)+
2904 020022 004537 017424          JSR      R5,$TLKf          ;GO READ DEVICE
2905
2906 020026 000000          $OFS=. 3$:    .WORD      0          ;OFFSET OF DEVICE
2907
2908 020030 013735 017570          MOV      $DATR,@($)+          ;STORE DATA.
2909 020034 012601          MOV      (SP)+,R1          ;RESTORE R1
2910 020036 012600          MOV      (SP)+,R0          ;RESTORE R2
2911 020040 000205          RTS
2912
2913 020042 017520 000000          10$:   MOV      @($),(0)+
2914 020046 005010          CLR      (0)
2915 020050 004537 016270          JSR      R5,$LPAI
2916 020054 001464          .WORD      .DVLS
2917 020056 000756          BR       2$
2918
2919          ;*
2920          ;*STOUT ROUTINE USED TO WATCH IF
2921          ;*WE'RE IN A LOOP TOO-LONG
2922          ;*CALL= JSR R5,$STOUT
2923          ;*ERROR X ;RETURNS HERE ON TIMEOUT
2924          ;*BR
2925          ;*RETURNS HERE NO ERROR
2926          ;*
2927 020060 020537 020114          $STOUT: CMP      R5,$$AD          ;SAME ADDR?
2928 020064 001405          BEQ      1$
2929 020066 010537 020114          MOV      R5,$$AD          ;NO-SAVE THIS ADDR.
2930 020072 005037 020116          CLR      $CNT          ;CLR CNT AT ADDR.
2931 020076 000403          BR       2$
2932 020100 005237 020116          1$:    INC      $CNT          ;OVERFLOW?
2933 020104 100402          BMI      3$              ;YES-ERROR RETURN
2934 020106 062705 000004          2$:    ADD      #4,R5          ;NO-NON ERROR RETURN
2935 020112 000205          3$:    RTS      R5          ;RETURN.
2936

```

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TRAP TABLE

```

2937 020114 000000          $$AD: .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          $RESET: RESET          ;RESET THE WORLD.
2950
2951          ;*          MOV          22$,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          ;*          CALL= JSR PC, SDELAY
2976          ;*
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$: INCB          TIME
2990 020224 001375          ;*          BNE          10$

```



MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TRAP TABLE

|      |        |        |        |        |         |             |  |
|------|--------|--------|--------|--------|---------|-------------|--|
| 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 |        |        |        |        |         |             |  |
| 3000 |        |        |        |        |         |             |  |
| 3001 |        |        |        |        |         |             |  |
| 3002 |        |        |        |        |         |             |  |
| 3003 |        |        |        |        |         |             |  |
| 3004 |        |        |        |        |         |             |  |
| 3005 |        |        |        |        |         |             |  |
| 3006 |        |        |        |        |         |             |  |
| 3007 |        |        |        |        |         |             |  |
| 3008 |        |        |        |        |         |             |  |
| 3009 | 020242 |        |        |        | \$UTK:  |             |  |
| 3010 | 020242 | 005037 | 001464 |        | CLR     | .DVLS       |  |
| 3011 | 020246 |        |        |        | 21\$:   |             |  |
| 3012 | 020246 | 104401 | 020254 |        | TYPE    | 65\$        | ::TYPE ASCIZ STRING                    |
| 3013 | 020252 | 000405 |        |        | BR      | 64\$        | ::GET OVER THE ASCIZ                   |
| 3014 |        |        |        |        | ::65\$: | .ASCIZ      | <200>#E OR D?#                         |
| 3015 | 020266 |        |        |        | 64\$:   |             |  |
| 3016 | 020266 | 105777 | 160652 |        | 1\$:    | TSTB        | 2\$TKS                                 |
| 3017 | 020272 | 100375 |        |        | BPL     | 1\$         |  |
| 3018 | 020274 | 117737 | 160646 | 020416 | MOVB    | 2\$TKB,20\$ | ;GET INPUT                             |
| 3019 | 020302 | 104401 | 020416 |        | TYPE,   | 20\$        | ;ECHO NEXT MESSAGE.                    |
| 3020 | 020306 | 142737 | 000240 | 020416 | BICB    | #240,20\$   | ;STRIP PARITY, LC                      |
| 3021 | 020314 | 104411 |        |        | RDOCT   |             | ;GET ADDR.                             |
| 3022 | 020316 | 012637 | 020414 |        | MOV     | (SP)+,14\$  |  |
| 3023 | 020322 | 123727 | 020416 | 000104 | CMPB    | 20\$,#D     | ;DEPOSIT?                              |
| 3024 | 020330 | 001411 |        |        | BEQ     | 10\$        |  |
| 3025 |        |        |        |        |         |             |  |
| 3026 | 020332 | 004537 | 017764 |        | JSR     | R5,\$INLP   | ;GET DATA                              |
| 3027 | 020336 | 020414 |        |        | 2\$:    | .WORD       | 14\$                                   |
| 3028 | 020340 | 020352 |        |        | .WORD   | 5\$         |  |
| 3029 |        |        |        |        |         |             |  |
| 3030 | 020342 | 013746 | 020352 |        | MOV     | 5\$,-(SP)   | ::SAVE 5\$ FOR TYPEOUT                 |
| 3031 | 020346 | 104402 |        |        | TYPOC   |             | ::GO TYPE--OCTAL ASCII(ALL DIGITS)     |
| 3032 | 020350 | 000736 |        |        | BR      | 21\$        | ;LOOP.                                 |
| 3033 | 020352 | 000000 |        |        | 5\$:    | .WORD       | 0                                      |
| 3034 |        |        |        |        |         |             |  |
| 3035 | 020354 |        |        |        | 10\$:   |             |  |
| 3036 | 020354 | 104401 | 020362 |        | TYPE    | 67\$        | ::TYPE ASCIZ STRING                    |
| 3037 | 020360 | 000404 |        |        | BR      | 66\$        | ::GET OVER THE ASCIZ                   |
| 3038 |        |        |        |        | ::67\$: | .ASCIZ      | <200>#DATA= #                          |
| 3039 | 020372 |        |        |        | 66\$:   |             |  |
| 3040 | 020372 | 104411 |        |        | RDOCT   |             |  |
| 3041 | 020374 | 012637 | 020412 |        | MOV     | (SP)+,13\$  |  |
| 3042 |        |        |        |        |         |             |  |
| 3043 | 020400 | 004537 | 017666 |        | 11\$:   | JSR         | R5,\$OUTLP                             |
| 3044 | 020404 | 020414 |        |        | 12\$:   | .WORD       | 14\$ ;OUTPUT ROUTINE.<br>;DEVICE ADDR. |

MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 TRAP TABLE

|      |        |        |        |        |
|------|--------|--------|--------|--------|
| 3045 | 020406 | 020412 |        |        |
| 3046 | 020410 | 000716 |        |        |
| 3047 |        |        |        |        |
| 3048 | 020412 | 000000 |        |        |
| 3049 | 020414 | 000000 |        |        |
| 3050 | 020416 | 100001 | 042504 | 044526 |
| 3051 | 020424 | 042503 | 040440 | 042104 |
| 3052 | 020432 | 036522 | 000040 |        |
| 3053 |        |        |        |        |
| 3054 |        |        |        |        |
| 3055 |        |        |        |        |
| 3056 |        |        |        |        |
| 3057 |        |        |        |        |
| 3058 |        |        |        |        |
| 3059 |        |        |        |        |
| 3060 |        |        |        |        |
| 3061 |        |        |        |        |
| 3062 |        |        |        |        |
| 3063 |        |        |        |        |
| 3064 |        |        |        |        |
| 3065 |        |        |        |        |
| 3066 |        |        |        |        |
| 3067 |        |        |        |        |
| 3068 |        |        |        |        |
| 3069 |        |        |        |        |
| 3070 |        |        |        |        |
| 3071 |        |        |        |        |
| 3072 |        |        |        |        |
| 3073 |        |        |        |        |
| 3074 |        |        |        |        |
| 3075 |        |        |        |        |
| 3076 |        |        |        |        |
| 3077 | 020436 | 012537 | 020446 |        |
| 3078 | 020442 | 004537 | 017764 |        |
| 3079 | 020446 | 000000 |        |        |
| 3080 | 020450 | 020544 |        |        |
| 3081 | 020452 | 113777 | 020026 | 160772 |
| 3082 | 020460 | 113777 | 020026 | 160766 |
| 3083 | 020466 | 013737 | 020446 | 020506 |
| 3084 | 020474 | 062737 | 000002 | 020506 |
| 3085 | 020502 | 004537 | 017764 |        |
| 3086 | 020506 | 000000 |        |        |
| 3087 | 020510 | 020544 |        |        |
| 3088 | 020512 | 113777 | 020026 | 160724 |
| 3089 | 020520 | 152777 | 000340 | 160724 |
| 3090 | 020526 | 152777 | 000300 | 160720 |
| 3091 | 020534 | 152777 | 000300 | 160702 |
| 3092 | 020542 | 000205 |        |        |
| 3093 | 020544 | 000000 |        |        |
| 3094 |        |        |        |        |
| 3095 |        | 000001 |        |        |

```

.WORD 13$ ;DATA
BR 21$
13$: .WORD 0
14$: .WORD 0
20$: .ASCIZ <1><200>#DEVICE ADDR= #

```

.EVEN

```

: THIS ROUTINE LOOKS THROUGH CURENT .DVL$ FOR A/D ADDR.
: IF UNFOUND GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
: TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
: SAMPLE TAKEING PURPOSES.
: TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
: A/D CSR IN BSEL 4, AND 5.
: (2) HE MUST CALL THIS ROUTINE:
: JSR R5,$PUTS ;CALL SET UP ROUTINE.
: .WORD ADCSR ;ADDR. OF A/D CSR.
: RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
: ;(UNTILL ONE DOES A RESET)

: (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
: START CONVERSION CAUTION*DO WITH MOV$ INSTR.!
: (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
: (5)READ KMC REG 4,5 FOR A/D RESULT.
: (6) TO TAKE MORE SAMPLES, SIMPLY PUT A/D CSR INTO
: BSEL 4,5 AND CODE 6 INTO BSEL 2.

```

```

SPUTS: MOV (5)+,1$ ;GET ADDR OF ADDR. OF A/D
JSR R5,$INLP
1$: .WORD 0
.WORD 10$
MOVB $OFS,$KMA$6
MOVB $OFS,$KMA$7
MOV 1$,$2$
ADD #2,$2$
JSR R5,$INLP
2$: .WORD 0
.WORD 10$
MOVB $OFS,$KMA$3
BISB #340,$KMA$6
BISB #300,$KMA$7
BISB #300,$KMA$3
RTS R5
10$: .WORD 0
.END

```



|                 |      |       |       |       |     |
|-----------------|------|-------|-------|-------|-----|
| ABASE = 170410  | 168# | 280   | 321   |       |     |
| ACDW1 = 000000  | 280  | 323   |       |       |     |
| ACDW2 = 000000  | 280  | 324   |       |       |     |
| ACPUOP = 000000 | 280  | 325   |       |       |     |
| ADDW0 = 000000  | 280  | 326   |       |       |     |
| ADDW1 = 000000  | 280  | 327   |       |       |     |
| 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   |       |       |     |
| ADEVN = 000000  | 280  | 322   |       |       |     |
| RENV = 000000   | 280  | 291   |       |       |     |
| RENVM = 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   |       |       |     |
| AMSGLC = 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   |       |       |     |
| AUSMR = 000000  | 280  | 294   |       |       |     |
| AVECT1 = 000350 | 169# | 280   | 319   | 445   | 446 |
| AVECT2 = 000000 | 280  | 320   |       |       |     |
| BEGIN = 001564  | 193  | 472#  |       |       |     |
| BEGIN1 = 001570 | 194  | 475#  |       |       |     |







MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 CROSS REFERENCE TABLE

|        |          |       |       |       |       |       |       |       |       |       |       |       |       |       |
|--------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 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* |       |       |       |       |       |       |       |
|        |          | 73#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PIRQ   | = 177772 | 167#  |       |       |       |       |       |       |       |       |       |       |       |       |
| PIRQVE | = 000240 | 90#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR0    | = 000000 | 91#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR1    | = 000040 | 92#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR2    | = 000100 | 93#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR3    | = 000140 | 94#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR4    | = 000200 | 95#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR5    | = 000240 | 96#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR6    | = 000300 | 97#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PR7    | = 000340 | 70#   |       |       |       |       |       |       |       |       |       |       |       |       |
| PS     | = 177776 | 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* |       |       |       |       |       |       |       |       |       |
| RO     | =%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* |
| R1     | =%000001 | 2866* | 2869  | 2892* | 2910* |       |       |       |       |       |       |       |       |       |
|        |          | 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  |
| R2     | =%000002 | 2851* | 2856* | 2859  | 2865* | 2890  | 2893* | 2899* | 2902  | 2909* |       |       |       |       |
| R3     | =%000003 | 80#   | 1980  | 1984* | 1988* | 1990* | 1992* | 1998  | 1999* | 2278  | 2299* |       |       |       |
|        |          | 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* |
| R5     | =%000005 | 2672  | 2697* | 2702* |       |       |       |       |       |       |       |       |       |       |
|        |          | 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  |       |











MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 CROSS REFERENCE TABLE

|          |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
|----------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| \$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  |       |       |       |       |       |       |       |
| \$LPPI   | 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#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$MBAOR  | 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  |





MAINDEC-11-DRLPJ-A  
DRLPJ.P11

LPA/LPS-11-DRA DIAGNOSTIC  
CROSS REFERENCE TABLE

|                  |       |       |       |       |       |       |       |       |       |       |       |       |       |
|------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
|                  | 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# |       |       |       |       |
|                  | 2566  | 2620  | 2647  | 2761  | 2778  | 2927# |       |       |       |       |       |       |       |
| \$TOUT 020060    | 262#  | 2389* | 2400  |       |       |       |       |       |       |       |       |       |       |
| \$TPB 001152     | 266#  | 2338  | 2400  |       |       |       |       |       |       |       |       |       |       |
| \$TPFLG 001157   | 261#  | 2387  | 2400  |       |       |       |       |       |       |       |       |       |       |
| \$TPS 001150     | 489   | 2467# |       |       |       |       |       |       |       |       |       |       |       |
| \$TRAP 016210    | 2478# | 2489  |       |       |       |       |       |       |       |       |       |       |       |
| \$TRAP2 016232   | 2482# | 2491# | 2492# | 2493# | 2494# | 2495  | 2496# | 2497  | 2498# | 2499# | 2500# | 2501# |       |
| \$TRP = 000012   | 2472  | 2489# |       |       |       |       |       |       |       |       |       |       |       |
| \$TRPAD 016244   | 226#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$TSTM 001004    | 239#  | 1717* | 1856  | 1884  | 1906* | 1907  | 1912  | 1916  | 1933  | 1966  |       |       |       |
| \$TSTNM 001102   | 2114  | 2115  | 2132  | 2136# |       |       |       |       |       |       |       |       |       |
| \$TTYIN 014612   | 2494  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$TYPBN= ***** U | 2494  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$TYPOS= ***** U | 2338# | 2432  | 2482  | 2490  |       |       |       |       |       |       |       |       |       |
| \$TYPE 015460    | 2058  | 2368  | 2375  | 2382  | 2387# | 2388  |       |       |       |       |       |       |       |
| \$TYPEC 015672   | 2393  | 2395  | 2398# |       |       |       |       |       |       |       |       |       |       |
| \$TYPEX 015740   | 2222# | 2491  |       |       |       |       |       |       |       |       |       |       |       |
| \$TYPOC 015036   | 2221  | 2224# | 2493  |       |       |       |       |       |       |       |       |       |       |
| \$TYPON 015052   | 2217# | 2492  |       |       |       |       |       |       |       |       |       |       |       |
| \$TYPOS 015012   | 287#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$UNIT 001210    | 228#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$UNITM 001010   | 294#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$USWR 001222    | 3009# |       |       |       |       |       |       |       |       |       |       |       |       |
| \$UTK 020242     | 319#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$VECT1 001246   | 320#  |       |       |       |       |       |       |       |       |       |       |       |       |
| \$VECT2 001250   | 1871# |       |       |       |       |       |       |       |       |       |       |       |       |
| \$XTSTR 013346   | 1730# |       |       |       |       |       |       |       |       |       |       |       |       |
| \$SGE14= 000000  | 2218* | 2222* | 2232  | 2267# |       |       |       |       |       |       |       |       |       |
| \$OFILL 015235   | 1868  | 1940  |       |       |       |       |       |       |       |       |       |       |       |
| \$4OCAT= ***** U | 185#  | 189#  | 201   | 202#  | 204#  | 206#  | 207#  | 213   | 214#  | 216#  | 218#  | 236#  | 276   |





MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 CROSS REFERENCE TABLE

|        |       |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|--------|-------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | 2431 |      |      |      |      |      |      |      |      |
| PUSH   | 168#  | 1978 | 2276 | 2282 | 2408 | 2410 |      |      |      |      |      |      |      |      |      |
| 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 |





MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 CROSS REFERENCE TABLE

|        |      |      |
|--------|------|------|
| .SSAVE | 30#  |      |
| .SSCOP | 30#  | 1851 |
| .SSPAC | 30#  |      |
| .SSWDO | 30#  |      |
| .STLKW | 26#  | 2713 |
| .STOUT | 453# | 2918 |
| .STRAP | 30#  | 2459 |
| .STYPE | 30#  | 2321 |
| .STYPO | 30#  | 2192 |





MAINDEC-11-DRLPJ-A LPA/LPS-11-DRA DIAGNOSTIC  
DRLPJ.P11 CROSS REFERENCE TABLE

|      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|--|
| HALT | 189  | 1843 | 1952 | 1963 | 2285 | 2309 | 2340 | 2704 | 2831 |      |      |      |      |      |      |  |
| INC  | 530  | 1719 | 1901 | 1934 | 2069 | 2248 | 2256 | 2293 | 2443 | 2535 | 2553 | 2586 | 2588 | 2636 | 2681 |  |
| INCB | 2693 | 2812 | 2818 | 2856 | 2899 | 2932 |      |      |      |      |      |      |      |      |      |  |
| IOT  | 1906 | 1931 | 2396 | 2556 | 2607 | 2702 | 2989 |      |      |      |      |      |      |      |      |  |
| JMP  | 63   |      |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| JSR  | 193  | 194  | 195  | 539  | 932  | 1649 | 1736 | 1744 |      |      |      |      |      |      |      |  |
|      | 554  | 561  | 567  | 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 | 1643 | 1661 |  |
|      | 1663 | 1665 | 1668 | 1672 | 1680 | 1683 | 1685 | 1687 | 1690 | 1693 | 1695 | 1697 | 1731 | 1940 | 1946 |  |
|      | 2058 | 2349 | 2368 | 2375 | 2382 | 2432 | 2544 | 2566 | 2609 | 2611 | 2613 | 2620 | 2647 | 2723 | 2727 |  |
|      | 2732 | 2736 | 2755 | 2758 | 2761 | 2775 | 2778 | 2792 | 2862 | 2870 | 2904 | 2915 | 2952 | 3026 | 3043 |  |
|      | 3078 | 3085 |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| MOV  | 475  | 479  | 483  | 485  | 486  | 487  | 488  | 489  | 490  | 491  | 492  | 493  | 497  | 498  | 501  |  |
|      | 502  | 503  | 504  | 509  | 511  | 512  | 513  | 518  | 520  | 525  | 526  | 527  | 528  | 531  | 535  |  |
|      | 536  | 555  | 562  | 568  | 569  | 571  | 573  | 576  | 581  | 593  | 606  | 608  | 621  | 634  | 647  |  |
|      | 648  | 650  | 668  | 669  | 671  | 691  | 756  | 820  | 823  | 826  | 840  | 853  | 866  | 869  | 883  |  |
|      | 896  | 899  | 912  | 916  | 934  | 937  | 941  | 954  | 957  | 960  | 974  | 977  | 980  | 994  | 997  |  |
|      | 1000 | 1013 | 1015 | 1016 | 1018 | 1057 | 1058 | 1063 | 1066 | 1072 | 1090 | 1091 | 1096 | 1099 | 1102 |  |
|      | 1109 | 1128 | 1130 | 1131 | 1134 | 1138 | 1141 | 1146 | 1153 | 1156 | 1161 | 1168 | 1171 | 1176 | 1183 |  |
|      | 1186 | 1191 | 1198 | 1201 | 1206 | 1213 | 1216 | 1221 | 1228 | 1231 | 1236 | 1243 | 1246 | 1251 | 1258 |  |
|      | 1261 | 1266 | 1273 | 1276 | 1281 | 1288 | 1291 | 1296 | 1304 | 1314 | 1316 | 1317 | 1320 | 1324 | 1327 |  |
|      | 1332 | 1339 | 1342 | 1347 | 1354 | 1357 | 1362 | 1369 | 1372 | 1377 | 1384 | 1387 | 1392 | 1399 | 1402 |  |
|      | 1407 | 1414 | 1417 | 1422 | 1429 | 1432 | 1437 | 1444 | 1447 | 1452 | 1459 | 1462 | 1467 | 1474 | 1477 |  |
|      | 1482 | 1490 | 1499 | 1500 | 1518 | 1527 | 1544 | 1545 | 1547 | 1551 | 1554 | 1557 | 1585 | 1603 | 1604 |  |
|      | 1605 | 1610 | 1613 | 1616 | 1619 | 1622 | 1642 | 1644 | 1646 | 1656 | 1665 | 1669 | 1675 | 1677 | 1680 |  |
|      | 1687 | 1690 | 1724 | 1728 | 1848 | 1873 | 1874 | 1876 | 1879 | 1892 | 1904 | 1905 | 1908 | 1909 | 1912 |  |
|      | 1913 | 1933 | 1935 | 1956 | 1959 | 1976 | 1977 | 1978 | 1979 | 1980 | 1982 | 1997 | 1998 | 1999 | 2000 |  |
|      | 2001 | 2027 | 2051 | 2056 | 2085 | 2086 | 2113 | 2114 | 2129 | 2130 | 2131 | 2132 | 2153 | 2158 | 2167 |  |
|      | 2172 | 2177 | 2179 | 2183 | 2217 | 2225 | 2226 | 2227 | 2233 | 2240 | 2258 | 2259 | 2260 | 2261 | 2262 |  |
|      | 2274 | 2275 | 2276 | 2277 | 2278 | 2279 | 2280 | 2281 | 2282 | 2283 | 2284 | 2290 | 2291 | 2295 | 2296 |  |
|      | 2297 | 2298 | 2299 | 2300 | 2301 | 2302 | 2303 | 2306 | 2342 | 2343 | 2348 | 2356 | 2371 | 2409 | 2410 |  |
|      | 2417 | 2421 | 2426 | 2427 | 2429 | 2431 | 2441 | 2447 | 2448 | 2467 | 2468 | 2472 | 2478 | 2479 | 2519 |  |
|      | 2534 | 2542 | 2551 | 2555 | 2564 | 2604 | 2639 | 2672 | 2673 | 2674 | 2678 | 2685 | 2696 | 2697 | 2720 |  |
|      | 2721 | 2724 | 2725 | 2728 | 2737 | 2752 | 2753 | 2759 | 2794 | 2808 | 2822 | 2847 | 2848 | 2850 | 2859 |  |
|      | 2861 | 2865 | 2866 | 2868 | 2889 | 2890 | 2892 | 2902 | 2908 | 2909 | 2910 | 2913 | 2929 | 2981 | 3022 |  |
|      | 3030 | 3041 | 3077 | 3083 |      |      |      |      |      |      |      |      |      |      |      |  |
| MOVB | 496  | 1590 | 1907 | 1911 | 1937 | 1945 | 1985 | 2018 | 2035 | 2089 | 2095 | 2118 | 2123 | 2218 | 2219 |  |
|      | 2222 | 2223 | 2224 | 2228 | 2231 | 2232 | 2251 | 2353 | 2381 | 2389 | 2404 | 2405 | 2407 | 2470 | 2603 |  |
|      | 2608 | 2610 | 2612 | 2617 | 2645 | 2726 | 2729 | 2731 | 2733 | 2734 | 2735 | 2756 | 2757 | 2774 | 2776 |  |











|        |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| .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

K07

MACY11 27(654) 15-DEC-77 08:37 PAGE 82

SEQ 0089

MAINDEC-11-DRLPJ-A  
DRLPJ.P11

LPA/LPS-11-DRA DIAGNOSTIC

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