

LPS11/DRA

DIAGNOSTIC I/O TEST
MD-11-DZLPI-B

EP-DCFPB-B DL-A

OCT 1976

COPYRIGHT ©1976

digital

FICHE 1 OF 1

Made in U.S.A.

This microfiche card contains a grid of frames, each displaying diagnostic test data for the LPS11/DRA system. The data is organized into columns and rows, with some frames showing numerical values and others showing graphical representations like bar charts or waveforms. The frames are arranged in a grid that is approximately 10 frames wide and 20 frames high. The data appears to be organized into several sections, possibly corresponding to different components or tests. The overall layout is dense and technical, typical of diagnostic documentation from the 1970s.

.REM %

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZLPI-B
PRODUCT NAME:	LPS-11 DRA DIGITAL I/O TEST
DATE:	MAY 21, 1976
MAINTAINER:	DIAGNOSTIC GROUP

COPYRIGHT (C) 1973, 1976 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY 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 ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

THIS PROGRAM IS A LOGIC TEST OF THE "LPS-11-DRA" DIGITAL INPUT OUTPUT CONTROL OPTION. ALL FUNCTIONS OF THE OPTION WILL BE TESTED. DUE TO THE FLEXIBILITY OF THE OPTION, THE OPERATOR MAY BE REQUIRED TO SUPPLY OPTION CHARASTICS. THE PROGRAM WILL HANDLE ALL CONFIGURATIONS OF SW15 THRU SW00 AND JUMPERS W15 THRU W00. THE FOLLOWING JUMPERS MUST BE INSERTED TO EXECUTE THE LOGIC TEST: W16-W17-W18-W19.

2.1 EQUIPMENT

PDP-11 FAMILY COMPUTER WITH 4K WORDS OF MEMORY
LPS-11 WITH DRA OPTION INSTALLED
TELETYPE

2.2 STORAGE

THIS PROGRAM USES 4K 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 TYPEDS
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.

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.

THIS PROGRAM USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR
ERROR REPORTING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE"
FOR TYPE AND DESCRIPTION OF ERRORS.

THE OPERATOR MUST SUPPLY THE CORRECT JUMPER AND SWITCH CONFIGURATION OR AN ERROR WILL OCCUR.

8.1 EXECUTION TIME

THE LOGIC TEST WILL TAKE APPROXIMATELY 60 SECONDS FOR COMPLETION
AND WILL TYPE 'END PASS'.
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.

146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175

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. TABLE OF CONTENTS

ATTACHED.

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
225
226
227
228
229
230
231

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

100000
040000

```

%
.TITLE MAINDEC-11-DZLPI-B      LPS-11-DRA DIAGNOSTIC
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY RAYMOND SHOOP
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-82), NOV 21, 1975.
;*
```

.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

STACK= 1100

.EQUIV EMT,ERROR

;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE

;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS

HT= 11

;;CODE FOR HORIZONTAL TAB

LF= 12

;;CODE FOR LINE FEED

CR= 15

;;CODE FOR CARRIAGE RETURN

CRLF= 200

;;CODE FOR CARRIAGE RETURN-LINE FEED

PS= 177776

;;PROCESSOR STATUS WORD

.EQUIV PS,PSW

STKLMT= 177774

;;STACK LIMIT REGISTER

PIRQ= 177772

;;PROGRAM INTERRUPT REQUEST REGISTER

DSWR= 177570

;;HARDWARE SWITCH REGISTER

DDISP= 177570

;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS

R0= %0

;;GENERAL REGISTER

R1= %1

;;GENERAL REGISTER

R2= %2

;;GENERAL REGISTER

R3= %3

;;GENERAL REGISTER

R4= %4

;;GENERAL REGISTER

R5= %5

;;GENERAL REGISTER

R6= %6

;;GENERAL REGISTER

R7= %7

;;GENERAL REGISTER

.EQUIV R6,SP

;;STACK POINTER

.EQUIV R7,PC

;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS

PR0= 0

;;PRIORITY LEVEL 0

PR1= 40

;;PRIORITY LEVEL 1

PR2= 100

;;PRIORITY LEVEL 2

PR3= 140

;;PRIORITY LEVEL 3

PR4= 200

;;PRIORITY LEVEL 4

PR5= 240

;;PRIORITY LEVEL 5

PR6= 300

;;PRIORITY LEVEL 6

PR7= 340

;;PRIORITY LEVEL 7

;*SWITCH REGISTER SWITCH DEFINITIONS

SW15= 100000

SW14= 40000

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLPIB.P11 BASIC DEFINITIONS

Address	Hex	ASCII	Label	Value
000000	020000		SW13=	20000
000001	010000		SW12=	10000
000002	004000		SW11=	4000
000003	002000		SW10=	2000
000004	001000		SW09=	1000
000005	000400		SW08=	400
000006	000200		SW07=	200
000007	000100		SW06=	100
000008	000040		SW05=	40
000009	000020		SW04=	20
000010	000010		SW03=	10
000011	000004		SW02=	4
000012	000002		SW01=	2
000013	000001		SW00=	1
000014			.EQUIV	SW09, SW9
000015			.EQUIV	SW08, SW8
000016			.EQUIV	SW07, SW7
000017			.EQUIV	SW06, SW6
000018			.EQUIV	SW05, SW5
000019			.EQUIV	SW04, SW4
000020			.EQUIV	SW03, SW3
000021			.EQUIV	SW02, SW2
000022			.EQUIV	SW01, SW1
000023			.EQUIV	SW00, SW0
000024				
000025				
000026				
000027				
000028				
000029				
000030				
000031				
000032				
000033				
000034				
000035				
000036				
000037				
000038				
000039				
000040				
000041				
000042				
000043				
000044				
000045				
000046				
000047				
000048				
000049				
000050				
000051				
000052				
000053				
000054				
000055				
000056				
000057				
000058				
000059				
000060				
000061				
000062				
000063				
000064				
000065				
000066				
000067				
000068				
000069				
000070				
000071				
000072				
000073				
000074				
000075				
000076				
000077				
000078				
000079				
000080				
000081				
000082				
000083				
000084				
000085				
000086				
000087				
000088				
000089				
000090				
000091				
000092				
000093				
000094				

H01

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 BASIC DEFINITIONS

MACY11 27(732) 17-SEP-76 14:21 PAGE 7

288
289
290
291
292
293
294
295
296
297
298
299
300

000014
000014
000014
000020
000024
000030
000034
000060
000064
000240
170410
000350
000200

TBITVEC=14
TRTVEC= 14
BPTVEC= 14
IOTVEC= 20
PWRVEC= 24
EMTVEC= 30
TRAPVEC=34
TKVEC= 60
TPVEC= 64
PIRQVEC=240
ABASE=170410
AVECT1=350
APRIOR=200

:: "T" BIT
:: TRACE TRAP
:: BREAKPOINT TRAP (BPT)
:: INPUT/OUTPUT TRAP (IOT) **SCOPE**
:: POWER FAIL
:: EMULATOR TRAP (EMT) **ERROR**
:: "TRAP" TRAP
:: TTY KEYBOARD VECTOR
:: TTY PRINTER VECTOR
:: PROGRAM INTERRUPT REQUEST VECTOR


```

301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328

```

```

.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
000174 000174 000000 DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
000176 000000 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

```

```

329
330
331 .SBTTL ACT11 HOOKS
332
333 ;*****
334 ;HOOKS REQUIRED BY ACT11
335         $SVPC=.          ;SAVE PC
336         .=46
337 000046 $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
338         .=52
339 000052 .WORD 0          ;;2)SET LOC.52 TO ZERO
340         .=$SVPC          ;; RESTORE PC
341         .=1000
342
343 .SBTTL APT PARAMETER BLOCK
344
345 ;*****
346 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
347 ;*****
348         .SX=.          ;;SAVE CURRENT LOCATION
349         .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
350 000024 200          ;;FOR APT START UP
351         .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
352 000044 $APTHDR      ;;POINT TO APT HEADER BLOCK
353         .=$X          ;;RESET LOCATION COUNTER
354 ;*****
355 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
356 ;INTERFACE SPEC.
357
358 001000 $APTHD:
359 001000 000000 $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
360 001002 001174 $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
361 001004 000012 $STMT: .WORD 10.        ;;RUN TIM OF LONGEST TEST
362 001006 000074 $PASTM: .WORD 60.       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
363 001010 000074 $UN1TM: .WORD 60.      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
364 001012 000052 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```


365		
366		
367		
368		
369		
370		
371		
372		001100
373	001100	
374	001100	000000
375	001102	000
376	001103	000
377	001104	000000
378	001106	000000
379	001110	000000
380	001112	000000
381	001114	000
382	001115	001
383	001116	000000
384	001120	000000
385	001122	000000
386	001124	000000
387	001126	000000
388	001130	000000
389	001132	000000
390	001134	000000
391	001136	177570
392	001140	177570
393	001142	177560
394	001144	177562
395	001146	177564
396	001150	177566
397	001152	000
398	001153	002
399	001154	012
400	001155	000
401	001156	000000
402		
403	001160	000000
404	001162	000000
405	001164	000000
406	001166	000000
407	001170	077
408	001171	015
409	001172	000012

```

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

```

SCMTAG:		
STSTNM:	.WORD	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
SEDADR:	.WORD	0
SGDDAT:	.WORD	0
SBDDAT:	.WORD	0
	.WORD	0
	.WORD	0
	.WORD	0
SWR:	.WORD	DSWR
DISPLAY:	.WORD	DDISP
STKS:	177560	
STKB:	177562	
STPS:	177564	
STPB:	177566	
SNUL:	.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	/?/
SCRFL:	.ASCII	<15>
SLF:	.ASCII	<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

:;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 ($REG0) WAS OBTAINED
:;CONTAINS (($REGAD)+0)
:;CONTAINS (($REGAD)+2)
:;MAX. NUMBER OF ITERATIONS
:;ESCAPE ON ERROR ADDRESS
:;QUESTION MARK
:;CARRIAGE RETURN
:;LINE FEED

```

```

410 ;:*****
411
412 .SBTTL APT MAILBOX-ETABLE
413
414 ;:*****
415 .EVEN
416 $MAIL: ;:APT MAILBOX
417 $MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
418 $FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
419 $TESTN: .WORD ATESTN ;:TEST NUMBER
420 $PASS: .WORD APASS ;:PASS COUNT
421 $DEVCT: .WORD ADEVCT ;:DEVICE COUNT
422 $UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
423 $MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS
424 $MSGLG: .WORD AMSGLG ;:MESSAGE LENGTH
425 $ETABLE: ;:APT ENVIRONMENT TABLE
426 $ENV: .BYTE AENV ;:ENVIRONMENT BYTE
427 $ENVN: .BYTE AENVN ;:ENVIRONMENT MODE BITS
428 $SWREG: .WORD ASWREG ;:APT SWITCH REGISTER
429 $USWR: .WORD AUSWR ;:USER SWITCHES
430 $CPUOP: .WORD ACPUOP ;:CPU TYPE, OPTIONS
431 ;:BITS 15-11=CPU TYPE
432 ;:11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
433 ;:11/70=06, PDQ=07, Q=10
434 ;:BIT 10=REAL TIME CLOCK
435 ;:BIT 9=FLOATING POINT PROCESSOR
436 ;:BIT 8=MEMORY MANAGEMENT
437 $MAMS1: .BYTE AMAMS1 ;:HIGH ADDRESS, M.S. BYTE
438 $MTYP1: .BYTE AMTYP1 ;:MEM. TYPE, BLK#1
439 ;:MEM. TYPE BYTE -- (HIGH BYTE)
440 ;:900 NSEC CORE=001
441 ;:300 NSEC BIPOLAR=002
442 ;:500 NSEC MOS=003
443 $MADR1: .WORD AMADR1 ;:HIGH ADDRESS, BLK#1
444 ;:MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
445 $MAMS2: .BYTE AMAMS2 ;:HIGH ADDRESS, M.S. BYTE
446 $MTYP2: .BYTE AMTYP2 ;:MEM. TYPE, BLK#2
447 $MADR2: .WORD AMADR2 ;:MEM. LAST ADDRESS, BLK#2
448 $MAMS3: .BYTE AMAMS3 ;:HIGH ADDRESS, M.S. BYTE
449 $MTYP3: .BYTE AMTYP3 ;:MEM. TYPE, BLK#3
450 $MADR3: .WORD AMADR3 ;:MEM. LAST ADDRESS, BLK#3
451 $MAMS4: .BYTE AMAMS4 ;:HIGH ADDRESS, M.S. BYTE
452 $MTYP4: .BYTE AMTYP4 ;:MEM. TYPE, BLK#4
453 $MADR4: .WORD AMADR4 ;:MEM. LAST ADDRESS, BLK#4
454 $VECT1: .BYTE AVECT1 ;:INTERRUPT VECTOR#1
455 $VECT2: .BYTE AVECT2 ;:INTERRUPT VECTOR#2
456 $PRIOR: .BYTE APRIOR ;:BUS PRIORITY #1, #2
457 ;:0 ;:SPARE, NOT USED
458 .EVEN
459 $BASE: .WORD ABASE ;:BASE ADDRESS OF EQUIPMENT UNDER TEST
460 $DEVN: .WORD ADEVN ;:DEVICE MAP
461 $CDW1: .WORD ACDW1 ;:CONTROLLER DESCRIPTION WORD#1
462 $CDW2: .WORD ACDW2 ;:CONTROLLER DESCRIPTION WORD#2
463 $DDW0: .WORD ADDW0 ;:DEVICE DESCRIPTOR WORD#0
464 $DDW1: .WORD ADDW1 ;:DEVICE DESCRIPTOR WORD#1
465 $DDW2: .WORD ADDW2 ;:DEVICE DESCRIPTOR WORD#2

```


466 001266 000000
 467 001270 000000
 468 001272 000000
 469 001274 000000
 470 001276 000000
 471 001300 000000
 472 001302 000000
 473 001304 000000
 474 001306 000000
 475 001310 000000
 476 001312 000000
 477 001314 000000
 478 001316 000000

\$DDW3: .WORD ADDW3 ;;DEVICE DESCRIPTOR WORD#3
 \$DDW4: .WORD ADDW4 ;;DEVICE DESCRIPTOR WORD#4
 \$DDW5: .WORD ADDW5 ;;DEVICE DESCRIPTOR WORD#5
 \$DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
 \$DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
 \$DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
 \$DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
 \$DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
 \$DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
 \$DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
 \$DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
 \$DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
 \$DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15

479
 480
 481 001320

\$ETEND:

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

492
 493
 494
 495
 496
 497
 498 001320

\$ERRTB:

499
 500
 501 001320 010312
 502 001322 010732
 503 001324 011116
 504 001326 011144

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

505
 506
 507 001330 010343
 508 001332 010763
 509 001334 011116
 510 001336 011144

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

511
 512
 513 001340 010373
 514 001342 011014
 515 001344 011116
 516 001346 011144

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

517
 518
 519 001350 010424
 520 001352 011045
 521 001354 011126

;ITEM 4
 EM4 ;;INPUT FAILED TO INTERRUPT
 DH4 ;;ERRPC
 DT4 ;;\$ERRPC

NO1

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 ERROR POINTER TABLE

MACY11 27(732) 17-SEP-76 14:21 PAGE 13

522	001356	011144	DF1	
523				
524				
525	001360	010456	; ITEM 5	
526	001362	011045	EM5	: OUTPUT FAILED TO INTERRUPT
527	001364	011126	DH4	: ERRPC
528	001366	011144	DT4	: SERRPC
529			DF1	
530				
531				
532	001370	010511	; ITEM 6	
533	001372	011045	EM6	: UNEXPECTED INTERRUPT
534	001374	011126	DH4	: ERRPC
535	001376	011144	DT4	: SERRPC
			DF1	

B02

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLPIS.P11 ERROR POINTER TABLE

MACY11 27(732) 17-SEP-76 14:21 PAGE 14

001400	010536	:ITEM 7	EM7	:OPERATOR INTERVENTION ERROR
001402	011045		DH4	:ERRPC
001404	011126		DT4	:SERRPC
001406	011144		DF1	
001410	010572	:ITEM 10	EM10	:INTERRUPT INPUT BIT FAILED TO SET INPUT READY
001412	011053		DH10	:ERRPC STATUS EXPECTED INPUT BIT
001414	011132		DT10	:SERRPC \$BDDAT \$GDDAT BRLEV3
001416	011150		DF10	
001420	010655	:ITEM 11	EM11	:NON-INTERRUPTING INPUT BIT SET INPUT READY
001422	011053		DH10	:ERRPC STATUS EXPECTED INPUT BIT
001424	011132		DT10	:SERRPC \$BDDAT \$GDDAT BRLEV3
001426	170410	DRADD: 170410		:DRA STARTING ADDRESS
001430	000350	DRIV: 350		:DRA STARTING INTERRUPT VECTOR
001432	000200	DRBRL: 200		:DRA BR LEVEL
001434	170410	GRSTAT: 170410		:DR STATUS
001436	170412	GRDAI: 170412		:INPUT REG.
001438	170414	GRDIO: 170414		:OUTPUT REG.
001440	170415	GRBHIO: 170415		:OUTPUT REG HIGH BYTE
001442	000000	NOTLCH: 0		
001444	000000	NOTINT: 0		
001446	000310	GRIVA: 310		
001448	000312	GRIVSA: 312		
001450	000314	GRIVB: 314		
001452	000316	GRIVSB: 316		
001454	000200	DIOBRL: 200		
001456	000000	BRLEV1: 0		
001458	000000	BRLEV2: 0		
001460	000000	BRLEV3: 0		


```

*****
: DIGITAL I-O LOGIC TEST
*****
BEGIN: CLR RO
      BR RBEG
BEGIN1: MOV #-1, RO
RBEG: RESET
: CLEAR THE COMMON TAGS ($CMTAG) AREA
      MOV $CMTAG, R6 : FIRST LOCATION TO BE CLEARED
      CLR (R6)+ : CLEAR MEMORY LOCATION
      CMP $BDDAT, R6 : DONE?
      BNE -6 : LOOP BACK IF NO
      MOV $STACK, SP : SETUP THE STACK POINTER
: INITIALIZE A FEW VECTORS
      MOV $SCOPE, $IOTVEC : IOT VECTOR FOR SCOPE ROUTINE
      MOV $340, $IOTVEC+2 : LEVEL 7
      MOV $ERROR, $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 $ENDCT, $EOPCT : SETUP END-OF-PROGRAM COUNTER
      CLR $TIMES : INITIALIZE NUMBER OF ITERATIONS
      CLR $ESCAPE : CLEAR THE ESCAPE ON ERROR ADDRESS
      MOVB $1, $ERMAX : ALLOW ONE ERROR PER TEST
      MOV $., $LPADR : INITIALIZE THE LOOP ADDRESS FOR SCOPE
      MOV $., $LPERR : SETUP THE ERROR LOOP ADDRESS
: SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
: EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
      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, $SWR : TRY TO REFERENCE HARDWARE SWR
      BNE 65$ : BRANCH IF NO TIMEOUT TRAP OCCURRED
      : AND THE HARDWARE SWR IS NOT = -1
      TST $#1 : FORCE A TRAP THROUGH ERRVEC
      MOV $SWREG, SWR : POINT TO SOFTWARE SWR
      MOV $DISPREG, DISPLAY : POINT TO SOFTWARE DISPLAY REG
      MOV $65$, (SP) : REPLACE OLD PC WITH NEW
      RTI : RESTORE PC AND PSW
      MOV (SP)+, $ERRVEC : RESTORE ERROR VECTOR
$ARG1: CLR $PASS : CLEAR PASS COUNT
      BITB $APTSIZE, $ENVN : TEST USER SIZE UNDER APT
      BEQ 64$ : YES, USE NON-APT SWITCH
      MOV $SSWREG, SWR : NO, USE APT SWITCH REGISTER
64$:

```



```

623 001754 013737 001250 001426 DONOT: MOV $BASE,DRADD
624 001762 013737 001244 001430 MOV $VECT1,DRIV
625 001770 013737 001246 001432 MOV $PRIOR,DRBAL
626 001776 005700 TST R0 ;TEST R0
627 002000 001402 BEQ 2$ ;BR IF CLEARED
628 002002 000137 002470 1$: JMP 3$ ;JUMP IF SET
629 002006 005737 000042 2$: TST 0#42 ;TEST IF IN "CHAIN MODE"
630 002012 001373 BNE 1$ ;BR IF YES
631 002014 104400 002022 TYPE 65$ ;;TYPE ASCIZ STRING
632 002020 000430 BR 64$ ;;GET OVER THE ASCIZ
633 002102 104400 002110 65$: .ASCIZ <15><12><15><12>/LPS-11-DRA DIGITAL INPUT OUTPUT LOGIC TEST/
634 002102 000413 TYPE 67$ ;;TYPE ASCIZ STRING
635 002106 BR 66$ ;;GET OVER THE ASCIZ
636 002136 104400 002144 67$: .ASCIZ <15><12>/MAINDEC-11-DZLPI-B/
637 002136 000435 TYPE 69$ ;;TYPE ASCIZ STRING
638 002142 BR 68$ ;;GET OVER THE ASCIZ
639 002236 004737 011170 69$: .ASCIZ <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-LATCHING BITS/
640 002236 017737 176670 001444 JSR PC,GETSWR ;GET THE SWITCH REGISTER VALUE
641 002242 104400 002256 MOV 0SWR,NOTLCH ;SAVE SWITCHES
642 002250 000437 TYPE 71$ ;;TYPE ASCIZ STRING
643 002254 BR 70$ ;;GET OVER THE ASCIZ
644 002354 71$: .ASCIZ <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-INTERRUPTING BITS/
645 70$:

```

E02

 MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
 DZLP1B.P11 ERROR POINTER TABLE

MACY11 27(732) 17-SEP-76 14:21 PAGE 17

```

649
650 002354 004737 011170 JSR PC,GETSWR ;GET THE SWITCH REGISTER VALUE
651 002360 017737 176552 001446 MOV JSWR,NOTINT ;SAVE SWITCHES
652 002366 104400 002374 TYPE .73$ ;TYPE ASCIZ STRING
653 002372 000434 BR .72$ ;GET OVER THE ASCIZ
654 ;.73$: .ASCIZ <15><12>/SET SWITCH REGISTER WITH THE DESIRED PROGRAM OPTIONS/
655 72$:
656 002464 004737 011170 JSR PC,GETSWR ;GET THE SWITCH REGISTER VALUE
657 002470 013737 001426 001434 MOV DRADD,GRSTAT ;LOAD INITIAL ADDRESS
658 002476 013737 001426 001436 MOV DRADD,GRDAI ;LOAD 2ND ADDRESS
659 002504 062737 000002 001436 ADD #2,GRDAI
660 002512 013737 001426 001440 MOV DRADD,GRDIO ;LOAD 3RD ADDRESS
661 002520 062737 000004 001440 ADD #4,GRDIO
662 002526 013737 001426 001442 MOV DRADD,GRBHIO ;LOAD 4TH ADDRESS
663 002534 062737 000005 001442 ADD #5,GRBHIO
664 002542 013737 001430 001450 MOV DRIV,GRIVA ;LOAD FIRST VECTOR
665 002550 013737 001430 001452 MOV DRIV,GRIVSA
666 002556 062737 000002 001452 ADD #2,GRIVSA
667 002564 013737 001430 001454 MOV DRIV,GRIVB ;LOAD 2ND VECTOR
668 002572 062737 000004 001454 ADD #4,GRIVB
669 002600 013737 001430 001456 MOV DRIV,GRIVSB
670 002606 062737 000006 001456 ADD #6,GRIVSB
671 002614 013737 001432 001460 MOV DRBRL,DIOBRL ;LOAD BR LEVEL
672 002622 000005 IOTEST: RESET
673 002624 012706 001100 MOV #STACK,SP ;LOAD STACK
674 002630 013777 001452 176612 MOV GRIVSA,@GRIVA ;RESET INPUT VECTOR
675 002636 005077 176610 CLR @GRIVSA
676 002642 013777 001456 176604 MOV GRIVSB,@GRIVB ;RESET OUTPUT VECTOR
677 002650 005077 176602 CLR @GRIVSB
678 ;*****
679 ;*TEST 1 TEST FOR NO BUS ERRORS
680 ;*****
681 002654 000004 TST1: SCOPE
682 002656 005077 176552 CLR @GRSTAT
683 002662 005077 176550 CLR @GRDAI
684 002666 005077 176546 CLR @GRDIO
685 ;*****
686 ;*TEST 2 TEST THAT OUTPUT REG. CAN HOLD #-1
687 ;*****
688 ;*****
689 002672 000004 TST2: SCOPE
690 002674 012737 177777 001124 MOV #-1,$GDDAT ;LOAD EXPECTED
691 002702 012777 177777 176530 MOV #-1,@GRDIO ;ALL ONES TO REGISTER
692 002710 017737 176524 001126 MOV @GRDIO,$BDDAT ;READ OUTPUT REG.
693 002716 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE
694 002724 001401 BEQ TST3 ;BR IF EQUAL
695 002726 104003 ERROR 3 ;REG WILL NOT HOLD ONES

```


F02

MAINDEC-11-DZLPI-B
DZLPIB.P11 T2LPS-11-DRA DIAGNOSTIC
TEST THAT OUTPUT REG. CAN HOLD #-1

MACY11 27(732) 17-SEP-76 14:21 PAGE 18

```

696
697
698
699
700 002730 000004
701 002732 012737 000040 001164
702 002740 005037 001124
703 002744 012777 177777 176466
704 002752 000005
705 002754 017737 176460 001126
706 002762 001401
707 002754 104003
708
709
710
711
712 002766 000004
713 002770 012737 052525 001124
714 002776 012777 052525 176434
715 003004 017737 176430 001126
716 003012 023737 001124 001126
717 003020 001401
718 003022 104003
719
720
721
722
723 003024 000004
724 003026 012737 125252 001124
725 003034 012777 125252 176376
726 003042 017737 176372 001126
727 003050 023737 001124 001126
728 003056 001401
729 003060 104003
730
731
732
733
734 003062 000004
735 003064 012737 000004 001164
736 003072 012737 003104 001110
737 003100 005037 001124
738 003104 013777 001124 176326
739 003112 017737 176322 001126
740 003120 023737 001124 001126
741 003126 001401
742 003130 104003
743 003132 005237 001124
744 003136 001362

;*****
;*TEST 3 TEST THAT RESET CLEARS OUTPUT REG.
;*****
TST3: SCOPE
MOV #40,$TIMES ;;DO 40 ITERATIONS
CLR $GDDAT
MOV #-1,$GRDIO
RESET ;SET DATA TO ALL ONES
MOV $GRDIO,$BDDAT ;READ OUTPUT REG.
BEQ TST4 ;;BR IF EQUAL
ERROR 3 ;REG FAILED TO CLEAR

;*****
;*TEST 4 TEST THAT OUTPUT REG. CAN HOLD #52525
;*****
TST4: SCOPE
MOV #52525,$GDDAT ;LOAD EXPECTED VALUE
MOV #52525,$GRDIO
MOV $GRDIO,$BDDAT ;READ OUTPUT REG.
CMP $GDDAT,$BDDAT ;COMPARE
BEQ TST5 ;;BR IF EQUAL
ERROR 3 ;DATA NOT=52525

;*****
;*TEST 5 TEST THAT OUTPUT REG. CAN HOLD #125252
;*****
TST5: SCOPE
MOV #125252,$GDDAT ;LOAD EXPECTED VALUE
MOV #125252,$GRDIO
MOV $GRDIO,$BDDAT ;READ OUTPUT REG.
CMP $GDDAT,$BDDAT ;COMPARE
BEQ TST6 ;;BR IF EQUAL
ERROR 3 ;DATA NOT=125252

;*****
;*TEST 6 TEST THAT OUTPUT REG. CAN HOLD A COUNT PATTERN
;*****
TST6: SCOPE
MOV #4,$TIMES ;;DO 4 ITERATIONS
MOV #2,$SLPERR ;LOAD SCOPE ERROR RETURN
CLR $GDDAT ;CLEAR PATTERN
MOV $GDDAT,$GRDIO ;LOAD THE OUTPUT REG.
MOV $GRDIO,$BDDAT ;READ THE REG.
CMP $GDDAT,$BDDAT ;COMPARE
BEQ 1$ ;;BR IF EQUAL
ERROR 3 ;OUTPUT REG. FAILED TO HOLD COUNT PATTERN
INC $GDDAT ;UPDATE THE PATTERN
BNE 2$ ;TRY AGAIN
2$: MOV $GDDAT,$GRDIO
MOV $GRDIO,$BDDAT
CMP $GDDAT,$BDDAT
BEQ 1$
ERROR 3
1$: INC $GDDAT
BNE 2$

```

G02

MAINDEC-11-DZLPI-B
DZLPIB.P11

T7

LPS-11-DRA DIAGNOSTIC
FLOAT A 1 ACROSS THE OUTPUT REGISTER

MACY11 27(732) 17-SEP-76 14:21 PAGE 19

```

745
746
747
748 003140 000004
749 003142 012737 003156 001110
750 003150 012737 000001 001124
751
752 003156 005077 176256 1S: CLR QGRDIO ;CLEAR OUTPUT
753 003162 053777 001124 176250 BIS $GDDAT,QGRDIO ;SET THAT BIT
754 003170 017737 176244 001126 MOV QGRDIO,$BDDAT ;READ OUTPUT REG.
755 003176 023737 001124 001126 CMP $GDDAT,$BDDAT ;TEST RESULTS
756 003204 001401 BEQ 2S ;BR IF EQUAL ?
757 003206 104003 ERROR 3
758
759 003210 006337 001124 2S: ASL $GDDAT ;SHIFT EXPECTED DATA
760 003214 001360 BNE 1S ;BR UNTIL DONE
761
762
763
764 003216 000004
765 003220 012737 003234 001110
766 003226 012737 000001 001124
767
768 003234 012777 177777 176176 1S: MOV #-1,QGRDIO ;LOAD OUTPUT TO A ONE
769 003242 043777 001124 176170 BIC $GDDAT,QGRDIO ;CLEAR A BIT
770 003250 017737 176164 001126 MOV QGRDIO,$BDDAT ;READ OUTPUT REG.
771 003256 005137 001126 COM $BDDAT ;COMPLEMENT IT
772 003262 023737 001124 001126 CMP $GDDAT,$BDDAT ;EQUAL ?
773 003270 001401 BEQ 2S ;BR IF EQUAL
774 003272 104003 ERROR 3
775
776 003274 006337 001124 2S: ASL $GDDAT ;SHIFT LEFT
777 003300 001355 BNE 1S ;BRANCH UNTIL DONE
778
779
780
781
782 003302 000004
783 003304 012737 125252 001124
784 003312 013777 001124 176120
785 003320 005177 176114
786 003324 005177 176110
787 003330 005177 176104
788 003334 005177 176100
789 003340 005177 176074
790 003344 005177 176070
791 003350 005177 176064
792 003354 005177 176060
793 003360 005177 176054
794 003364 005177 176050
795 003370 000240
796 003372 000240
797 003374 017737 176040 001126
798 003402 023737 001124 001126
799 003410 001401
800 003412 104003

*****
*TEST 7 FLOAT A 1 ACROSS THE OUTPUT REGISTER
*****
TST7: SCOPE
MOV #1S,$LPERR ;LOAD SCOPE ERROR RETURN
MOV #BIT0,$GDDAT ;LOAD EXPECTED VALUE

1S: CLR QGRDIO ;CLEAR OUTPUT
BIS $GDDAT,QGRDIO ;SET THAT BIT
MOV QGRDIO,$BDDAT ;READ OUTPUT REG.
CMP $GDDAT,$BDDAT ;TEST RESULTS
BEQ 2S ;BR IF EQUAL ?
ERROR 3

2S: ASL $GDDAT ;SHIFT EXPECTED DATA
BNE 1S ;BR UNTIL DONE

*****
*TEST 10 FLOAT A 0 ACROSS THE OUTPUT REGISTER
*****
TST10: SCOPE
MOV #1S,$LPERR ;LOAD SCOPE ERROR RETURN
MOV #BIT0,$GDDAT ;LOAD EXPECTED VALUE

1S: MOV #-1,QGRDIO ;LOAD OUTPUT TO A ONE
BIC $GDDAT,QGRDIO ;CLEAR A BIT
MOV QGRDIO,$BDDAT ;READ OUTPUT REG.
COM $BDDAT ;COMPLEMENT IT
CMP $GDDAT,$BDDAT ;EQUAL ?
BEQ 2S ;BR IF EQUAL
ERROR 3

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

*****
*TEST 11 TEST FOR SLOW OUTPUT GATES WITH #125252
*****
TST11: SCOPE
MOV #125252,$GDDAT ;LOAD EXPECTED VALUE
MOV $GDDAT,QGRDIO ;LOAD OUTPUT
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
COM QGRDIO ;COMPLEMENT OUTPUT REG.
NOP
NOP
MOV QGRDIO,$BDDAT ;READ OUTPUT REG.
CMP $GDDAT,$BDDAT ;TEST REGISTER
BEQ TST12 ;BR IF CORRECT
ERROR 3

```


H02

MAINDEC-11-DZLPI-B
DZLPIB.P11 T11LPS-11-DRA DIAGNOSTIC
TEST FOR SLOW OUTPUT GATES WITH #125252

MACY11 27(732) 17-SEP-76 14:21 PAGE 20

```

801
802
803
804
805 003414 000004
806 003416 012737 052525 001124
807 003424 013777 001124 176006
808 003432 005177 176002
809 003436 005177 175776
810 003442 005177 175772
811 003446 005177 175766
812 003452 005177 175762
813 003456 005177 175756
814 003462 005177 175752
815 003466 005177 175746
816 003472 005177 175742
817 003476 005177 175736
818 003502 000240
819 003504 017737 175730 001126
820 003512 023737 001124 001126
821 003520 001401
822 003522 104003
823
824
825
826
827 003524 000004
828 003526 012737 000400 001164
829 003534 012737 003546 001110
830 003542 005037 001124
831 003546 113777 001124 175664
832 003554 017737 175660 001126
833 003562 123737 001124 001126
834 003570 001401
835 003572 104003
836 003574 105237 001124
837 003600 001362
838
839
840
841
842 003602 000004
843 003604 012737 000400 001164
844 003612 012737 003624 001110
845 003620 005037 001124
846 003624 113777 001125 175610
847 003632 117737 175604 001127
848 003640 123737 001125 001127
849 003646 001401
850 003650 104003
851 003652 105237 001125
852 003656 001362

*****
;TEST 12 TEST FOR SLOW OUTPUT GATES WITH #52525
*****
TST12: SCOPE
MOV #52525,$GDDAT ;LOAD EXPECTED VALUE
MOV $GDDAT,$GRDIO ;LOAD OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
COM $GRDIO ;COMPLEMENT OUTPUT REG.
NOP
MOV $GRDIO,$BDDAT ;READ OUTPUT REG.
CMP $GDDAT,$BDDAT ;TEST PATTERN
BEQ TST13 ;;BR IF EQUAL
ERROR 3 ;OUTPUT REGISTER IN ERROR

*****
;TEST 13 TEST THAT OUTPUT CAN HOLD BYTE COUNT PATTERN
*****
TST13: SCOPE
MOV #400,$TIMES ;;DO 400 ITERATIONS
MOV #2,$LPERR ;LOAD SCOPE ERROR RETURN
CLR $GDDAT ;CLEAR PATTERN
2$: MOVB $GDDAT,$GRDIO ;LOAD THE OUTPUT
MOV $GRDIO,$BDDAT ;READ OUTPUT REG.
CMPB $GDDAT,$BDDAT
BEQ 1$ ;BRANCH IF EQUAL
ERROR 3 ;ERROR, LOW BYTE HAS BAD DATA
1$: INCB $GDDAT ;UPDATE PATTERN
BNE 2$

*****
;TEST 14 TEST THAT OUTPUT CAN HOLD HIGH BYTE COUNT PATTERN
*****
TST14: SCOPE
MOV #400,$TIMES ;;DO 400 ITERATIONS
MOV #2,$LPERR ;LOAD SCOPE ERROR RETURN
CLR $GDDAT ;CLEAR PATTERN
2$: MOVB $GDDAT+1,$GRBHIO ;LOAD THE HIGH BYTE
MOVB $GRBHIO,$BDDAT+1 ;READ HIGH BYTE OUTPUT REG.
CMPB $GDDAT+1,$BDDAT+1
BEQ 1$ ;BRANCH IF EQUAL
ERROR 3 ;ERROR, HIGH BYTE IN ERROR
1$: INCB $GDDAT+1
BNE 2$

```

```

853                                     ;*****
854                                     ;*TEST 15      TEST RELAY #1 STATUS BIT
855                                     ;*****
856 003660 000004                       TST15: SCOPE
857 003662 005077 175552                CLR      QGRDIO
858 003666 012777 177777 175542        MOV      #-1,QGRDAI
859 003674 012737 000001 001124        MOV      #BIT0,$GDDAT      ;LOAD EXPECTED
860 003702 012777 000001 175524        MOV      #BIT0,QGRSTAT      ;SET BIT 0
861 003710 017737 175520 001126        MOV      QGRSTAT,$BDDAT      ;READ STATUS
862 003716 033737 001124 001126        BIT      $GDDAT,$BDDAT      ;TEST IT
863 003724 001001                       BNE      TST16      ;;BR IF SET
864 003726 104001                       ERROR      1          ;ERROR, RELAY 1 FAILED TO SET
865
866                                     ;*****
867                                     ;*TEST 16      TEST RELAY #2 STATUS BIT
868                                     ;*****
869 003730 000004                       TST16: SCOPE
870 003732 012737 000400 001124        MOV      #BIT8,$GDDAT      ;LOAD EXPECT
871 003732 012737 000400 001124        MOV      #BIT8,QGRSTAT      ;SET RELAY 2
872 003740 012777 000400 175466        MOV      QGRSTAT,$BDDAT      ;READ STATUS
873 003746 017737 175462 001126        BIT      $GDDAT,$BDDAT
874 003754 033737 001124 001126        BNE      TST17      ;;BR IF SET
875 003762 001001                       ERROR      1          ;ERROR, RELAY 2 FAILED TO SET
876 003764 104001
877
878                                     ;*****
879                                     ;*TEST 17      TEST OUTPUT DATA ACCEPT FLAG
880                                     ;*****
881 003766 000004                       TST17: SCOPE
882 003770 012737 100000 001124        MOV      #BIT15,$GDDAT      ;LOAD EXPECTED
883 003776 012777 100000 175430        MOV      #BIT15,QGRSTAT      ;SET BIT 15
884 004004 017737 175424 001126        MOV      QGRSTAT,$BDDAT      ;READ STATUS
885 004012 033737 001124 001126        BIT      $GDDAT,$BDDAT
886 004020 001001                       BNE      TST20      ;;BR IF SET
887 004022 104001                       ERROR      1          ;ERROR, BIT 15 FAILED TO SET
888
889                                     ;*****
890                                     ;*TEST 20      TEST OUTPUT INTERRUPT ENABLE
891                                     ;*****
892 004024 000004                       TST20: SCOPE
893 004026 005077 175402                CLR      QGRSTAT      ;CLEAR STATUS
894 004032 012737 040000 001124        MOV      #BIT14,$GDDAT      ;LOAD EXPECTED
895 004040 012777 040000 175366        MOV      #BIT14,QGRSTAT      ;LOAD BIT 14
896 004046 017737 175362 001126        MOV      QGRSTAT,$BDDAT      ;READ STATUS
897 004054 033737 001124 001126        BIT      $GDDAT,$BDDAT
898 004062 001001                       BNE      TST21      ;;BR IF SET
899 004064 104001                       ERROR      1          ;ERROR BIT 14 FAILED TO SET
900

```



```

901
902
903      ;*****
904      ;*TEST 21      TEST INPUT DATA READY FLAG
905      ;*****
906      TST21: SCOPE
907      MOV      #BIT7,$GDDAT      ;LOAD EXPECTED
908      MOV      #BIT7,$GRSTAT      ;SET BIT 7
909      MOV      $GRSTAT,$BDDAT      ;READ RESULT
910      CMP      $GDDAT,$BDDAT      ;COMPARE
911      BEQ      TST22      ;;BR IF EQUAL
912      ERROR    1      ;ERROR, BIT 7 FAILED TO SET
913
914      ;*****
915      ;*TEST 22      TEST INPUT INTERRUPT ENABLE
916      ;*****
917      TST22: SCOPE
918      CLR      $GRSTAT      ;CLEAR STATUS
919      MOV      #BIT6,$GDDAT      ;LOAD EXPECTED
920      MOV      #BIT6,$GRSTAT      ;SET BIT 6
921      MOV      $GRSTAT,$BDDAT      ;READ RESULT
922      CMP      $GDDAT,$BDDAT      ;COMPARE
923      BEQ      TST23      ;;BR IF EQUAL
924      ERROR    1      ;ERROR, BIT 6 FAILED TO SET
925
926      ;*****
927      ;*TEST 23      TEST THAT RESET CLEARS THE DIGITAL STATUS REGISTER
928      ;*****
929      TST23: SCOPE
930      MOV      #40,$TIMES      ;;DO 40 ITERATIONS
931      CLR      $GRSTAT      ;CLEAR STATUS
932      CLR      $GDDAT      ;LOAD EXPECTED
933      MOV      #BIT15!BIT14!BIT8!BIT7!BIT6!BIT0,$GRSTAT
934      RESET
935      MOV      $GRSTAT,$BDDAT      ;READ RESULT
936      BEQ      TST24      ;;BR IF EQUAL
937      ERROR    1      ;ERROR, RESET FAILED TO CLEAR DIGITAL
938      ;STATUS REGISTER

```

K02

MAINDEC-11-DZLPI-B
DZLP1B.P11 T23

LPS-11-DRA DIAGNOSTIC

MACY11 27(732)

17-SEP-76 14:21 PAGE 23

TEST THAT RESET CLEARS THE DIGITAL STATUS REGISTER

```

939
940
941
942
943 004230 000004
944 004232 032777 002000 174676
945 004240 001402
946 004242 000137 006156
947 004246
948 004246 005077 175166
949 004252 012777 177777 175156
950 004260 005037 001124
951 004264 012777 000000 175146
952 004272 017737 175140 001126
953 004300 023737 001124 001126
954 004306 001401
955 004310 104002
956
957
958
959
960 004312 000004
961 004314 005077 175120
962 004320 012777 177777 175110
963 004326 012737 177777 001124
964 004334 012777 177777 175076
965 004342 017737 175070 001126
966 004350 023737 001124 001126
967 004356 001401
968 004360 104002
969
970
971
972
973
974 004362 000004
975 004364 005077 175050
976 004370 012777 177777 175040
977 004376 012737 052525 001124
978 004404 012777 052525 175026
979 004412 017737 175020 001126
980 004420 023737 001124 001126
981 004426 001401
982 004430 104002
983

;*****
;*TEST 24 TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
;*****
TST24: SCOPE
        BIT      #BIT10,ASWR      ;TEST SWITCH BIT
        BEQ      1$              ;BRANCH IF DOWN
        JMP      DRT21            ;BYPASS SOME TEST USING THE EXTERNAL CABLE
1$:      CLR      @GRDIO          ;CLEAR OUTPUT REGISTER
        MOV      #-1,@GRDAI       ;CLEAR INPUT
        CLR      $GDDAT          ;CLEAR EXPECTED
        MOV      #0,@GRDIO        ;LOAD THE OUTPUT
        MOV      @GRDAI,$BDDAT    ;READ THE INPUT
        CMP      $GDDAT,$BDDAT    ;COMPARE
        BEQ      TST25            ;;BR IF EQUAL
        ERROR    2                ;ERROR, INPUT DID NOT EQUAL THE OUTPUT REG.

;*****
;*TEST 25 TEST INPUT WITH #-1
;*****
TST25:  SCOPE
        CLR      @GRDIO          ;CLEAR OUTPUT REGISTER
        MOV      #-1,@GRDAI       ;CLEAR INPUT
        MOV      #-1,$GDDAT       ;LOAD EXPECTED
        MOV      #-1,@GRDIO       ;LOAD THE OUTPUT
        MOV      @GRDAI,$BDDAT    ;READ INPUT
        CMP      $GDDAT,$BDDAT    ;COMPARE
        BEQ      TST26            ;;BR IF EQUAL
        ERROR    2                ;ERROR, INPUT DID NOT EQUAL THE OUTPUT
                                   ;IS WRAP-AROUND CABLE CONNECTED ???

;*****
;*TEST 26 TEST INPUT WITH #52525
;*****
TST26:  SCOPE
        CLR      @GRDIO          ;CLEAR OUTPUT REGISTER
        MOV      #-1,@GRDAI       ;CLEAR INPUT
        MOV      #52525,$GDDAT    ;LOAD EXPECTED
        MOV      #52525,@GRDIO    ;LOAD THE OUTPUT
        MOV      @GRDAI,$BDDAT    ;READ INPUT
        CMP      $GDDAT,$BDDAT    ;COMPARE
        BEQ      TST27            ;;BR IF EQUAL
        ERROR    2                ;ERROR, INPUT DID NOT EQUAL OUTPUT

```



```

984
985
986
987
988 004432 000004
989 004434 005077 175000
990 004440 012777 177777 174770
991 004446 012737 125252 001124
992 004454 012777 125252 174756
993 004462 017737 174750 001126
994 004470 023737 001124 001126
995 004476 001401
996 004500 104002
997
998
999
1000
1001 004502 000004
1002 004504 012737 004532 001110
1003
1004 004512 012737 000001 001464
1005 004520 013737 001444 001466
1006 004526 005137 001466
1007 004532 013737 001464 001124 2$:
1008 004540 033737 001124 001444
1009 004546 001436
1010
1011 004550 013777 001124 174662
1012 004556 017737 174654 001126
1013 004564 043737 001466 001126
1014 004572 023737 001124 001126
1015 004600 001401
1016 004602 104002
1017
1018
1019
1020
1021 004604 043777 001124 174626 3$:
1022 004612 017737 174620 001126
1023 004620 043737 001466 001126
1024 004626 005037 001124
1025 004632 033737 001464 001126
1026 004640 001401
1027 004642 104002
1028
1029
1030 004644 006337 001464 1$:
1031 004650 001330
1032

```

```

*****
*TEST 27 TEST INPUT WITH #125252
*****
TST27: SCOPE
CLR JGRDIO ;CLEAR OUTPUT REGISTER
MOV #-1,JGRDAI ;CLEAR INPUT
MOV #125252,$GDDAT ;LOAD EXPECTED
MOV #125252,JGRDIO ;LOAD THE OUTPUT
MOV JGRDAI,$BDDAT ;READ INPUT
CMP $GDDAT,$BDDAT ;COMPARE
BEQ TST30 ;;BR IF EQUAL
ERROR 2 ;ERROR, INPUT DID NOT EQUAL OUTPUT

*****
*TEST 30 FLOAT A 1 ACROSS NON-LATCHING INPUT BITS
*****
TST30: SCOPE
MOV #2,$LPERR ;LOAD ERROR SCOPE RETURN
MOV #BIT0,BRLEV2 ;LOAD EXPECTED
MOV NOTLCH,BRLEV3 ;GET NON-LATCH
COM BRLEV3 ;COMPLEMENT
MOV BRLEV2,$GDDAT ;LOAD GOOD
BIT $GDDAT,NOTLCH ;TEST FOR NON-LATCH
BEQ 1$ ;BR IF LATCHING

MOV $GDDAT,JGRDIO ;LOAD OUTPUT
MOV JGRDAI,$BDDAT ;READ INPUT
BIC BRLEV3,$BDDAT ;MASK TO LATCH BITS
CMP $GDDAT,$BDDAT ;COMPARE
BEQ 3$ ;;BR IF EQUAL
ERROR 2 ;INPUT REGISTER IN ERROR
;SUB-TEST CLEAR THE OUTPUT BIT AND TEST THE INPUT DOES NOT LATCH
;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

3$: BIC $GDDAT,JGRDIO ;CLEAR OUTPUT BIT
MOV JGRDAI,$BDDAT ;READ INPUT
BIC BRLEV3,$BDDAT ;MASK TO LATCH BITS
CLR $GDDAT ;CLEAR EXPECTED
BIT BRLEV2,$BDDAT ;TEST FOR BIT
BEQ 1$ ;;BR IF CLEARED
ERROR 2 ;INPUT BIT LATCHED IN ERROR
;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

1$: ASL BRLEV2 ;CHANGE PATTERN
BNE 2$ ;BR UNTIL DONE

```

M02

MAINDEC-11-DZLPI-B
DZLPB.P11 T30LPS-11-DRA DIAGNOSTIC
FLOAT A 1 ACROSS NON-LATCHING INPUT BITS

MACY11 27(732) 17-SEP-76 14:21 PAGE 25

```

1033
1034
1035      ;*****
1036      ;*TEST 31      FLOAT A 1 ACROSS LATCHING INPUT BITS
1037      ;*****
1037      TST31: SCOPE
1038      004652 000004      MOV      #2$, $LPERR      ;LOAD ERROR SCOPE RETURN
1039      004654 012737 004670 001110      MOV      #BIT0, $GDDAT      ;LOAD EXPECTED VALUE
1040      004662 012737 000001 001124
1041      004670 033737 001124 001444 2$: BIT      $GDDAT, NOTLCH      ;TEST FOR NON-LATCHING
1042      004676 001022      BNE      1$      ;BR IF NON-LATCH
1043
1044      004700 005077 174534      CLR      @GRDIO      ;CLEAR OUTPUT REGISTER
1045      004704 012777 177777 174524      MOV      #-1, @GRDAI      ;CLEAR INPUT
1046
1047      004712 013777 001124 174520      MOV      $GDDAT, @GRDIO      ;LOAD OUTPUT
1048      004720 005077 174514      CLR      @GRDIO      ;CLEAR OUTPUT REGISTER
1049      004724 017737 174506 001126      MOV      @GRDAI, $BDDAT      ;READ INPUT REG.
1050      004732 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE
1051      004740 001401      BEQ      1$      ;;BR IF EQUAL
1052      004742 104002      ERROR      2      ;INPUT REGISTER FAILED TO LATCH DATA
1053      ;IS THIS REALLY A "LPS-11-DRA" ?? FIRST ERROR
1054      ;IF RUNNING ON A "LPS-11-DR" IF "DR" USE MD-11-DZLPD
1055
1056      004744 006337 001124      1$: ASL      $GDDAT      ;CHANGE PATTERN
1057      004750 001347      BNE      2$      ;BR UNTIL DONE
1058
1059      ;*****
1060      ;*TEST 32      FLOAT A 0 ACROSS LATCHING INPUT BITS
1061      ;*****
1062      TST32: SCOPE
1063      004752 000004      MOV      #2$, $LPERR      ;LOAD ERROR SCOPE RETURN
1064      004754 012737 004770 001110      MOV      #BIT0, BRLEV3      ;LOAD EXPECTED
1065      004762 012737 000001 001466
1066      004770 033737 001466 001444 2$: BIT      BRLEV3, NOTLCH      ;TEST FOR LATCHING
1067      004776 001035      BNE      1$      ;BR IF NOT
1068
1069      005000 005077 174434      CLR      @GRDIO      ;CLEAR OUTPUT REGISTER
1070      005004 012777 177777 174424      MOV      #-1, @GRDAI      ;CLEAR INPUT
1071      005012 012737 177777 001124      MOV      #-1, $GDDAT      ;LOAD
1072      005020 043737 001444 001124      BIC      NOTLCH, $GDDAT
1073      005026 000240      NOP
1074      005030 000240      NOP
1075      005032 043737 001466 001124      BIC      BRLEV3, $GDDAT      ;MAKE BRLEV3
1076      005040 013777 001124 174372      MOV      $GDDAT, @GRDIO      ;LOAD OUTPUT
1077      005046 005077 174366      CLR      @GRDIO      ;CLEAR OUTPUT REGISTER
1078
1079      005052 017737 174360 001126      MOV      @GRDAI, $BDDAT      ;READ INPUT
1080      005060 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE
1081      005066 001401      BEQ      1$      ;;BR IF EQUAL
1082      005070 104002      ERROR      2      ;INPUT REGISTER FAILED TO LATCH DATA
1083
1084      005072 006337 001466      1$: ASL      BRLEV3      ;CHANGE PATTERN
1085      005076 001334      BNE      2$      ;BR UNTIL DONE
1086

```


1087								:*****
1088								:*****
1089								:TEST 33 TEST FOR SLOW INPUT GATES WITH #125252
1090								:*****
1091	005100	000004						TST33: SCOPE
1092								
1093	005102	012737	125252	001124	MOV	#125252,\$GDDAT	;LOAD EXPECTED	
1094	005110	043737	001444	001124	BIC	NOTLCH,\$GDDAT	;CONVERT	
1095	005116	013700	001124		MOV	\$GDDAT,R0	;LOAD PATTERN	
1096	005122	005077	174312		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1097	005126	012777	177777	174302	MOV	#-1,@GRDAI	;CLEAR INPUT	
1098								
1099	005134	010077	174300		MOV	R0,@GRDIO	;LOAD OUTPUT	
1100	005140	005077	174274		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1101	005144	017701	174266		MOV	@GRDAI,R1	;READ INPUT	
1102	005150	050177	174262		BIS	R1,@GRDAI	;CLEAR INPUT	
1103	005154	005100			COM	R0		
1104	005156	010077	174256		MOV	R0,@GRDIO	;LOAD OUTPUT	
1105	005162	005077	174252		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1106	005166	017701	174244		MOV	@GRDAI,R1	;READ INPUT	
1107	005172	050177	174240		BIS	R1,@GRDAI	;CLEAR INPUT	
1108	005176	005100			COM	R0		
1109	005200	010077	174234		MOV	R0,@GRDIO	;LOAD OUTPUT	
1110	005204	005077	174230		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1111	005210	017701	174222		MOV	@GRDAI,R1	;READ INPUT	
1112	005214	050177	174216		BIS	R1,@GRDAI	;CLEAR INPUT	
1113	005220	005100			COM	R0		
1114	005222	010077	174212		MOV	R0,@GRDIO	;LOAD OUTPUT	
1115	005226	005077	174206		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1116	005232	017701	174200		MOV	@GRDAI,R1	;READ INPUT	
1117	005236	050177	174174		BIS	R1,@GRDAI	;CLEAR INPUT	
1118	005242	005100			COM	R0		
1119	005244	010077	174170		MOV	R0,@GRDIO	;LOAD OUTPUT	
1120	005250	005077	174164		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1121	005254	017701	174156		MOV	@GRDAI,R1	;READ INPUT	
1122	005260	050177	174152		BIS	R1,@GRDAI	;CLEAR INPUT	
1123	005264	005100			COM	R0		
1124	005266	010077	174146		MOV	R0,@GRDIO	;LOAD OUTPUT	
1125	005272	005077	174142		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1126	005276	017701	174134		MOV	@GRDAI,R1	;READ INPUT	
1127	005302	050177	174130		BIS	R1,@GRDAI	;CLEAR INPUT	
1128	005306	005100			COM	R0		
1129	005310	010077	174124		MOV	R0,@GRDIO	;LOAD OUTPUT	
1130	005314	005077	174120		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1131	005320	017701	174112		MOV	@GRDAI,R1	;READ INPUT	
1132	005324	050177	174106		BIS	R1,@GRDAI	;CLEAR INPUT	
1133	005330	005100			COM	R0		
1134	005332	010077	174102		MOV	R0,@GRDIO	;LOAD OUTPUT	
1135	005336	005077	174076		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1136	005342	017701	174070		MOV	@GRDAI,R1	;READ INPUT	
1137	005346	050177	174064		BIS	R1,@GRDAI	;CLEAR INPUT	
1138	005352	005100			COM	R0		
1139	005354	010077	174060		MOV	R0,@GRDIO	;LOAD OUTPUT	
1140	005360	005077	174054		CLR	@GRDIO	;CLEAR OUTPUT REGISTER	
1141	005364	017701	174046		MOV	@GRDAI,R1	;READ INPUT	
1142	005370	050177	174042		BIS	R1,@GRDAI	;CLEAR INPUT	

MAINDEC-11-DZLPI-8
 DZLPIB.P11 T33

 LPS-11-DRA DIAGNOSTIC
 TEST FOR SLOW INPUT GATES WITH #125252

MACY11 27(732) 17-SEP-76 14:21 PAGE 27

1143	005374	005100		COM	RO	
1144	005376	010077	174036	MOV	RO, JGRDIO	:LOAD OUTPUT
1145	005402	005077	174032	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1146	005406	017701	174024	MOV	JGRDAI, R1	:READ INPUT
1147	005412	050177	174020	BIS	R1, JGRDAI	:CLEAR INPUT
1148	005416	005100		COM	RO	
1149	005420	010077	174014	MOV	RO, JGRDIO	:LOAD OUTPUT
1150	005424	005077	174010	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1151	005430	017701	174002	MOV	JGRDAI, R1	:READ INPUT
1152	005434	050177	173776	BIS	R1, JGRDAI	:CLEAR INPUT
1153	005440	005100		COM	RO	
1154						
1155	005442	010137	001126	MOV	R1, \$BDDAT	:LOAD READ
1156	005446	023737	001124 001126	CMP	\$GDDAT, \$BDDAT	:COMPARE
1157	005454	001401		BEO	TST34	::BR IF EQUAL
1158	005456	104002		ERROR	2	:INPUT GATE SLOW

 :TEST 34 TEST FOR SLOW INPUT GATES WITH #52525

 TST34: SCOPE

1160						
1161						
1162						
1163	005460	000004				
1164						
1165	005462	012737	052525 001124	MOV	#52525, \$GDDAT	:SETUP EXPECTED
1166	005470	043737	001444 001124	BIC	NOTLCH, \$GDDAT	:CONVERT
1167	005476	013700	001124	MOV	\$GDDAT, RO	
1168	005502	005077	173732	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1169	005506	012777	177777 173722	MOV	#-1, JGRDAI	:CLEAR INPUT
1170						
1171	005514	010077	173720	MOV	RO, JGRDIO	:LOAD OUTPUT
1172	005520	005077	173714	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1173	005524	017701	173706	MOV	JGRDAI, R1	:READ INPUT
1174	005530	050177	173702	BIS	R1, JGRDAI	:CLEAR INPUT
1175	005534	005100		COM	RO	
1176	005536	010077	173676	MOV	RO, JGRDIO	:LOAD OUTPUT
1177	005542	005077	173672	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1178	005546	017701	173664	MOV	JGRDAI, R1	:READ INPUT
1179	005552	050177	173660	BIS	R1, JGRDAI	:CLEAR INPUT
1180	005556	005100		COM	RO	
1181	005560	010077	173654	MOV	RO, JGRDIO	:LOAD OUTPUT
1182	005564	005077	173650	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1183	005570	017701	173642	MOV	JGRDAI, R1	:READ INPUT
1184	005574	050177	173636	BIS	R1, JGRDAI	:CLEAR INPUT
1185	005600	005100		COM	RO	
1186	005602	010077	173632	MOV	RO, JGRDIO	:LOAD OUTPUT
1187	005606	005077	173626	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1188	005612	017701	173620	MOV	JGRDAI, R1	:READ INPUT
1189	005616	050177	173614	BIS	R1, JGRDAI	:CLEAR INPUT
1190	005622	005100		COM	RO	
1191	005624	010077	173610	MOV	RO, JGRDIO	:LOAD OUTPUT
1192	005630	005077	173604	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1193	005634	017701	173576	MOV	JGRDAI, R1	:READ INPUT
1194	005640	050177	173572	BIS	R1, JGRDAI	:CLEAR INPUT
1195	005644	005100		COM	RO	
1196	005646	010077	173566	MOV	RO, JGRDIO	:LOAD OUTPUT
1197	005652	005077	173562	CLR	JGRDIO	:CLEAR OUTPUT REGISTER
1198	005656	017701	173554	MOV	JGRDAI, R1	:READ INPUT

MAINDEC-11-DZLPI-B
 DZLPIB.P11 T34

 LPS-11-DRA DIAGNOSTIC
 TEST FOR SLOW INPUT GATES WITH #52525

MACY11 27(732) 17-SEP-76 14:21 PAGE 28

1199	005662	050177	173550	BIS	R1, @GRDAI	:CLEAR INPUT
1200	005666	005100		COM	RO	
1201	005670	010077	173544	MOV	RO, @GRDIO	:LOAD OUTPUT
1202	005674	005077	173540	CLR	@GRDIO	:CLEAR OUTPUT REGISTER
1203	005700	017701	173532	MOV	@GRDAI, R1	:READ INPUT
1204	005704	050177	173526	BIS	R1, @GRDAI	:CLEAR INPUT
1205	005710	005100		COM	RO	
1206	005712	010077	173522	MOV	RO, @GRDIO	:LOAD OUTPUT
1207	005716	005077	173516	CLR	@GRDIO	:CLEAR OUTPUT REGISTER
1208	005722	017701	173510	MOV	@GRDAI, R1	:READ INPUT
1209	005726	050177	173504	BIS	R1, @GRDAI	:CLEAR INPUT
1210	005732	005100		COM	RO	
1211	005734	010077	173500	MOV	RO, @GRDIO	:LOAD OUTPUT
1212	005740	005077	173474	CLR	@GRDIO	:CLEAR OUTPUT REGISTER
1213	005744	017701	173466	MOV	@GRDAI, R1	:READ INPUT
1214	005750	050177	173462	BIS	R1, @GRDAI	:CLEAR INPUT
1215	005754	005100		COM	RO	
1216	005756	010077	173456	MOV	RO, @GRDIO	:LOAD OUTPUT
1217	005762	005077	173452	CLR	@GRDIO	:CLEAR OUTPUT REGISTER
1218	005766	017701	173444	MOV	@GRDAI, R1	:READ INPUT
1219	005772	050177	173440	BIS	R1, @GRDAI	:CLEAR INPUT
1220	005776	005100		COM	RO	
1221	006000	010077	173434	MOV	RO, @GRDIO	:LOAD OUTPUT
1222	006004	005077	173430	CLR	@GRDIO	:CLEAR OUTPUT REGISTER
1223	006010	017701	173422	MOV	@GRDAI, R1	:READ INPUT
1224	006014	050177	173416	BIS	R1, @GRDAI	:CLEAR INPUT
1225	006020	005100		COM	RO	
1226						
1227	006022	010137	001126	MOV	R1, \$BDDAT	:LOAD VALUE READ
1228	006026	023737	001124 001126	CMP	\$GDDAT, \$BDDAT	:COMPARE
1229	006034	001401		BEQ	TST35	::BR IF EQUAL
1230	006036	104002		ERROR	2	

E03

MAINDEC-11-DZLPI-B
DZLPIS.P11 T36

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 30

TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET

```

1260      :TEST THAT THE DIGITAL I/O DOES NOT INTERRUPT
1261      :SET INPUT-OUTPUT FLAGS BUT DO NOT ENABLE INTERRUPTS
1262
1263      006156
1264
1265      DRT21:
1266      :*****
1267      :*TEST 37      TEST FOR UNEXPECTED INTERRUPT
1268      :*****
1269      TST37:  SCOPE
1270      MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1271      RESET
1272      CLR     @GRSTAT
1273      CLR     PSW
1274      MOV     #1$,@GRIVA      ;SET UP INTERRUPT INPUT VECTOR
1275      MOV     #340,@GRIVSA
1276      MOV     #2$,@GRIVB      ;SET UP INTERRUPT OUTPUT VECTOR
1277      MOV     #340,@GRIVSB
1278      MOV     #0,@GRDIO      ;OUTPUT
1279      MOV     @GRDAI,R0      ;INPUT
1280      NOP
1281      NOP
1282      NOP
1283      CLR     @GRSTAT      ;CLEAR STATUS
1284      BR      TST40      ;;
1285      1$:  ERROR 6      ;ERROR, DIGITAL INPUT INTERRUPTED
1286      RTI
1287
1288      2$:  ERROR 6      ;ERROR, DIGITAL OUTPUT INTERRUPTED
1289      RTI

```

```

1290
1291
1292
1293
1294 006270 000004
1295 006272 000240
1296 006274 000240
1297 006276 005077 173132
1298 006302 012777 006344 173140
1299 006310 012777 000340 173134
1300 006316 013777 001456 173130
1301 006324 012777 000000 173124
1302 006332 052777 000040 173074
1303 006340 000240
1304 006342 104004
1305
1306
1307
1308
1309 006344 012777 006366 173076 1$: MOV #2$, @GRIVA ;LOAD INPUT VECTOR
1310 006352 022626 CMP (SP)+, (SP)+ ;POP THE STACK *4
1311 006354 005037 177776 CLR PSW ;LOWER PRIOR.
1312 006360 000240
1313 006362 000240
1314 006364 000404 BR TST41 ;; <NEXT TEST>
1315
1316 006366 022626 2$: CMP (SP)+, (SP)+ ;POP THE STACK
1317 006370 104006 ERROR 6 ;UNEXPECTED INTERRUPT
1318 006372 005037 177776 CLR PSW

```

;*****
 ;*TEST 40 TEST THAT THE INPUT CAN INT. USING THE MAINT. BIT
 ;*****
 TST40: SCOPE
 NOP
 NOP
 CLR @GRSTAT ;CLEAR STATUS
 MOV #1\$, @GRIVA ;LOAD RETURN VECTOR
 MOV #340, @GRIVSA
 MOV GRIVSB, @GRIVB ;LOAD OUTPUT VECTOR
 MOV #0, @GRIVSB
 BIS #BITS, @GRSTAT ;MAINT. INT C
 NOP
 ERROR 4 ;ERROR, INPUT FAILED TO INTERRUPT

;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN NO INTERRUPT WILL OCCUR
 ; IF INTERRUPT OCCURS 'INT DONE C' FAILED TO CLEAR INT C FLOP


```

1319
1320
1321
1322
1323 006376 000004
1324 006400 000240
1325 006402 000240
1326 006404 005077 173024
1327 006410 012777 006460 173032
1328 006416 012777 000340 173026
1329 006424 013777 001456 173022
1330 006432 012777 000000 173016
1331 006440 012777 000100 172766
1332 006446 052777 000040 172760
1333 006454 000240
1334 006456 104004
1335
1336 006460 012777 006522 172762 1S:
1337 006466 022626
1338 006470 005037 177776
1339 006474 005037 001124
1340 006500 017737 172730 001126
1341 006506 032737 000100 001126
1342 006514 001406
1343 006516 104001
1344 006520 000404
1345
1346 006522 022626
1347 006524 104006
1348 006526 005037 177776

;*****
;TEST 41 TEST THAT THE INPUT INT. CLEARS INT. ENABLE VIA MAINT. BIT
;*****
TST41: SCOPE
NOP
NOP
CLR @GRSTAT ;CLEAR STATUS
MOV #1S,@GRIVA ;LOAD RETURN VECTOR
MOV #340,@GRIVSA
MOV GRIVSB,@GRIVB ;LOAD OUTPUT VECTOR
MOV #0,@GRIVSB
MOV #BIT6,@GRSTAT ;LOAD INPUT INT. ENABLE
BIS #BIT5,@GRSTAT ;MAINT. INT C
NOP
ERROR 4 ;ERROR, INPUT FAILED TO INTERRUPT
1336 006460 012777 006522 172762 1S: MOV #2S,@GRIVA ;LOAD INPUT VECTOR
CMP (SP)+,(SP)+ ;POP THE STACK #4
CLR PSW ;LOWER PRIOR.
CLR $GDDAT ;CLEAR EXPECTED
MOV @GRSTAT,$BDDAT ;READ STATUS
BIT #BIT6,$BDDAT ;TEST BIT 6
BEQ TST42 ;;BR IF CLEARED
ERROR 1 ;INPUT INT. FAILED TO CLEAR
BR TST42 ;;<NEXT TEST>

2S: CMP (SP)+,(SP)+ ;POP THE STACK
ERROR 6 ;UNEXPECTED INTERRUPT
CLR PSW

```

H03

MAINDEC-11-DZLPI-B
DZLP1B.P11 T41

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 33

TEST THAT THE INPUT INT. CLEARS INT. ENABLE VIA MAINT. BIT

1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378

006532	000004		
006534	000240		
006536	005077	172672	
006542	013777	001452	172700
006550	012777	000000	172674
006556	012777	006604	172670
006564	012777	000340	172664
006572	052777	020000	172634
006600	000240		
006602	104005		
006604	012777	006626	172642
006612	022626		
006614	005037	177776	
006620	000240		
006622	000240		
006624	000404		
006626	022626		
006630	104006		
006632	005037	177776	

```

*****
:TEST 42      TEST THAT THE OUTPUT CAN INT. USING THE MAINT. BIT
*****
TST42:  SCOPE
        NOP
        CLR      @GRSTAT          ;CLEAR STATUS
        MOV      @RIVSA,@GRIVA    ;LOAD INPUT VECTOR
        MOV      #0,@GRIVSA
        MOV      #1$,@GRIVB       ;LOAD OUTPUT VECTOR
        MOV      #340,@GRIVSB
        BIS      #BIT13,@GRSTAT   ;MAINT. INTERRUPT
        NOP
        ERROR    5                ;OUTPUT FAILED TO INTERRUPT

;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN, NO INTERRUPT WILL OCCUR
; IF IT DOES, 'INT DONE D HIGH' FAILED TO CLEAR 'INT D' FLOP

1$:     MOV      #2$,@GRIVB        ;LOAD OUTPUT VECTOR
        CMP      (SP)+,(SP)+      ;POP THE STACK *4
        CLR      PSW
        NOP
        NOP
        BR       TST43            ;;<NEXT TEST>
2$:     CMP      (SP)+,(SP)+      ;POP THE STACK
        ERROR    6                ;UNEXPECTED INTERRUPT
        CLR      PSW

```


MAINDEC-11-DZLPI-B
DZLP1B.P11 T42

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 34

TEST THAT THE OUTPUT CAN INT. USING THE MAINT. BIT

```

1379
1380
1381
1382
1383 006636 000004
1384 006640 000240
1385 006642 000240
1386 006644 032777 002000 172264
1387 006652 001401
1388 006654 000455
1389 006656 005077 172552
1390 006662 012777 006752 172560
1391 006670 012777 000340 172554
1392 006676 013777 001456 172550
1393 006704 005077 172546
1394 006710 012777 000100 172516
1395 006716 012777 000000 172514
1396 006724 017700 172506
1397 006730 000240
1398 006732 000240
1399 006734 000240
1400 006736 042777 000100 172470
1401 006744 000240
1402 006746 104004
1403 006750 000417
1404 006752 022626
1405 006754 005037 177776
1406 006760 005077 172450
1407 006764 013777 001452 172456
1408 006772 005077 172454
1409 006776 013777 001456 172450
1410 007004 005077 172446
1411

*****
: *TEST 43 TEST FOR INT. FROM DRA INPUT
*****
TST43: SCOPE
NOP
NOP
BIT #BIT10, @SWR ;TEST SWITCH
BEQ 1$
BR TST44 ;;
1$: CLR @GRSTAT
MOV #2$, @GRIVA ;IN CASE OF INTERRUPT
MOV #340, @GRIVSA
MOV GRIVSB, @GRIVB
CLR @GRIVSB
MOV #BIT6, @GRSTAT ;ENABLE INPUT INT.
MOV #0, @GRADIO ;OUTPUT
MOV @GRDAI, R0 ;INPUT
NOP ;LET INTERRUPT OCCUR
NOP
NOP
BIC #BIT6, @GRSTAT
NOP
ERROR 4
BR TST44 ;;
2$: CMP (SP)+, (SP)+
CLR PSW
CLR @GRSTAT ;CLEAR STATUS
MOV GRIVSA, @GRIVA ;RESET INPUT VECTOR
CLR @GRIVSA
MOV GRIVSB, @GRIVB ;RESET OUTPUT VECTOR
CLR @GRIVSB

```

J03

MAINDEC-11-DZLPI-B
DZLPIS.P11 T44

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 35
TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG

```

1412
1413
1414
1415 007010 000004
1416 007012 032777 002000 172116
1417 007020 001061
1418
1419 007022 012737 007036 001110
1420 007030 012737 000001 001466
1421 007036 005037 001126
1422 007042 012737 000200 001124
1423
1424 007050 033737 001466 001446
1425 007056 001037
1426 007060 005077 172354
1427 007064 012777 177777 172344
1428 007072 005077 172336
1429 007076 013777 001466 172334
1430 007104 005077 172324
1431
1432 007110 105777 172320
1433 007114 100401
1434 007116 104010
1435
1436
1437
1438 007120 043777 001466 172312 2$:
1439 007126 053777 001466 172302
1440 007134 005037 001124
1441 007140 005077 172270
1442 007144 117737 172264 001126
1443 007152 100001
1444 007154 104001
1445
1446 007156 006337 001466 3$:
1447 007162 001325

```

```

*****
*TEST 44      TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG
*****
TST44:  SCOPE
        BIT      #BIT10,JSWR          ;TEST CABLE SWITCH
        BNE      TST45                ;;BYPASS IF NO I/O CABLE

        MOV      #1$,SLPERR           ;LOAD ERROR SCOPE RETURN
        MOV      #BIT0,BRLEV3         ;LOAD INTERRUPT BIT
1$:     CLR      $BDDAT                ;CLEAR BAD DATA
        MOV      #BIT7,$GDDAT         ;LOAD GOOD DATA

        BIT      BRLEV3,NOTINT        ;TEST IF SET TO INT
        BNE      3$                   ;;NO TRY NEXT BIT
        CLR      @GRDIO                ;CLEAR OUTPUT REGISTER
        MOV      #-1,@GRDAI            ;CLEAR INPUT
        CLR      @GRSTAT               ;CLEAR STATUS
        MOV      BRLEV3,@GRDIO         ;LOAD OUTPUT/INPUT
        CLR      @GRSTAT               ;CLEAR FLAG FROM DATA READY
        SHOULD REMAIN SET VIA DIRECT SET SIDE
        TSTB     @GRSTAT               ;TEST READY BIT
        BMI      2$                   ;;BR IF SET
        ERROR    10                   ;INPUT INTERRUPT BIT FAILED TO SET INPUT READY
        ;;BR IF SET
        ?? DID OPERATOR GIVE CORRECT
        INPUT INTERRUPT BITS ??

        BIC      BRLEV3,@GRDIO         ;CLEAR OUTPUT
        BIS      BRLEV3,@GRDAI         ;CLEAR INPUT
        CLR      $GDDAT                ;CLEAR EXPECTED
        CLR      @GRSTAT               ;CLEAR STATUS
        MOVB     @GRSTAT,$BDDAT        ;READ STATUS
        BPL      3$                   ;;BR IF CLEARED
        ERROR    1                    ;INPUT READY FAILED TO CLEAR

        ASL      BRLEV3                ;CHANGE BIT
        BNE      1$                   ;BR IF NOT DONE

```


K03

MAINDEC-11-DZLPI-B
DZLPIB.P11 T45

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 36

TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG

```

1448
1449
1450
1451 007164 000004
1452 007166 032777 002000 171742
1453 007174 001042
1454
1455 007176 012737 007212 001110
1456 007204 012737 000001 001466
1457 007212 012737 000200 001126 1$:
1458 007220 005037 001124
1459
1460 007224 033737 001466 001446
1461 007232 001420
1462 007234 005077 172200
1463 007240 012777 177777 172170
1464 007246 005077 172162
1465 007252 013777 001466 172160
1466 007260 005077 172150
1467
1468 007264 105777 172144
1469 007270 100001
1470 007272 104011
1471
1472
1473
1474
1475 007274 006337 001466 3$:
1476 007300 001344

```

 ;*TEST 45 TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG

 TST45: SCOPE
 BIT #BIT10, QSWR ;TEST CABLE SWITCH
 BNE TST46 ;;BYPASS IF NO I/O CABLE
 MOV #1\$, \$LPERR ;LOAD ERROR SCOPE RETURN
 MOV #BIT0, BRLEV3 ;LOAD NON-INTERRUPT BIT
 MOV #200, \$BDDAT ;LOAD BAD DATA
 CLR \$GDDAT ;CLEAR GOOD DATA
 BIT BRLEV3, NOTINT ;TEST IF SET TO INT
 BEQ 3\$;;NO SKIP AND TRY NEXT BIT
 CLR QGRDIO ;CLEAR OUTPUT REGISTER
 MOV #-1, QGRDAI ;CLEAR INPUT
 CLR QGRSTAT ;CLEAR STATUS
 MOV BRLEV3, QGRDIO ;LOAD OUTPUT/INPUT
 CLR QGRSTAT ;CLEAR FLAG FROM DATA READY
 ;SHOULD REMAIN SET VIA DIRECT SET SIDE
 ;TEST READY BIT
 TSTB QGRSTAT
 BPL 3\$;;BR IF CLEAR
 ERROR 11 ;INPUT NON-INTERRUPT BIT SET INPUT READY
 ;?? DID OPERATOR GIVE CORRECT
 ;INPUT INTERRUPT BITS ??
 ASL BRLEV3 ;CHANGE BIT
 BNE 1\$;BR IF NOT DONE

```

1477
1478
1479
1480
1481
1482 007302 000004
1483 007304 000240
1484 007306 000240
1485 007310 032777 002000 171620
1486 007316 001401
1487 007320 000455
1488 007322 005077 172106
1489 007326 012777 007416 172120
1490 007334 012777 000340 172114
1491 007342 013777 001452 172100
1492 007350 005077 172076
1493 007354 012777 040000 172052
1494 007362 012777 000000 172050
1495 007370 017700 172042
1496 007374 000240
1497 007376 000240
1498 007400 000240
1499 007402 042777 040000 172024
1500 007410 000240
1501 007412 104005
1502 007414 000417
1503 007416 022626
1504 007420 013777 001452 172022
1505 007426 005077 172020
1506 007432 013777 001456 172014
1507 007440 005077 172012
1508 007444 005037 177776
1509 007450 005077 171760
1510
1511
1512
1513
1514
1515 007454 000004
1516 007456 012737 000001 001164
1517 007464 000005
1518 007466 042737 177437 001460
1519 007474 001001
1520 007476 104007
1521 007500 022737 000340 001460
1522 007506 001001
1523 007510 104007
1524 007512 013737 001460 001462
1525 007520 162737 000040 001462
1526 007526 013737 001460 001464
1527 007534 013737 001460 001466
1528 007542 062737 000040 001466

;*****
;TEST 46 TEST FOR INT. FROM DRA OUTPUT
;*****
TST46: SCOPE
NOP
NOP
BIT #BIT10,DSWR ;TEST SWITCH
BEQ 1$
BR TST47 ;;
CLR DGRSTAT
MOV #2$,DGRIVB ;IN CASE OF INTERRUPT
MOV #340,DGRIVSB
MOV GRIVSA,DGRIVA
CLR DGRIVSA
MOV #BIT14,DGRSTAT ;ENABLE OUTPUT INT.
MOV #0,DGRDIO ;OUTPUT
MOV DGRDAI,R0 ;INPUT
NOP ;LET INTERRUPT OCCUR
NOP
NOP
BIC #BIT14,DGRSTAT
NOP
ERROR 5 ;ERROR, OUTPUT FAILED TO INTERRUPT
BR TST47 ;;
2$: CMP (SP)+,(SP)+
MOV GRIVSA,DGRIVA ;RESET INPUT VECTOR
CLR DGRIVSA
MOV GRIVSB,DGRIVB ;RESET OUTPUT VECTOR
CLR DGRIVSB
CLR PSW
CLR DGRSTAT ;CLEAR STATUS

;PRE INTERRUPT SETUP
;*****
;TEST 47 PRE-INTERRUPT SETUP
;*****
TST47: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
RESET
BIC #177437,DIOBRL
BNE 1$
ERROR 7 ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 0
1$: CMP #340,DIOBRL
BNE 2$
ERROR 7 ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 7
2$: MOV DIOBRL,BRLEV1
SUB #40,BRLEV1
MOV DIOBRL,BRLEV2
MOV DIOBRL,BRLEV3
ADD #40,BRLEV3

```


M03

MAINDEC-11-DZLPI-B
DZLPIB.P11 T50

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 38
TEST FOR INT. FROM DRA INPUT ON LEVEL INDICATED -1 VIA MAINT. INT

```

1529
1530
1531
1532 007550 000004
1533 007552 005077 171656
1534 007556 012777 007612 171664
1535 007564 013737 001462 177776
1536 007572 052777 000040 171634
1537 007600 000240
1538 007602 000240
1539 007604 000240
1540 007606 000240
1541 007610 104004
1542 007612 022626
1543 007614 013777 001452 171626
1544 007622 005077 171624
1545 007626 013777 001456 171620
1546 007634 005077 171616
1547 007640 005037 177776
1548
1549
1550
1551 007644 000004
1552 007646 005077 171562
1553 007652 012777 007724 171570
1554 007660 013737 001464 177776
1555 007666 052777 000040 171540
1556 007674 000240
1557 007676 000240
1558 007700 000240
1559
1560 007702 012777 007736 171540
1561 007710 005037 177776
1562 007714 000240
1563 007716 000240
1564 007720 000240
1565 007722 000403
1566 007724 022626
1567 007726 005037 177776
1568 007732 104006
1569 007734 000415
1570 007736 022626
1571 007740 013777 001452 171502
1572 007746 005077 171500
1573 007752 013777 001456 171474
1574 007760 005077 171472
1575 007764 005037 177776

;*****
;TEST 50 TEST FOR INT. FROM DRA INPUT ON LEVEL INDICATED -1 VIA MAINT. INT
;*****
TST50: SCOPE
CLR @GRSTAT ;CLEAR ENABLES
MOV #1$, @GRIVA ;SET UP VECTORS
MOV BRLEV1, PSW ;CHANGE PSW
BIS #BITS, @GRSTAT ;GENERATE INPUT MAINT. INTERRUPT
NOP
NOP
NOP
NOP
ERROR 4 ;ERROR, NO INTERRUPT FROM DIGITAL I/O INPUT
1$: CMP (SP)+, (SP)+
MOV GRIVSA, @GRIVA ;RESET INPUT VECTOR
CLR @GRIVSA
MOV GRIVSB, @GRIVB ;RESET OUTPUT VECTOR
CLR @GRIVSB
CLR PSW ;LOWER PSW
;*****
;TEST 51 TEST FOR NO INT. FROM DRA INPUT ON LEVEL INDICATED VIA MAINT. INT
;*****
TST51: SCOPE
CLR @GRSTAT ;CLEAR ENABLES
MOV #1$, @GRIVA ;SET UP VECTORS
MOV BRLEV2, PSW ;CHANGE PSW
BIS #BITS, @GRSTAT ;GENERATE INPUT MAINT. INTERRUPT
NOP
NOP
NOP
;SUB-TEST, NOW LOWER THE PSW AND ALLOW THE INTERRUPT
MOV #2$, @GRIVA ;RESET VECTOR
CLR PSW ;LOWER PSW
NOP
NOP
NOP
BR 3$ ;ERROR
1$: CMP (SP)+, (SP)+
CLR PSW
3$: ERROR 6 ;UNEXPECTED INTERRUPT
BR TST52
2$: CMP (SP)+, (SP)+ ;POP THE STACK
MOV GRIVSA, @GRIVA ;RESET INPUT VECTOR
CLR @GRIVSA
MOV GRIVSB, @GRIVB ;RESET OUTPUT VECTOR
CLR @GRIVSB
CLR PSW

```

NO3

MAINDEC-11-DZLPI-B
DZLPIB.P11 T51

LPS-11-DRA DIAGNOSTIC

MACY11 27(732) 17-SEP-76 14:21 PAGE 39

TEST FOR NO INT. FROM DRA INPUT ON LEVEL INDICATED VIA MAINT. INT

```

1576
1577
1578
1579
1580 007770 000004
1581 007772 012737 000001 001164
1582 010000 000005
1583 010002 013777 001452 171440
1584 010010 005077 171436
1585 010014 013777 001456 171432
1586 010022 005077 171430
1587 010026 005037 177776
1588 010032 000137 010164
1589
1590
1591
1592
1593
1594 010036 000004
1595 010040 012777 000001 171366
1596 010046 004737 010066
1597 010052 012777 000400 171354
1598 010060 004737 010066
1599 010064 000765
1600
1601 010066 012737 040000 010162
1602 010074
1603 010074 005077 171340
1604 010100 012777 125252 171332
1605 010106 017737 171324 010160
1606 010114 013777 010160 171314
1607 010122 005077 171312
1608 010126 012777 052525 171304
1609 010134 017737 171276 010160
1610 010142 013777 010160 171266
1611 010150 005337 010162
1612 010154 001347
1613 010156 000207
1614
1615 010160 000000
1616 010162 000000

;*****
;TEST 52      END OF THE PROGRAM
;*****
TST52:  SCOPE
        MOV     #1,$TIMES      ;;DO 1 ITERATION
        RESET
        MOV     GRIVSA,GRIVA    ;RESET INPUT VECTOR
        CLR     GRIVSA
        MOV     GRIVSB,GRIVB    ;RESET OUTPUT VECTOR
        CLR     GRIVSB
        CLR     PSW
        JMP     $EOP

;*****
;TEST 53      MISC. EXTERNAL LOGIC TEST
;*****
TST53:  SCOPE
EXTTST: MOV     #BIT0,GRSTAT      ;SET RELAY #1
        JSR     PC,FLIP          ;TOGGLE THE INPUT-OUTPUT REG
        MOV     #BIT8,GRSTAT      ;SET RELAY #2
        JSR     PC,FLIP          ;TOGGLE THE INPUT-OUTPUT REG
        BR      EXTST
FLIP:   MOV     #40000,EXTCNT      ;LOAD COUNT
1$:     CLR     GRDIO             ;CLEAR OUTPUT REGISTER
        MOV     #125252,GRDIO    ;LOAD OUTPUT
        MOV     GRDAI,EXTTMP      ;READ INPUT
        MOV     EXTTMP,GRDAI      ;CLEAR INPUT
        CLR     GRDIO             ;CLEAR OUTPUT REGISTER
        MOV     #52525,GRDIO     ;LOAD OUTPUT
        MOV     GRDAI,EXTTMP      ;READ INPUT
        MOV     EXTTMP,GRDAI      ;CLEAR INPUT
        DEC     EXTCNT            ;FINISHED COUNT
        BNE     1$
        RTS     PC               ;EXIT

EXTTMP: 0
EXTCNT: 0

```


1661						
1662	010312	052123	052101	051525	EM1:	.ASCIZ /STATUS REGISTER IN ERROR/
1663	010320	051040	043505	051511		
1664	010326	042524	020122	047111		
1665	010334	042440	051122	051117		
1666	010342	000				
1667	010343	111	050116	052125	EM2:	.ASCIZ /INPUT REGISTER IN ERROR/
1668	010350	051040	043505	051511		
1669	010356	042524	020122	047111		
1670	010364	042440	051122	051117		
1671	010372	000				
1672	010373	117	052125	052520	EM3:	.ASCIZ /OUTPUT REGISTER IN ERROR/
1673	010400	020124	042522	044507		
1674	010406	052123	051105	044440		
1675	010414	020116	051105	047522		
1676	010422	000122				
1677	010424	047111	052520	020124	EM4:	.ASCIZ /INPUT FAILED TO INTERRUPT/
1678	010432	040506	046111	042105		
1679	010440	052040	020117	047111		
1680	010446	042524	051122	050125		
1681	010454	000124				
1682	010456	052517	050124	052125	EM5:	.ASCIZ /OUTPUT FAILED TO INTERRUPT/
1683	010464	043040	044501	042514		
1684	010472	020104	047524	044440		
1685	010500	052116	051105	052522		
1686	010506	052120	000			
1687	010511	125	042516	050130	EM6:	.ASCIZ /UNEXPECTED INTERRUPT/
1688	010516	041505	042524	020104		
1689	010524	047111	042524	051122		
1690	010532	050125	000124			
1691	010536	050117	051105	052101	EM7:	.ASCIZ /OPERATOR INTERVENTION ERROR/
1692	010544	051117	044440	052116		
1693	010552	051105	042526	052116		
1694	010560	047511	020116	051105		
1695	010566	047522	000122			
1696	010572	047111	042524	051122	EM10:	.ASCIZ /INTERRUPT INPUT BIT FAILED TO SET INPUT READY FLAG/
1697	010600	050125	020124	047111		
1698	010606	052520	020124	044502		
1699	010614	020124	040506	046111		
1700	010622	042105	052040	020117		
1701	010630	042523	020124	047111		
1702	010636	052520	020124	042522		
1703	010644	042101	020131	046106		
1704	010652	043501	000			
1705	010655	116	047117	044455	EM11:	.ASCIZ /NON-INTERRUPT INPUT BIT SET INPUT READY FLAG/
1706	010662	052116	051105	052522		
1707	010670	052120	044440	050116		
1708	010676	052125	041040	052111		
1709	010704	051440	052105	044440		
1710	010712	050116	052125	051040		
1711	010720	040505	054504	043040		
1712	010726	040514	000107			

004

 MAINDEC-11-DZLPI-8 LPS-11-DRA DIAGNOSTIC
 DZLPIB.P11 END OF PASS ROUTINE

MACY11 27(732) 17-SEP-76 14:21 PAGE 42

```

1713
1714 010732 051105 050122 020103 DH1: .ASCIZ /ERRPC STATUS EXPECTED/
1715 010740 020040 052123 052101
1716 010746 051525 020040 054105
1717 010754 042520 052103 042105
1718 010762 000
1719 010763 105 051122 041520 DH2: .ASCIZ /ERRPC INPUT EXPECTED/
1720 010770 020040 044440 050116
1721 010776 052125 020040 042440
1722 011004 050130 041505 042524
1723 011012 000104
1724 011014 051105 050122 020103 DH3: .ASCIZ /ERRPC OUTPUT EXPECTED/
1725 011022 020040 052517 050124
1726 011030 052125 020040 054105
1727 011036 042520 052103 042105
1728 011044 000
1729 011045 105 051122 041520 DH4: .ASCIZ /ERRPC/
1730 011052 000
1731 011053 105 051122 041520 DH10: .ASCIZ /ERRPC STATUS EXPECT INPUT BIT/
1732 011060 020040 051440 040524
1733 011066 052524 020123 042440
1734 011074 050130 041505 020124
1735 011102 044440 050116 052125
1736 011110 041040 052111 000
1737 011116 000
1738 011116 001116 001126 001124 DT1: .EVEN $ERRPC,$BDDAT,$GDDAT,0
1739 011124 000000
1740 011126 001116 000000 DT4: $ERRPC,0
1741 011132 001116 001126 001124 DT10: $ERRPC,$BDDAT,$GDDAT,BRLEV3,0
1742 011140 001466 000000
1743 011144 000 000 000 DF1: .BYTE 0,0,0,0
1744 011147 000
1745 011150 000 000 000 DF10: .BYTE 0,0,0,0,0,0
1746 011153 000 000 000
1747 011156 005015 053523 020122 MSGSWR: .ASCIZ <15><12>/SWR = /
1748 011164 020075 000
1749 011170
1750 .EVEN
1751 .SBTTL GETSWR SUBROUTINE
1752 011170 022737 000176 001136 GETSWR: CMP #SWREG,SWR ;TEST IF REAL SWR
1753 011176 001415 BEQ 1$ ;BR IF NOT
1754 011200 104400 011206 TYPE 65$ ;:TYPE ASCIZ STRING
1755 011204 000410 BR 64$ ;:GET OVER THE ASCIZ
1756 ;:65$: .ASCIZ <15><12>/DEPRESS CONT./
1757 64$:
1758 011226 000000 HALT ;WAIT FOR OPERATOR
1759 011230 000207 RTS PC ;EXIT
1760 011232 104400
1761 011234 011156
1762 011236 104407
1763 011240 012677 167672 MOV (SP)+,3SWR ;SAVE VALUE TYPED
1764 011244 000207 RTS PC ;EXIT

```

E04

 MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
 DZLPIB.P11 GETSWR SUBROUTINE

MACY11 27(732) 17-SEP-76 14:21 PAGE 43

.SBTTL SCOPE HANDLER ROUTINE

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW14=1      LOOP ON TEST
*SW11=1      INHIBIT ITERATIONS
*SW09=1      LOOP ON ERROR
*SW08=1      LOOP ON TEST IN SWR<7:0>
*CALL
*          SCOPE          ;;SCOPE=IOT

```

```

$SCOPE:
1$      CKSWR
        BIT      #BIT14,$SWR      ;;LOOP ON PRESENT TEST?
        BNE      $OVER            ;;YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR      6$
        MOV      2$ERRVEC, -(SP)   ;;IF RUNNING ON THE "XOR" TESTER CHANGE
        MOV      2$,$ERRVEC       ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
        TST      2$177060        ;;SAVE THE CONTENTS OF THE ERROR VECTOR
        MOV      (SP)+, 2$ERRVEC   ;;SET FOR TIMEOUT
        BR      $SVLAD           ;;TIME OUT ON XOR?
        CMP      (SP)+, (SP)+     ;;RESTORE THE ERROR VECTOR
        MOV      (SP)+, 2$ERRVEC   ;;GO TO THE NEXT TEST
        BR      7$              ;;CLEAR THE STACK AFTER A TIME OUT
5$:      CMP      (SP)+, 2$ERRVEC   ;;RESTORE THE ERROR VECTOR
        MOV      (SP)+, 2$ERRVEC   ;;LOOP ON THE PRESENT TEST
        BR      7$
6$: *****END OF CODE FOR THE XOR TESTER*****
        BIT      #BIT08,$SWR      ;;LOOP ON SPEC. TEST?
        BEQ      2$              ;;BR IF NO
        CMPB     2$SWR,$TSTNM     ;;ON THE RIGHT TEST? SWR<7:0>
        BEQ      $OVER           ;;BR IF YES
        TSTB     $ERFLG          ;;HAS AN ERROR OCCURRED?
        BEQ      3$              ;;BR IF NO
        CMPB     $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
        BHI      3$              ;;BR IF NO
        BIT      #BIT09,$SWR      ;;LOOP ON ERROR?
        BEQ      4$              ;;BR IF NO
        MOV      $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
        BR      $OVER
        CLRB     $ERFLG          ;;ZERO THE ERROR FLAG
        CLR      $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
        BR      1$              ;;ESCAPE TO THE NEXT TEST
3$:      BIT      #BIT11,$SWR     ;;INHIBIT ITERATIONS?
        BNE      1$              ;;BR IF YES
        TST      $PASS           ;;IF FIRST PASS OF PROGRAM
        BEQ      1$              ;;INHIBIT ITERATIONS
        INC      $ICNT           ;;INCREMENT ITERATION COUNT
        CMP      $TIMES,$ICNT     ;;CHECK THE NUMBER OF ITERATIONS MADE
        BGE      $OVER           ;;BR IF MORE ITERATION REQUIRED
        MOV      #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
        MOV      $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO

```

```

1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781 011246
1782 011246 104404
1783 011250 032777 040000 167660
1784 011256 001114
1785
1786 011260 000416
1787
1788 011262 013746 000004
1789 011266 012737 011306 000004
1790 011274 005737 177060
1791 011300 012637 000004
1792 011304 000463
1793 011306 022626
1794 011310 012637 000004
1795 011314 000423
1796 011316
1797 011316 032777 000400 167612
1798 011324 001404
1799 011326 127737 167604 001102
1800 011334 001465
1801 011336 105737 001103
1802 011342 001421
1803 011344 123737 001115 001103
1804 011352 101015
1805 011354 032777 001000 167554
1806 011362 001404
1807 011364 013737 001110 001106
1808 011372 000446
1809 011374 105037 001103
1810 011400 005037 001164
1811 011404 000415
1812 011406 032777 004000 167522
1813 011414 001011
1814 011416 005737 001202
1815 011422 001406
1816 011424 005237 001104
1817 011430 023737 001164 001104
1818 011436 002024
1819 011440 012737 000001 001104
1820 011446 013737 011524 001164

```


F04

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 SCOPE HANDLER ROUTINE

MACY11 27(732) 17-SEP-76 14:21 PAGE 44

```

1821 011454 105237 001102 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
1822 011460 113737 001102 001200 MOV $STNM,$STNM ;;SET TEST NUMBER IN APT MAILBOX
1823 011466 011637 001106 MOV (SP), $LPADR ;;SAVE SCOPE LOOP ADDRESS
1824 011472 011637 001110 MOV (SP), $LPERR ;;SAVE ERROR LOOP ADDRESS
1825 011476 005037 001166 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
1826 011502 112737 000001 001115 MOV #1, $ERMAX ;;ONLY ALLOW ONE(1) ERROR CN NEXT TEST
1827 011510 013777 001102 167422 $OVER: MOV $STNM, $DISPLAY ;;DISPLAY TEST NUMBER
1828 011516 013716 001106 MOV $LPADR, (SP) ;;FUDGE RETURN ADDRESS
1829 011522 000002 RTI ;;FIXES PS
1830 011524 003720 $MXCNT: 2000. ;;MAX. NUMBER OF ITERATIONS
1831
1832 .SBTTL ERROR HANDLER ROUTINE
1833
1834 *****
1835 *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1836 *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1837 *AND GO TO $ERRTYP ON ERROR
1838 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1839 *$SW15=1 HALT ON ERROR
1840 *$SW13=1 INHIBIT ERROR TIMEOUTS
1841 *$SW09=1 LOOP ON ERROR
1842 *CALL
1843 *
1844 * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1845
1846 011526 105237 001103 $ERROR: INCB $ERFLG ;;SET THE ERROR FLAG
1847 011532 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
1848 011534 013777 001102 167376 MOV $STNM, $DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
1849 011542 005237 001112 INC $ERTTL ;;INC THE ERROR COUNT
1850 011546 011637 001116 MOV (SP), $ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
1851 011552 162737 000002 001116 SUB #2, $ERRPC
1852 011560 117737 167332 001114 MOV $ERRPC, $ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
1853 011566 032777 020000 167342 BIT #BIT13, $SWR ;;SKIP TIMEOUT IF SET
1854 011574 001004 BNE 20$ ;;SKIP TIMEOUTS
1855 011576 004737 012502 JSR PC, $ERRTYP ;;GO TO USER ERROR ROUTINE
1856 011602 104400 001171 TYPE , $CRLF
1857 011606
1858 011606 122737 000001 001214 20$: CMPB #APTENV, $ENV ;;RUNNING IN APT MODE
1859 011614 001007 BNE 2$ ;;NO SKIP APT ERROR REPORT
1860 011616 113737 001114 011630 MOV $ITEMB, 21$ ;;SET ITEM NUMBER AS ERROR NUMBER
1861 011624 004737 013604 JSR PC, $ATY4 ;;REPORT FATAL ERROR TO APT
1862 011630 000 21$: .BYTE 0
1863 011631 000 .BYTE 0
1864 011632 000777 22$: BR 22$ ;;APT ERROR LOOP
1865 011634 005777 167276 2$: TST $SWR ;;HALT ON ERROR
1866 011640 100001 BPL 3$ ;;SKIP IF CONTINUE
1867 011642 000000 HALT ;;HALT ON ERROR!
1868 011644 032777 001000 167264 3$: BIT #BIT09, $SWR ;;LOOP ON ERROR SWITCH SET?
1869 011652 001402 BEQ 4$ ;;BR IF NO
1870 011654 013716 001110 MOV $LPERR, (SP) ;;FUDGE RETURN FOR LOOPING
1871 011660 005737 001166 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
1872 011664 001402 BEQ 5$ ;;BR IF NONE
1873 011666 013716 001166 MOV $ESCAPE, (SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
1874 011672
1875 011672 022737 010240 000042 5$: CMP #SENDAD, $42 ;;ACT-11 AUTO-ACCEPT?
1876 011700 001001 BNE 6$ ;;BRANCH IF NO

```

G04

 MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
 DZLPIB.P11 ERROR HANDLER ROUTINE

MACY11 27(732) 17-SEP-76 14:21 PAGE 45

```

1877 011702 000000          HALT                ;;YES
1878 011704          6$:                ;;
1879 011704 000002          RTI                ;;RETURN
1880
1881          .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
1882
1883          ;*****
1884          ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
1885          ;*CHANGE IT TO BINARY.
1886          ;*CALL:
1887          ;*      RDOCT                ;;READ AN OCTAL NUMBER
1888          ;*      RETURN HERE          ;;LOW ORDER BITS ARE ON TOP OF THE STACK
1889          ;*                          ;;HIGH ORDER BITS ARE IN $HIOCT
1890
1891 011706 011646          $RDOCT: MOV      (SP),-(SP)    ;;PROVIDE SPACE FOR THE
1892 011710 016666 000004 000002      MOV      4(SP),2(SP)  ;;INPUT NUMBER
1893 011716 010046          MOV      R0,-(SP)    ;;PUSH R0 ON STACK
1894 011720 010146          MOV      R1,-(SP)    ;;PUSH R1 ON STACK
1895 011722 010246          MOV      R2,-(SP)    ;;PUSH R2 ON STACK
1896 011724 104406          1$:  RDLIN          ;;READ AN ASCII LINE
1897 011726 012600          MOV      (SP)+,R0      ;;GET ADDRESS OF 1ST CHARACTER
1898 011730 005001          CLR      R1          ;;CLEAR DATA WORD
1899 011732 005002          CLR      R2
1900 011734 112046          2$:  MOV      (R0)+,-(SP)    ;;PICKUP THIS CHARACTER
1901 011736 001412          BEQ      3$          ;;IF ZERO GET OUT
1902 011740 006301          ASL      R1          ;;*2
1903 011742 006102          ROL      R2
1904 011744 006301          ASL      R1          ;;*4
1905 011746 006102          ROL      R2
1906 011750 006301          ASL      R1          ;;*8
1907 011752 006102          ROL      R2
1908 011754 042716 177770          BIC      #1C7,(SP)    ;;STRIP THE ASCII JUNK
1909 011760 062601          ADD      (SP)+,R1      ;;ADD IN THIS DIGIT
1910 011762 000764          BR      2$          ;;LOOP
1911 011764 005726          3$:  TST      (SP)+      ;;CLEAN TERMINATOR FROM STACK
1912 011766 010166 000012          MOV      R1,12(SP)    ;;SAVE THE RESULT
1913 011772 010237 012006          MOV      R2,$HIOCT
1914 011776 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
1915 012000 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
1916 012002 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
1917 012004 000002          RTI                ;;RETURN
1918 012006 000000          $HIOCT: .WORD      0      ;;HIGH ORDER BITS GO HERE
1919
1920          .SBTTL  TTY INPUT ROUTINE
1921
1922          ;*****
1923          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
1924          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
1925          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
1926          ;*WHEN OPERATING IN TTY FLAG MODE.
1927 012010 022737 000176 001136  $CKSWR: CMP      #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED?
1928 012016 001073          BNE      14$          ;;BRANCH IF NO
1929 012020 105777 167116          TST      2$TKS      ;;CHAR THERE?
1930 012024 100070          BPL      14$          ;;IF NO, DON'T WAIT AROUND
1931 012026 117746 167112          2$:  MOV      2$TKB,-(SP)    ;;SAVE THE CHAR
1932 012032 042716 177600          BIC      #1C177,(SP)    ;;STRIP-OFF THE ASCII

```


1933	012036	022726	000007		CMP	#7,(SP)+	:: IS IT A CONTROL G?
1934	012042	001061			BNE	14\$	:: NO, RETURN TO USER
1935	012044	104400	012453		TYPE	,SCNTLG	:: YES, ECHO CONTROL G
1936							
1937	012050	104400	012460	6\$:	TYPE	\$MSWR	:: TYPE CURRENT CONTENTS
1938	012054	013746	000176		MOV	\$WREG,-(SP)	:: SAVE SWREG FOR TYPEOUT
1939	012060	104401			TPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
1940	012062	104400	012471		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR
1941	012066	005046			CLR	-(SP)	:: CLEAR COUNTER
1942	012070	005046			CLR	-(SP)	:: THE NEW SWR
1943	012072	104405		7\$:	RDCHR		:: GET NEXT CHAR
1944							
1945	012074	022716	000025	8\$:	CMP	#25,(SP)	:: IS IT A CONTROL U?
1946	012100	001005			BNE	9\$	:: BRANCH IF NO
1947	012102	104400	012446		TYPE	, SCNTLU	:: YES, ECHO IT
1948	012106	062706	000006		ADD	#6, SP	:: IGNORE PREVIOUS INPUT
1949	012112	000756			BR	6\$	:: LET'S TRY IT AGAIN
1950							
1951	012114	022716	000015	9\$:	CMP	#15,(SP)	:: IS IT A <CR>?
1952	012120	001011			BNE	11\$	:: BRANCH IF NO
1953	012122	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
1954	012126	001403			BEQ	10\$	:: BRANCH IF YES
1955	012130	016677	000002	167000	MOV	2(SP), @SWR	:: SAVE NEW SWR
1956	012136	062706	000006	10\$:	ADD	#6, SP	:: CLEAR UP STACK
1957	012142	000417			BR	13\$	:: RETURN TO USER
1958	012144	022716	000012	11\$:	CMP	#12,(SP)	:: IS IT A <LF>
1959	012150	001017			BNE	15\$	:: BRANCH IF NO
1960	012152	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
1961	012156	001403			BEQ	12\$	:: YES
1962	012160	016677	000002	166750	MOV	2(SP), @SWR	:: SAVE NEW SWR
1963	012166	062706	000006	12\$:	ADD	#6, SP	:: CLEAR UP STACK
1964	012172	013716	000046		MOV	@#46,(SP)	:: GET RESTART
1965	012176	062716	000010		ADD	#10,(SP)	:: ADDRESS
1966	012202	104400	001171	13\$:	TYPE	, \$CALF	:: ECHO <CR> AND <LF>
1967	012206	000002		14\$:	RTI		:: RETURN
1968	012210	004737	013516	15\$:	JSR	PC,\$TYPEC	:: ECHO CHAR
1969	012214	042726	177770		BIC	#177770,(SP)+	:: RESTRICT TO D-7
1970	012220	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
1971	012224	001403			BEQ	16\$	:: BRANCH IF YES
1972	012226	006316			ASL	(SP)	:: NO, SHIFT PRESENT
1973	012230	006316			ASL	(SP)	:: CHAR OVER TO MAKE
1974	012232	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
1975	012234	005266	000002	16\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
1976	012240	056616	177776		BIS	-2(SP),(SP)	:: SET IN NEW CHAR
1977	012244	000712			BR	7\$	:: GET THE NEXT ONE
1978							::*****
1979							::THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1980							::CALL:
1981					* RDCHR		:: INPUT A SINGLE CHARACTER FROM THE TTY
1982					* RETURN HERE		:: CHARACTER IS ON THE STACK
1983					*		:: WITH PARITY BIT STRIPPED OFF
1984					*		
1985					:		
1986	012246	011646		\$RDCHR:	MOV	(SP), -(SP)	:: PUSH DOWN THE PC
1987	012250	016666	000004	000002	MOV	4(SP), 2(SP)	:: SAVE THE PS
1988	012256	105777	166660	1\$:	TSTB	@\$TKS	:: WAIT FOR

```

1989 012262 100375          BPL      1$          ;; A CHARACTER
1990 012264 117766 166654 000004  MOVB   $STKB,4(SP)  ;; READ THE TTY
1991 012272 042766 177600 000004  BIC     #1C<177>,4(SP)  ;; GET RID OF JUNK IF ANY
1992 012300 026627 000004 000140  CMP     4(SP),#140  ;; IS IT UPPER CASE?
1993 012306 002407          BLT     2$          ;; BRANCH IF YES
1994 012310 026627 000004 000175  CMP     4(SP),#175  ;; IS IT A SPECIAL CHAR?
1995 012316 003003          BGT     2$          ;; BRANCH IF YES
1996 012320 042766 000040 000004  BIC     #40,4(SP)   ;; MAKE IT UPPER CASE
1997 012326 000002          RTI          ;; GO BACK TO USER
1998                                     *****
1999                                     ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2000                                     ;*CALL:
2001                                     ;*      RDLIN          ;; INPUT A STRING FROM THE TTY
2002                                     ;*      RETURN HERE    ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2003                                     ;*                                     ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
2004
2005 012330 010346          $RDLIN: MOV     R3,-(SP)      ;; SAVE R3
2006 012332 012703 012436 1$:      MOV     #STTYIN,R3    ;; GET ADDRESS
2007 012336 022703 012446 2$:      CMP     #STTYIN+8.,R3  ;; BUFFER FULL?
2008 012342 101405          BLOS     4$                  ;; BR IF YES
2009 012344 104405          RDCHR    (SP)+,(R3)          ;; GO READ ONE CHARACTER FROM THE TTY
2010 012346 112613          MOVB     #177,(R3)          ;; GET CHARACTER
2011 012350 122713 000177 10$:     CMPB     3$            ;; IS IT A RUBOUT
2012 012354 001003          BNE      3$                  ;; SKIP IF NOT
2013 012356 104400 001170 4$:      TYPE     $QUES        ;; TYPE A '?'
2014 012362 000763          BR       1$                  ;; CLEAR THE BUFFER AND LOOP
2015 012364 111337 012434 3$:      MOVB     (R3),9$        ;; ECHO THE CHARACTER
2016 012370 104400 012434          TYPE     ,9$
2017 012374 122723 000015          CMPB     #15,(R3)+    ;; CHECK FOR RETURN
2018 012400 001356          BNE      2$                  ;; LOOP IF NOT RETURN
2019 012402 105063 177777          CLRB     -1(R3)        ;; CLEAR RETURN (THE 15)
2020 012406 104400 001172          TYPE     ,SLF          ;; TYPE A LINE FEED
2021 012412 012603          MOV     (SP)+,R3            ;; RESTORE R3
2022 012414 011646          MOV     (SP),-(SP)          ;; ADJUST THE STACK AND PUT ADDRESS OF THE
2023 012416 016666 000004 000002  MOV     4(SP),2(SP)    ;; FIRST ASCII CHARACTER ON IT
2024 012424 012766 012436 000004  MOV     #STTYIN,4(SP)
2025 012432 000002          RTI
2026 012434          000          9$:      .BYTE     0    ;; RETURN
2027 012435          000          .BYTE     0    ;; STORAGE FOR ASCII CHAR. TO TYPE
2028 012436 000010          $TTYIN: .BLKB     8.      ;; TERMINATOR
2029 012446 052536 005015          $CNTLU: .ASCIZ  /?U/<15><12>  ;; RESERVE 8 BYTES FOR TTY INPUT
2030 012453          136 006507 000012  $CNTLG: .ASCIZ  /?G/<15><12>  ;; CONTROL "U"
2031 012460 005015 053523 020122  $MSWR:  .ASCIZ  <15><12>/SWR = /  ;; CONTROL "G"
2032 012466 020075          000
2033 012471          040 047040 053505  $MNEW: .ASCIZ  / NEW = /
2034 012476 036440 000040

```


2035
2036
2037
2038
2039
2040
2041
2042
2043
2044 012502
2045 012502 104400 001171
2046 012506 010046
2047 012510 005000
2048 012512 153700 001114
2049 012516 001004
2050
2051 012520 013746 001116
2052
2053 012524 104401
2054 012526 000426
2055 012530 005300
2056 012532 006300
2057 012534 006300
2058 012536 006300
2059 012540 062700 001320
2060 012544 012037 012554
2061 012550 001404
2062 012552 104400
2063 012554 000000
2064 012556 104400 001171
2065 012562 012037 012572
2066 012566 001404
2067 012570 104400
2068 012572 000000
2069 012574 104400 001171
2070 012600 011000
2071 012602 001004
2072 012604 012600
2073 012606 104400 001171
2074 012612 000207
2075 012614
2076 012614 013046
2077 012616 104401
2078 012620 005710
2079 012622 001770
2080 012624 104400 012632
2081 012630 000771
2082 012632 020040 000
2083 012636

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

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

\$ERRTYP:
TYPE \$CRLF ;; "CARRIAGE RETURN" & "LINE FEED"
MOV RO, -(SP) ;; SAVE RO
CLR RO ;; PICKUP THE ITEM INDEX
BISB @#\$ITEMB, RO
BNE 1\$;; IF ITEM NUMBER IS ZERO, JUST
MOV \$ERRPC, -(SP) ;; TYPE THE PC OF THE ERROR
;; SAVE \$ERRPC FOR TYPEOUT
;; ERROR ADDRESS
TYP0C GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 6\$;; GET OUT
1\$: DEC RO ;; ADJUST THE INDEX SO THAT IT WILL
ASL RO ;; WORK FOR THE ERROR TABLE
ASL RO
ASL RO
ADD #\$ERRTB, RO ;; FORM TABLE POINTER
MOV (RO)+, 2\$;; PICKUP "ERROR MESSAGE" POINTER
BEQ 3\$;; SKIP TYPEOUT IF NO POINTER
TYPE "ERROR MESSAGE" ;; TYPE THE "ERROR MESSAGE"
2\$: .WORD 0 ;; "ERROR MESSAGE" POINTER GOES HERE
TYPE \$CRLF ;; "CARRIAGE RETURN" & "LINE FEED"
3\$: MOV (RO)+, 4\$;; PICKUP "DATA HEADER" POINTER
BEQ 5\$;; SKIP TYPEOUT IF 0
TYPE "DATA HEADER" ;; TYPE THE "DATA HEADER"
4\$: .WORD 0 ;; "DATA HEADER" POINTER GOES HERE
TYPE \$CRLF ;; "CARRIAGE RETURN" & "LINE FEED"
5\$: MOV (RO), RO ;; PICKUP "DATA TABLE" POINTER
BNE 7\$;; GO TYPE THE DATA
6\$: MOV (SP)+, RO ;; RESTORE RO
TYPE \$CRLF ;; "CARRIAGE RETURN" & "LINE FEED"
RTS PC ;; RETURN
7\$: MOV @ (RO)+, -(SP) ;; SAVE @ (RO)+ FOR TYPEOUT
TYP0C GO TYPE--OCTAL ASCII(ALL DIGITS)
TST (RO) ;; IS THERE ANOTHER NUMBER?
BEQ 6\$;; BR IF NO
TYPE TWO(2) SPACES ;; TYPE TWO(2) SPACES
BR 7\$;; LOOP
8\$: .ASCIZ / / ;; TWO(2) SPACES
.EVEN

K04

```

2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139

```

```

.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.
;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;CALL:
;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPON      ;;CALL FOR TYPEOUT
;      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;      .BYTE      M      ;;M=1 OR 0
;                               ;;1=TYPE LEADING ZEROS
;                               ;;0=SUPPRESS LEADING ZEROS
;$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;$TYPOS OR $TYPON
;CALL:
;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPON      ;;CALL FOR TYPEOUT
;$TYPON---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
;CALL:
;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
;      TYPON      ;;CALL FOR TYPEOUT
2111 012636 017646 000000 013061 $TYPOS: MOV 3(SP),-(SP) ;;PICKUP THE MODE
2112 012642 116637 000001 013061 MOV 1(SP),$OFILL ;;LOAD ZERO FILL SWITCH
2113 012650 112637 013063 MOV (SP)+,$OMODE+1 ;;NUMBER OF DIGITS TO TYPE
2114 012654 062716 000002 ADD #2,(SP) ;;ADJUST RETURN ADDRESS
2115 012660 000406 BR $TYPON
2116 012662 112737 000001 013061 $TYPON: MOV #1,$OFILL ;;SET THE ZERO FILL SWITCH
2117 012670 112737 000006 013063 MOV #6,$OMODE+1 ;;SET FOR SIX(6) DIGITS
2118 012676 112737 000005 013060 $TYPON: MOV #5,$OCNT ;;SET THE ITERATION COUNT
2119 012704 010346 MOV R3,-(SP) ;;SAVE R3
2120 012706 010446 MOV R4,-(SP) ;;SAVE R4
2121 012710 010546 MOV R5,-(SP) ;;SAVE R5
2122 012712 113704 013063 MOV $OMODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
2123 012716 005404 NEG R4
2124 012720 062704 000006 ADD #6,R4 ;;SUBTRACT IT FOR MAX. ALLOWED
2125 012724 110437 013062 MOV R4,$OMODE ;;SAVE IT FOR USE
2126 012730 113704 013061 MOV $OFILL,R4 ;;GET THE ZERO FILL SWITCH
2127 012734 016605 000012 MOV 12(SP),R5 ;;PICKUP THE INPUT NUMBER
2128 012740 005003 CLR R3 ;;CLEAR THE OUTPUT WORD
2129 012742 006105 1$: ROL R5 ;;ROTATE MSB INTO "C"
2130 012744 000404 BR 3$ ;;GO DO MSB
2131 012746 006105 2$: ROL R5 ;;FORM THIS DIGIT
2132 012750 006105 ROL R5
2133 012752 006105 ROL R5
2134 012754 010503 MOV R5,R3
2135 012756 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
2136 012760 105337 013062 DECB $OMODE ;;TYPE THIS DIGIT?
2137 012764 100016 BPL 7$ ;;BR IF NO
2138 012766 042703 BIC #177770,R3 ;;GET RID OF JUNK
2139 012772 001002 BNE 4$ ;;TEST FOR 0

```


L04

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 BINARY TO OCTAL (ASCII) AND TYPE

MACY11 27(732) 17-SEP-76 14:21 PAGE 50

2140	012774	005704		TST	R4	:: SUPPRESS THIS 0?
2141	012776	001403		BEQ	5\$	:: BR IF YES
2142	013000	005204		INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
2143	013002	052703	000060	BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
2144	013006	052703	000040	BIS	#',R3	:: MAKE ASCII IF NOT ALREADY
2145	013012	110337	013056	MOVB	R3,8\$	:: SAVE FOR TYPING
2146	013016	104400	013056	TYPE	8\$	:: GO TYPE THIS DIGIT
2147	013022	105337	013060	DECB	\$OCNT	:: COUNT BY 1
2148	013026	003347		BGT	2\$	:: BR IF MORE TO DO
2149	013030	002402		BLT	6\$	:: BR IF DONE
2150	013032	005204		INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
2151	013034	000744		BR	2\$	:: GO DO THE LAST DIGIT
2152	013036	012605		MOV	(SP)+,R5	:: RESTORE R5
2153	013040	012604		MOV	(SP)+,R4	:: RESTORE R4
2154	013042	012603		MOV	(SP)+,R3	:: RESTORE R3
2155	013044	016666	000002 000004	MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
2156	013052	012616		MOV	(SP)+,(SP)	
2157	013054	000002		RTI		:: RETURN
2158	013056	000		.BYTE	0	:: STORAGE FOR ASCII DIGIT
2159	013057	000		.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
2160	013060	000		.BYTE	0	:: OCTAL DIGIT COUNTER
2161	013061	000		.BYTE	0	:: ZERO FILL SWITCH
2162	013062	000000		.WORD	0	:: NUMBER OF DIGITS TO TYPE

```

2163
2164
2165          .SBTTL  POWER DOWN AND UP ROUTINES
2166
2167          ::*****
2168          ::POWER DOWN ROUTINE
2169          $PWRDN: MOV      $SILLUP, @#PWRVEC      ;; SET FOR FAST UP
2170                  MOV      #340, @#PWRVEC+2      ;; PRI0:7
2171                  MOV      R0, -(SP)              ;; PUSH R0 ON STACK
2172                  MOV      R1, -(SP)              ;; PUSH R1 ON STACK
2173                  MOV      R2, -(SP)              ;; PUSH R2 ON STACK
2174                  MOV      R3, -(SP)              ;; PUSH R3 ON STACK
2175                  MOV      R4, -(SP)              ;; PUSH R4 ON STACK
2176                  MOV      R5, -(SP)              ;; PUSH R5 ON STACK
2177                  MOV      @SWR, -(SP)            ;; PUSH @SWR ON STACK
2178                  MOV      SP, $SAVR6             ;; SAVE SP
2179                  MOV      $PWRUP, @#PWRVEC        ;; SET UP VECTOR
2180                  HALT
2181                  BR        .-2                    ;; HANG UP
2182
2183          ::*****
2184          ::POWER UP ROUTINE
2185          $PWRUP: MOV      $SILLUP, @#PWRVEC      ;; SET FOR FAST DOWN
2186                  MOV      $SAVR6, SP             ;; GET SP
2187                  CLR      $SAVR6                 ;; WAIT LOOP FOR THE TTY
2188                  1$: INC   $SAVR6                 ;; WAIT FOR THE INC
2189                  BNE      1$                      ;; OF WORD
2190                  MOV      (SP)+, @SWR             ;; POP STACK INTO @SWR
2191                  MOV      (SP)+, R5               ;; POP STACK INTO R5
2192                  MOV      (SP)+, R4               ;; POP STACK INTO R4
2193                  MOV      (SP)+, R3               ;; POP STACK INTO R3
2194                  MOV      (SP)+, R2               ;; POP STACK INTO R2
2195                  MOV      (SP)+, R1               ;; POP STACK INTO R1
2196                  MOV      (SP)+, R0               ;; POP STACK INTO R0
2197                  MOV      $PWRDN, @#PWRVEC        ;; SET UP THE POWER DOWN VECTOR
2198                  MOV      #340, @#PWRVEC+2        ;; PRI0:7
2199                  TYPE     PWRMSG                   ;; REPORT THE POWER FAILURE
2200                  SPWRMG: .WORD PWRMSG              ;; POWER FAIL MESSAGE POINTER
2201                  MOV      (PC)+, (SP)              ;; RESTART AT IOTEST
2202                  SPWRAD: .WORD IOTEST              ;; RESTART ADDRESS
2203                  RTI
2204                  $ILLUP: HALT                      ;; THE POWER UP SEQUENCE WAS STARTED
2205                  BR        .-2                    ;; BEFORE THE POWER DOWN WAS COMPLETE
2206                  $SAVR6: 0                          ;; PUT THE SP HERE
2207                  PWRMSG: .ASCIZ <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
2208
2209
2210
2211
2212
2213
2214          .EVEN

```


2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270

.SBTTL TYPE ROUTINE

*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*

*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*

\$TYPE:	TSTB	\$TPFLG	;; IS THERE A TERMINAL?
	BPL	1\$	;; BR IF YES
	HALT		;; HALT HERE IF NO TERMINAL
	BR	3\$	;; LEAVE
1\$:	MOV	RO, -(SP)	;; SAVE RO
	MOV	22(SP), RO	;; GET ADDRESS OF ASCIZ STRING
	CMPB	#APTENV, \$ENV	;; RUNNING IN APT MODE
	BNE	62\$	;; NO, GO CHECK FOR APT CONSOLE
	BITB	#APTSPool, \$ENV	;; SPOOL MESSAGE TO APT
	BEQ	62\$	;; NO, GO CHECK FOR CONSOLE
	MOV	RO, 61\$	;; SETUP MESSAGE ADDRESS FOR APT
	JSR	PC, \$ATY3	;; SPOOL MESSAGE TO APT
61\$:	.WORD	0	;; MESSAGE ADDRESS
62\$:	BITB	#APTCSUP, \$ENV	;; APT CONSOLE SUPPRESSED
	BNE	60\$	;; YES, SKIP TYPE OUT
2\$:	MOVB	(RO)+, -(SP)	;; PUSH CHARACTER TO BE TYPED ONTO STACK
	BNE	4\$	;; BR IF IT ISN'T THE TERMINATOR
	TST	(SP)+	;; IF TERMINATOR POP IT OFF THE STACK
60\$:	MOV	(SP)+, RO	;; RESTORE RO
3\$:	ADD	#2, (SP)	;; ADJUST RETURN PC
	RTI		;; RETURN
4\$:	CMPB	#HT, (SP)	;; BRANCH IF <HT>
	BEQ	8\$	
	CMPB	#CRLF, (SP)	;; BRANCH IF NOT <CRLF>
	BNE	5\$	
	TST	(SP)+	;; POP <CR><LF> EQUIV
	TYPE		;; TYPE A CR AND LF
	\$CRLF		
	CLRB	\$CHARCNT	;; CLEAR CHARACTER COUNT
	BR	2\$	;; GET NEXT CHARACTER
5\$:	JSR	PC, \$TYPEC	;; GO TYPE THIS CHARACTER
6\$:	CMPB	\$FILLC, (SP)+	;; IS IT TIME FOR FILLER CHARS.?
	BNE	2\$	;; IF NO GO GET NEXT CHAR.
	MOV	\$NULL, -(SP)	;; GET # OF FILLER CHARS. NEEDED
			;; AND THE NULL CHAR.
7\$:	DECB	1(SP)	;; DOES A NULL NEED TO BE TYPED?
	BLT	6\$	;; BR IF NO--GO POP THE NULL OFF OF STACK

```

013460 004737 013516 JSR PC,$TYPEC ::GO TYPE A NULL
013464 105337 013562 DECB $CHARCNT ::DO NOT COUNT AS A COUNT
013470 000770 BR 7$ ::LOOP

                                :HORIZONTAL TAB PROCESSOR

013472 112716 000040 8$: MOVB 8' (SP) ::REPLACE TAB WITH SPACE
013476 004737 013516 9$: JSR PC,$TYPEC ::TYPE A SPACE
013502 132737 000007 013562 BITB 87,$CHARCNT ::BRANCH IF NOT AT
013510 001372 BNE 9$ ::TAB STOP
013512 005726 TST (SP)+ ::POP SPACE OFF STACK
013514 000724 BR 2$ ::GET NEXT CHARACTER
013516 105777 165424 $TYPEC: TSTB 2$TPS ::WAIT UNTIL PRINTER IS READY
013522 100375 BPL $TYPEC
013524 116677 000002 165416 MOVB 2(SP),2$TPB ::LOAD CHAR TO BE TYPED INTO DATA REG.
013532 122766 000015 000002 CMPB 8CR,2(SP) ::IS CHARACTER A CARRIAGE RETURN?
013540 001003 BNE 1$ ::BRANCH IF NO
013542 105037 013562 CLRB $CHARCNT ::YES--CLEAR CHARACTER COUNT
013546 000406 BR $TYPEX ::EXIT
013550 122766 000012 000002 1$: CMPB 8LF,2(SP) ::IS CHARACTER A LINE FEED?
013556 001402 BEQ $TYPEX ::BRANCH IF YES
013560 105227 INCB (PC)+ ::COUNT THE CHARACTER
013562 000000 $CHARCNT: WORD 0 ::CHARACTER COUNT STORAGE
013564 000207 $TYPEX: RTS PC

```


.SBTTL APT COMMUNICATIONS ROUTINE

```

*****
2301 013566 112737 000001 014032 SATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
2302 013574 112737 000001 014030 SATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
2303 013602 000403 BR SATYC
2304 013604 112737 000001 014032 SATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
2305 013612 SATYC:
2306 013612 010046 MOV RO,-(SP) ;;PUSH RO ON STACK
2307 013614 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
2308 013616 105737 014030 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
2309 013622 001450 BEQ 5$ ;;IF NOT: BR
2310 013624 122737 000001 001214 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
2311 013632 001031 BNE 3$ ;;IF NOT: BR
2312 013634 132737 000100 001215 BITB #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2313 013642 001425 BEQ 3$ ;;IF NOT: BR
2314 013644 017600 000004 MOV #4(SP),RO ;;GET MESSAGE ADDR.
2315 013650 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
2316 013656 005737 001174 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
2317 013662 001375 BNE 1$ ;;IF NOT: WAIT
2318 013664 010037 001210 MOV RO,$MSGAD ;;PUT ADDR IN MAILBOX
2319 013670 105720 2$: TSTB (RO)+ ;;FIND END OF MESSAGE
2320 013672 001376 BNE 2$
2321 013674 163700 001210 SUB $MSGAD,RO ;;SUB START OF MESSAGE
2322 013700 006200 ASR RO ;;GET MESSAGE LGTH IN WORDS
2323 013702 010037 001212 MOV RO,$MSGLGTH ;;PUT LENGTH IN MAILBOX
2324 013706 012737 000004 001174 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
2325 013714 000413 BR 5$
2326 013716 017637 000004 013742 3$: MOV #4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
2327 013724 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
2328 013732 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
2329 013736 004737 013304 JSR PC,$TYPE ;;CALL TYPE MACRO
2330 013742 000000 4$: .WORD 0
2331 013744 5$:
2332 013744 105737 014032 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
2333 013750 001416 BEQ 12$ ;;IF NOT: BR
2334 013752 005737 001214 TST $ENV ;;RUNNING UNDER APT?
2335 013756 001413 BEQ 12$ ;;IF NOT: BR
2336 013760 005737 001174 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
2337 013764 001375 BNE 11$ ;;IF NOT: WAIT
2338 013766 017637 000004 001176 MOV #4(SP),$FATAL ;;GET ERROR #
2339 013774 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
2340 014002 005237 001174 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
2341 014006 105037 014032 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
2342 014012 105037 014031 CLRB $LFLG ;;CLEAR LOG FLAG
2343 014016 105037 014030 CLRB $MFLG ;;CLEAR MESSAGE FLAG
2344 014022 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
2345 014024 012600 MOV (SP)+,RO ;;POP STACK INTO RO
2346 014026 000207 RTS PC ;;RETURN
2347 014030 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
2348 014031 000 $LFLG: .BYTE 0 ;;LOG FLAG
2349 014032 000 $FFLG: .BYTE 0 ;;FATAL FLAG
2350 014034 .EVEN
2351 000200 APTSIZE=200

```

2352 000001 APTENV=001
2353 000100 APTSPool=100
2354 000040 APTCSUP=040
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390

014034 010046
014036 016600 000002
014042 005740
014044 111000
014046 006300
014050 016000 014056
014054 000200

.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 \$TRPAD(RO), RO ;;INDEX TO TABLE
RTS RO ;;GO TO ROUTINE

.SBTTL TRAP TABLE

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

	ROUTINE		

\$TRPAD:	\$TYPE	::CALL=TYPE	TRAP+0(104400) TTY TYPEOUT ROUTINE
	\$TYPOC	::CALL=TYPOC	TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	::CALL=TYPOS	TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	::CALL=TYPON	TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$CKSWR	::CALL=CKSWR	TRAP+4(104404) TEST FOR CHANGE IN SOFT-SWR
	\$RDCHR	::CALL=RDCHR	TRAP+5(104405) TTY TYPEIN CHARACTER ROUTINE
	\$RDLIN	::CALL=RDLIN	TRAP+6(104406) TTY TYPEIN STRING ROUTINE
	\$RDOCT	::CALL=RDOCT	TRAP+7(104407) READ AN OCTAL NUMBER FROM TTY
	.END		

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC
DZLP1B.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]


```

EM2      010343
EM3      010373
EM4      010424
EM5      010456
EM6      010511
EM7      010536
ERRVEC=  000004
EXTCNT   010162
EXTTMP   010160
EXTTST   010040
FLIP     010066
FULNUL   010272
GETSWR   011170
GNS      =  *****

```

GRBHIO 001442
GRDAI 001436

GRDIO 001440

GRIVA 001450

GRIVE 001454

GRIYSA 001452

GRIVSB 001456

GRSTAT 001434

HT = 000011

IO TEST 002622

IOTVEC= 000020

LF = 000012
MSGSR 011156

MSGSWR 011156
NOTINT 001446

NOTLCH 001444

PC =%000007

BTSA - 133333

PIRO = 177772
PIRAVE = 000240

PINOVE= 000240

LPS-11-DRA DIAGNOSTIC
REFERENCE TABLE -- USER SYMBOLS

507	1667#												
513	1672#												
519	1677#												
525	1682#												
531	1687#												
537	1691#												
286#	603	604*	615*	1788	1789*	1791*	1794*						
1601*	1611*	1616#											
1605*	1606	1609*	1610	1615#									
328	1595#	1599											
1596	1598	1601#											
1652	1656#												
643	650	656	1752#										
321	633	637	641	647	654	1756	2382	2383	2384	2385	2386	2387	
2388	2389												
561#	662*	663*	846*	847									
559#	658*	659*	683*	858*	949*	952	962*	965	976*	979	990*	993	
1012	1022	1045*	1049	1070*	1079	1097*	1101	1102*	1106	1107*	1111	1112*	
1116	1117*	1121	1122*	1126	1127*	1131	1132*	1136	1137*	1141	1142*	1146	
1147*	1151	1152*	1159*	1173	1174*	1178	1179*	1183	1184*	1188	1189*	1193	
1194*	1198	1199*	1203	1204*	1208	1209*	1213	1214*	1218	1219*	1223	1224*	
1255	1277	1396	1427*	1439*	1463*	1495	1605	1606*	1609	1610*			
560#	660*	661*	684*	691*	692	703*	705	714*	715	725*	726	738*	
739	752*	753*	754	768*	769*	770	784*	785*	786*	787*	788*	789*	
790*	791*	792*	793*	794*	797	807*	808*	809*	810*	811*	812*	813*	
814*	815*	816*	817*	819	831*	832	857*	948*	951*	961*	964*	975*	
978*	989*	992*	1011*	1021*	1044*	1047*	1048*	1069*	1076*	1077*	1096*	1099*	
1100*	1104*	1105*	1109*	1110*	1114*	1115*	1119*	1120*	1124*	1125*	1129*	1130*	
1134*	1135*	1139*	1140*	1144*	1145*	1149*	1150*	1168*	1171*	1172*	1176*	1177*	
1181*	1182*	1186*	1187*	1191*	1192*	1196*	1197*	1201*	1202*	1206*	1207*	1211*	
1212*	1216*	1217*	1221*	1222*	1238*	1253*	1276*	1395*	1426*	1429*	1438*	1462*	
1465*	1494*	1603*	1604*	1607*	1608*								
564#	664*	674*	1272*	1298*	1309*	1327*	1336*	1358*	1390*	1407*	1491*	1504*	
1534*	1543*	1553*	1560*	1571*	1583*								
566#	667*	668*	676*	1274*	1300*	1329*	1360*	1369*	1392*	1409*	1489*	1506*	
1545*	1573*	1585*											
565#	665*	666*	674	675*	1273*	1299*	1328*	1358	1359*	1391*	1407	1408*	
1491	1492*	1504	1505*	1543	1544*	1571	1572*	1583	1584*				
567#	669*	670*	676	677*	1275*	1300	1301*	1329	1330*	1361*	1392	1393*	
1409	1410*	1490*	1506	1507*	1545	1546*	1573	1574*	1585	1586*			
558#	657*	682*	860*	861	872*	873	883*	884	89				

PR0	=	000000
PR1	=	000040
PR2	=	000100
PR3	=	000140
PR4	=	000200
PR5	=	000240
PR6	=	000300
PR7	=	000340
PS	=	177776
PSW	=	177776

```
PWRMSG 013236
PWRVEC= 000024
RBEG 001500
RDCHR = 104405
RDLIN = 104406
RDOCT = 104407
RESVEC= 000010
RO      =%000000
```

R1 = %000001

```
R2      =%000002
R3      =%000003
```

```
R4      =%000004
R5      =%000005
R6      =%000006
R7      =%000007
SP      =%000006
```

STACK = 001100
STKLMT= 177774
SWR 001136

SWREG 000176

220#												
221#												
222#												
223#												
224#												
225#												
226#												
227#												
200#	201											
201#	1271*	1311*	1318*	1338*	1348*	1371*	1377*	1405*	1508*	1535*	1547*	1554*
1561*	1567*	1575*	1587*									
2200	2207#											
292#	593*	594*	2169*	2170*	2179*	2185*	2197*	2198*				
577	579#											
1943	2009	2387#										
1896	2388#											
1762	2389#											
287#												
208#	576*	578*	626	1095*	1099	1103*	1104	1108*	1109	1113*	1114	1118*
1119	1123*	1124	1128*	1129	1133*	1134	1138*	1139	1143*	1144	1148*	1149
1153*	1167*	1171	1175*	1176	1180*	1181	1185*	1186	1190*	1191	1195*	1196
1200*	1201	1205*	1206	1210*	1211	1215*	1216	1220*	1221	1225*	1255*	1277*
1396*	1495*	1643*	1646	1893	1897*	1900	1916*	2046	2047*	2048*	2055*	2056*
2057*	2058*	2059*	2060	2065	2070*	2072*	2076	2078	2171	2196*	2238	2239*
2244	2249	2252*	2306	2314*	2318	2319	2321*	2322*	2323	2345*	2365	2366*
2367	2368*	2369*	2370*	2371*								
209#	1101*	1102	1106*	1107	1111*	1112	1116*	1117	1121*	1122	1126*	1127
1131*	1132	1136*	1137	1141*	1142	1146*	1147	1151*	1152	1155	1173*	1174
1178*	1179	1183*	1184	1188*	1189	1193*	1194	1198*	1199	1203*	1204	1208*
1209	1213*	1214	1218*	1219	1223*	1224	1227	1894	1898*	1902*	1904*	1906*
1909*	1912	1915*	2172	2195*	2307	2344*						
210#	1895	1899*	1903*	1905*	1907*	1913	1914*	2173	2194*			
211#	2005	2006*	2007	2010*	2011	2015	2017	2019*	2021*	2119	2128*	2134*
2135*	2138*	2143*	2144*	2145	2154*	2174	2193*					
212#	2120	2122*	2123*	2124*	2125	2126*	2140	2142*	2150*	2153*	2175	2192*
213#	2121	2127*	2129*	2131*	2132*	2133*	2134	2152*	2176	2191*		
214#	216	581*	582*	583								
215#	217											
216#	585*	603*	613*	615	673*	1310	1316	1337	1346	1370	1375	1404
1503	1542	1566	1570	1763	1788*	1791	1793	1794	1823	1824	1828*	1950
1870*	1873*	1891*	1892*	1893*	1894*	1895*	1897	1900*	1908*	1909	1911	1912*
1914	1915	1916	1931*	1932*	1933	1938*	1941*	1942*	1945	1948*	1951	1953
1955	1956*	1958	1960	1962	1963*	1964*	1965*	1969*	1970	1972*	1973*	1974*

SW0	=	000001	255#		
SW00	=	000001	245#	255	
SW01	=	000002	244#	254	
SW02	=	000004	243#	253	
SW03	=	000010	242#	252	
SW04	=	000020	241#	251	
SW05	=	000040	240#	250	
SW06	=	000100	239#	249	
SW07	=	000200	238#	248	
SW08	=	000400	237#	247	
SW09	=	001000	236#	246	
SW1	=	000002	254#		
SW10	=	002000	235#		
SW11	=	004000	234#		
SW12	=	010000	233#		
SW13	=	020000	232#		
SW14	=	040000	231#		
SW15	=	100000	230#		
SW2	=	000004	253#		
SW3	=	000010	252#		
SW4	=	000020	251#		
SW5	=	000040	250#		
SW6	=	000100	249#		
SW7	=	000200	248#		
SW8	=	000400	247#		
SW9	=	001000	246#		
TBITVE	=	000014	288#		
TKVEC	=	000060	295#		
TPVEC	=	000064	296#		
TRAPVE	=	000034	294#	591*	592*
TRTVEC	=	000014	289#		
TST1	=	002654	681#		
TST10	=	003216	764#		
TST11	=	003302	782#		
TST12	=	003414	799	805#	
TST13	=	003524	821	827#	
TST14	=	003602	842#		
TST15	=	003660	856#		
TST16	=	003730	863	870#	
TST17	=	003766	875	881#	
TST2	=	002672	689#		
TST20	=	004024	886	892#	
TST21	=	004066	898	905#	
TST22	=	004124	910	916#	
TST23	=	004166	922	928#	
TST24	=	004230	935	943#	
TST25	=	004312	954	960#	
TST26	=	004362	967	974#	
TST27	=	004432	981	988#	
TST3	=	002730	694	700#	
TST30	=	004502	995	1001#	
TST31	=	004652	1037#		
TST32	=	004752	1062#		
TST33	=	005100	1091#		
TST34	=	005460	1157	1163#	
TST35	=	006040	1229	1235#	

```
TST36      006110
TST37      006156
TST4       002766
TST40      006270
TST41      006376
TST42      006532
TST43      006636
TST44      007010
TST45      007164
TST46      007302
TST47      007454
TST5       003024
TST50      007550
TST51      007644
TST52      007770
TST53      010036
TST6       003062
TST7       003140
TYPE      = 104400
```

```

TYP0C = 104401
TYP0N = 104403
TYP0S = 104402
$APTHD 001000
$ARG1 001732
$ASTAT= ***** U
$ATYC 013612
$ATY1 013566
$ATY3 013574
$ATY4 013604
$BASE 001250
$BDADR 001122
$BDDAT 001126

```

\$CDW1	001254
\$CDW2	001256
\$CHARC	013562
\$CKSWR	012010
\$CMTAG	001100
\$CM1 =	000002
\$CM2 =	000004
\$CM3 =	000002
\$CNTLG	012453
\$CNTLU	012446
\$CPUOP	001222
\$CRLF	001171
\$DDWD	001260
\$DDW1	001262
\$DDW10	001304
\$DDW11	001306
\$DDW12	001310

[illegible]

MACY11 27(732) 17-SEP-76 14:21 PAGE 63

[illegible]

\$MADR1	001226	443#																		
\$MADR2	001232	447#																		
\$MADR3	001236	450#																		
\$MADR4	001242	453#																		
\$MAIL	001174	360	364	416#	617	1922	1958	2240												
\$MAMS1	001224	437#																		
\$MAMS2	001230	445#																		
\$MAMS3	001234	448#																		
\$MAMS4	001240	451#																		
\$MBADR	001002	360#																		
\$MFLG	014030	2302*	2308	2343*	2347#															
\$MNEW	012471	1940	2033#																	
\$MSGAD	001210	423#	2318*	2321																
\$MSGLG	001212	424#	2323*																	
\$MSGTY	001174	417#	2316	2324*	2336	2340*														
\$MSWR	012460	1937	2031#																	
\$MTYP1	001225	438#																		
\$MTYP2	001231	446#																		
\$MTYP3	001235	449#																		
\$MTYP4	001241	452#																		
\$MXCNT	011524	1820	1830#																	
\$NULL	001152	397#	2267	2296																
\$NWTST=	000001	678#	686#	697#	709#	720#	731#	745#	761#	779#	802#	824#	839#	853#						
		867#	878#	889#	902#	913#	925#	940#	957#	971#	985#	998#	1034#	1059#						
		1088#	1160#	1232#	1247#	1264#	1291#	1320#	1352#	1380#	1412#	1448#	1479#	1512#						
		1529#	1548#	1577#	1591#															
		2118*	2147*	2160#																
\$OCNT	013060	2113*	2117*	2122	2125*	2136*	2162#													
\$OMODE	013062	1784	1800	1808	1818	1827#														
\$OVER	011510	420#	618*	1634*	1635*	1653	1814	1831												
\$PASS	001202	362#																		
\$PASTM	001006	456#	625																	
\$PRIOR	001246	2202#																		
\$PWRAD	013224	593	2169#	2197																
\$PWRDN	013064	2200#																		
\$PWARMG	013220	2179	2185#																	
\$PWRUP	013136	407#	1880	2013	2029	2296														
\$QUES	001170	1986#	2387																	
\$RDCHR	012246	2390																		
\$RDDEC=	***** U	2005#	2388																	
\$RDLIN	012330	1891#	2389																	
\$RDOCT	011706	1998#																		
\$RDSZ =	000010	401#																		
\$REGAD	001156	403#																		
\$REGO	001160	404#																		
\$REG1	001162	1652#																		
\$RTNAD	010252	2390																		
\$R2A =	***** U	2390																		
\$SAVRE=	***** U	2390																		
\$SAVR6	013234	2178*	2186	2187*	2188*	2206#														
\$SCOPE	011246	587	1781#																	
\$SETUP=	000037	536#	586	587	589	591	593	595	596	597	599	616	1632	1782						
		1846	1868	1875																
\$STUP =	177777	536#																		
\$SVLAD	011454	1792	1821#																	
\$SVPC =	000214	335#	340																	
\$SWR =	165400	177#	187	306	307	308	309	310	311	312	405	406	407	596						

I-B LPS-11-DRA DIAGNOSTIC
CROSS REFERENCE TABLE -- MACRO NAMES

CLERIN	940#	949	962	976	990	1045	1070	1097	1169	1427	1463				
CLEROT	939#	947	961	975	989	1044	1048	1069	1077	1096	1100	1105	1110	1115	1120
	1125	1130	1135	1140	1145	1150	1168	1172	1177	1182	1187	1192	1197	1202	1207
CLRVCT	1212	1217	1222	1426	1462	1602	1607								
COMMEN	573#	674	1407	1504	1543	1571	1583								
ENDCOM	298#														
ERROR	298#														
	192#	695	707	718	729	742	757	774	800	822	835	850	864	876	887
	899	911	923	936	955	968	982	996	1016	1027	1052	1082	1158	1230	1243
	1258	1285	1288	1304	1317	1334	1343	1347	1364	1376	1402	1434	1444	1470	1501
	1520	1523	1541	1568											
ESCAPE	298#														
GETPRI	298#														
MULT	298#														
NEWST	298#	678	686	697	709	720	731	745	761	779	802	824	839	853	867
	878	889	902	913	925	940	957	971	985	998	1034	1059	1088	1160	1232
	1247	1263	1291	1320	1352	1380	1412	1448	1479	1512	1529	1548	1577	1591	
POP	298#	1914	2190	2191	2344	2345									
PUSH	298#	1893	2171	2177	2305	2307	2328								
REPORT	298#														
SCOPE	193#	681	689	700	712	723	734	748	764	782	805	827	842	856	870
	881	892	905	916	928	943	960	974	988	1001	1037	1062	1091	1163	1235
	1250	1267	1294	1323	1355	1383	1415	1451	1482	1515	1532	1551	1580	1594	1631
SETPRI	298#														
SETTRA	2373#	2383	2384	2385	2386	2387	2388	2389							
SETUP	298#	580													
SKIP	298#	694	706	717	728	741	799	821	863	875	886	898	910	922	935
	954	967	981	995	1015	1026	1051	1081	1157	1229	1242	1257	1283	1314	1342
	1344	1374	1388	1403	1417	1425	1433	1443	1453	1461	1469	1487	1502	1569	
SLASH	298#														
SPACE	298#														
STARS	298#	333	345	347	354	368	410	414	573	575	678	680	686	688	697
	699	709	711	720	722	731	733	745	747	761	763	779	781	802	804
	824	826	839	841	853	855	867	869	878	880	889	891	902	904	913
	915	925	927	940	942	957	959	971	973	985	987	998	1000	1034	1036
	1059	1061	1089	1090	1160	1162	1232	1234	1247	1249	1264	1266	1291	1293	1320
	1322	1352	1354	1380	1382	1412	1414	1448	1450	1479	1481	1512	1514	1529	1531
	1548	1550	1577	1579	1591	1593	1621	1769	1834	1883	1922	1978	1998	2039	2088
	2167	2183	2219	2300	2359										
SWRSU	298#	601#													
TRMTRP	2373#														
TYPBIN	298#														
TYPDEC	298#														
TYPNAM	298#														
TYPNUM	298#														
TYPOCS	298#														
TYPOCT	298#	1938	2051	2075											
TYPTXT	298#	631	635	639	645	652	1754								
\$\$CMRE	365#	403	404												
\$\$CMTM	365#														
\$\$ESCA	298#														
\$\$NEWT	298#	678	686	697	709	720	731	745	761	779	802	824	839	853	867
	878	889	902	913	925	940	957	971	985	998	1034	1059	1088	1160	1232
	1247	1264	1291	1320	1352	1380	1412	1448	1479	1512	1529	1548	1577	1591	
\$\$SET	2373#	2383	2384	2385	2386	2387	2388	2389							
\$\$SETM	617#	618													

95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110
14994	14995	14996	14997	14998	14999	15000	15001	15002	15003	15004	15005	15006	15007	15008	15009
15010	15011	15012	15013	15014	15015	15016	15017	15018	15019	15020	15021	15022	15023	15024	15025
15026	15027	15028	15029	15030	15031	15032	15033	15034	15035	15036	15037	15038	15039	15040	15041
15042	15043	15044	15045	15046	15047	15048	15049	15050	15051	15052	15053	15054	15055	15056	15057
15058	15059	15060	15061	15062	15063	15064	15065	15066	15067	15068	15069	15070	15071	15072	15073
15074	15075	15076	15077	15078	15079	15080	15081	15082	15083	15084	15085	15086	15087	15088	15089
15090	15091	15092	15093	15094	15095	15096	15097	15098	15099	15100	15101	15102	15103	15104	15105
15106	15107	15108	15109	15110	15111	15112	15113	15114	15115	15116	15117	15118	15119	15120	15121
15122	15123	15124	15125	15126	15127	15128	15129	15130	15131	15132	15133	15134	15135	15136	15137
15138	15139	15140	15141	15142	15143	15144	15145	15146	15147	15148	15149	15150	15151	15152	15153
15154	15155	15156	15157	15158	15159	15160	15161	15162	15163	15164	15165	15166	15167	15168	15169
15170	15171	15172	15173	15174	15175	15176	15177	15178	15179	15180	15181	15182	15183	15184	15185
15186	15187	15188	15189	15190	15191	15192	15193	15194	15195	15196	15197	15198	15199	15200	15201
15202	15203	15204	15205	15206	15207	15208	15209	15210	15211	15212	15213	15214	15215	15216	15217
15218	15219	15220	15221	15222	15223	15224	15225	15226	15227	15228	15229	15230	15231	15232	15233
15234	15235	15236	15237	15238	15239	15240	15241	15242	15243	15244	15245	15246	15247	15248	15249
15250	15251	15252	15253	15254	15255	15256	15257	15258	15259	15260	15261	15262	15263	15264	15265
15266	15267	15268	15269	15270	15271	15272	15273	15274	15275	15276	15277	15278	15279	15280	15281
15282	15283	15284	15285	15286	15287	15288	15289	15290	15291	15292	15293	15294	15295	15296	15297
15298	15299	15300	15301	15302	15303	15304	15305	15306	15307	15308	15309	15310	15311	15312	15313
15314	15315	15316	15317	15318	15319	15320	15321	15322	15323	15324	15325	15326	15327	15328	15329
15330	15331	15332	15333	15334	15335	15336	15337								

ADD	6559	661	663	666	668	670	1528	1909	1948	1956	1963	1965	2059	2114	2124
ASL	2353	2315	2327	2339											
ASR	759	776	1030	1056	1084	1446	1475	1902	1904	1906	1972	1973	1974	2056	2057
BEQ	2058	2369													
BGE	627	694	706	717	728	741	756	773	799	821	834	849	910	922	
BGT	945	954	967	981	995	1009	1015	1026	1051	1081	1157	1229	1242	1342	
BIC	1387	1461	1486	1644	1753	1798	1800	1802	1806	1815	1847	1869	1872	1901	1954
BIS	1961	1971	2061	2066	2079	2141	2243	2256	2291	2309	2313	2333	2335		
BISB	1818														
BIT	1638	1995	2148												
BITB	1804														
BLOS	769	1013	1021	1023	1072	1075	1094	1166	1400	1438	1499	1518	1635	1908	1932
BNE	1969	1991	1996	2138											
BPL	753	1102	1107	1112	1117	1122	1127	1132	1137	1142	1147	1152	1174	1179	1184
BR	1189	1194	1199	1204	1209	1214	1219	1224	1302	1332	1362	1439	1536	1555	1976
BRB	2042	2144													
BT	862	874	885	897	944	1008	1025	1041	1066	1341	1386	1416	1424	1452	1460
BTB	1485	1783	1797	1805	1812	1853	1868								
BW	619	2242	2247	2279	2312										
BWL	2008														
BWT	1993	2149	2270												
BZ	1257	1433													
BZB	584	608	630	744	760	777	837	852	863	875	886	898	1031	1042	1057
BZC	1067	1085	1417	1425	1447	1453	1476	1519	1522	1612	1658	1784	1813	1854	1859
BZD	1876	1928	1934	1946	1952	1959	2012	2018	2049	2071	2139	2189	2241	2248	2250
BZE	2258	2266	2280	2287	2311	2317	2320	2337							
BZF	1443	1469	1866	1930	1989	2137	2235	2284							
BZG	577	632	636	640	646	653	1283	1314	1344	1374	1388	1403	1487	1502	1565
BZH	1569	1599	1755	1786	1792	1795	1808	1811	1864	1910	1949	1957	1977	2014	2054
BZJ	2081	2115	2130	2151	2181	2205	2237	2263	2273	2282	2289	2303	2325		
BZK	576	582	596	597	618	675	677	682	683	684	702	737	752	830	845
BZL	857	893	917	930	931	948	950	961	975	989	1024	1044	1048	1069	1077
BZM	1096	1100	1105	1110	1115	1120	1125	1130	1135	1140	1145	1150	1168	1172	1177
BZN	1182	1187	1192	1197	1202	1207	1212	1217	1222	1237	1252	1270	1271	1282	1297
	1311	1318	1326	1338	1339	1348	1357	1371	1377	1389	1393	1405	1406	1408	1410
	1421	1426	1428	1430	1440	1441	1458	1462	1464	1466	1488	1492	1505	1507	1508
	1509	1533	1544	1546	1547	1552	1561	1567	1572	1574	1575	1584	1586	1587	1603
	1607	1632	1633	1656	1810	1825	1898	1899	1941	1942	2047	2128	2187		
CLRB	1809	2019	2262	2288	2341	2342	2343								
CMP	583	607	693	716	727	740	755	772	798	820	909	921	953	966	980
	994	1014	1050	1080	1156	1228	1239	1241	1254	1310	1316	1337	1346	1370	1375
	1404	1503	1521	1542	1566	1570	1752	1793	1817	1875	1927	1933	1945	1951	1958
	1992	1994	2007												
CMPB	833	848	1799	1803	1858	2011	2017	2240	2255	2257	2265	2286	2290	2310	
COM	771	785	786	787	788	789	790	791	792	793	794	808	809	810	811
	812	813	814	815	816	817	1006	1103	1108	1113	1118	1123	1128	1133	1138
	1143	1148	1153	1175	1180	1185	1190	1195	1200	1205	1210	1215	1220	1225	
DEC	1611	1636	1657	2055											
DECB	2136	2147	2269	2272											
EMT	1922														
HALT	321	1758	1867	1877	2180	2204	2236								
INC	743	1634	1816	1849	1975	2142	2150	2188	2340						
INCB	836	851	1821	1846	2292										
LOT	1932														

006

MAINDEC-11-DZLPI-S
DZLPIB.P11LPS-11-DRA DIAGNOSTIC
CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

MACY11 27(732) 17-SEP-76 14:21 PAGE 71

JMP	326	327	328	628	946	1588	1651	1659							
JSR	643	650	656	1596	1598	1646	1955	1861	1968	2245	2264	2271	2278	2329	
MOV	579	581	585	587	588	589	590	591	592	593	594	595	599	600	603
	604	605	606	611	612	613	615	621	623	624	625	644	651	657	658
	660	662	664	665	667	669	671	673	674	676	690	691	692	701	703
	705	713	714	715	724	725	726	735	736	738	739	749	750	754	755
	766	768	770	763	784	797	806	807	819	828	829	832	843	844	858
	859	860	861	871	872	873	882	883	884	894	895	896	906	907	908
	918	919	920	929	932	934	949	951	952	962	963	964	965	976	977
	978	979	990	991	992	993	1002	1004	1005	1007	1011	1012	1022	1038	1039
	1045	1047	1049	1063	1064	1070	1071	1076	1079	1093	1095	1097	1099	1101	1104
	1105	1109	1111	1114	1116	1119	1121	1124	1126	1129	1131	1134	1136	1139	1141
	1144	1146	1149	1151	1155	1165	1167	1169	1171	1173	1176	1178	1181	1183	1186
	1188	1191	1193	1196	1198	1201	1203	1206	1208	1211	1213	1216	1218	1221	1223
	1227	1236	1238	1240	1251	1253	1255	1256	1268	1272	1273	1274	1275	1276	1277
	1298	1299	1300	1301	1309	1327	1328	1329	1330	1331	1336	1340	1358	1359	1360
	1361	1369	1390	1391	1392	1394	1395	1396	1407	1409	1419	1420	1422	1427	1429
	1455	1456	1457	1463	1465	1489	1490	1491	1493	1494	1495	1504	1506	1516	1524
	1526	1527	1534	1535	1543	1545	1553	1554	1560	1571	1573	1581	1583	1585	1595
	1597	1601	1604	1605	1606	1608	1609	1610	1639	1643	1763	1788	1789	1791	1794
	1807	1819	1820	1823	1824	1827	1828	1848	1850	1870	1873	1891	1892	1893	1894
	1895	1897	1912	1913	1914	1915	1916	1938	1955	1962	1964	1986	1987	2005	2006
	2021	2022	2023	2024	2046	2051	2060	2065	2070	2072	2076	2111	2119	2120	2121
	2127	2134	2152	2153	2154	2155	2156	2169	2170	2171	2172	2173	2174	2175	2176
	2177	2178	2179	2185	2186	2190	2191	2192	2193	2194	2195	2196	2197	2198	2201
	2238	2239	2244	2252	2267	2306	2307	2314	2318	2323	2324	2326	2328	2338	2344
MOVB	2345	2365	2366	2370											
	598	831	846	847	1442	1822	1826	1852	1860	1900	1931	1990	2010	2015	2112
	2113	2116	2117	2118	2122	2125	2126	2145	2249	2277	2285	2301	2302	2304	2368
NEG	2123														
NOP	795	796	818	1073	1074	1278	1279	1280	1281	1295	1296	1303	1312	1313	1324
	1325	1333	1356	1363	1372	1373	1384	1385	1397	1398	1399	1401	1483	1484	1496
	1497	1498	1500	1537	1538	1539	1540	1556	1557	1558	1562	1563	1564	1647	1648
	1649														
RESET	579	672	704	933	1269	1517	1582	1645							
ROL	1903	1905	1907	2129	2131	2132	2133	2135							
RTI	614	1286	1289	1829	1879	1917	1967	1997	2025	2157	2203	2254			
RTS	1613	1759	1764	2074	2294	2346	2371								
SUB	1525	1851	2321												
TRAP	2373	2383	2384	2385	2386	2387	2388	2389							
TST	610	526	629	1790	1814	1865	1871	1911	1953	1960	1970	2078	2140	2251	2259
	2281	2316	2334	2336	2367										
TSTB	1432	1468	1801	1929	1988	2234	2283	2308	2319	2332					
.ASCII	407	408													
.ASCIIZ	409	634	638	642	648	655	1654	1662	1667	1672	1677	1682	1687	1691	1696
	1705	1714	1719	1724	1729	1731	1747	1757	2029	2030	2031	2033	2082	2207	
.BLKB	2028														
.BYTE	375	376	381	382	397	398	399	400	426	427	437	438	445	446	448
	449	451	452	454	455	456	457	1653	1743	1745	1862	1863	2026	2027	2158
	2159	2160	2161	2347	2348	2349									
.ENABL	177														
.END	2390														
.ENDC	182	192	284	298	309	311	312	313	327	334	338	340	346	348	355
	369	373	375	401	405	406	407	411	415	437	445	448	451	454	455
	456	457	460	461	462	463	464	465	466	467	468	469	470	471	472
	473	474	475	476	477	478	479	483	536	574	576	585	586	589	591

E06

MAINDEC-11-DZLPI-B
DZLPIB.P11LPS-11-DRA DIAGNOSTIC
CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

MACY11 27(732) 17-SEP-76 14:21 PAGE 72

	593	595	596	597	599	601	623	634	638	642	648	655	679	680	681
	682	687	688	689	690	695	698	699	700	701	702	707	710	711	712
	713	718	721	722	723	724	729	732	733	734	735	736	742	746	747
	748	749	762	763	764	765	780	781	782	783	800	803	804	805	806
	822	825	826	827	828	829	840	841	842	843	844	854	855	856	857
	864	868	869	870	871	876	879	880	881	882	887	890	891	892	893
	899	903	904	905	906	911	914	915	916	917	923	926	927	928	929
	930	936	941	942	943	944	955	958	959	960	961	968	972	973	974
	975	982	986	987	988	989	996	999	1000	1001	1002	1016	1027	1035	1036
	1037	1038	1052	1060	1061	1062	1063	1082	1089	1090	1091	1092	1158	1161	1162
	1163	1164	1230	1233	1234	1235	1236	1243	1248	1249	1250	1251	1258	1265	1266
	1267	1268	1269	1284	1292	1293	1294	1295	1315	1321	1322	1323	1324	1343	1345
	1353	1354	1355	1356	1375	1381	1382	1383	1384	1389	1404	1413	1414	1415	1416
	1418	1426	1434	1444	1449	1450	1451	1452	1454	1462	1470	1480	1481	1492	1483
	1488	1503	1513	1514	1515	1516	1517	1530	1531	1532	1533	1549	1550	1551	1552
	1570	1578	1579	1580	1581	1582	1592	1593	1594	1595	1622	1624	1625	1627	1629
	1632	1638	1641	1642	1643	1645	1651	1653	1656	1757	1770	1773	1778	1783	1785
	1796	1799	1800	1801	1803	1805	1812	1816	1821	1823	1827	1830	1831	1835	1838
	1846	1850	1855	1856	1857	1865	1875	1879	1880	1884	1886	1919	1923	1979	1998
	1999	2006	2008	2011	2013	2029	2040	2055	2084	2089	2168	2177	2178	2194	2190
	2191	2201	2203	2207	2220	2249	2301	2302	2305	2332	2347	2360	2366	2369	2382
	2383	2384	2385	2386	2387	2388	2389	2390							
.EQUIV	192	193	201	216	217	246	247	248	249	250	251	252	253	254	255
	274	275	276	277	278	279	280	281	282	283					
.EVEN	415	458	634	638	642	648	655	1737	1749	1757	2083	2214	2350		
.IF	178	190	256	284	309	310	311	312	313	324	333	336	338	345	347
	354	368	372	374	401	405	406	407	410	411	414	437	445	448	451
	454	455	456	457	460	461	462	463	464	465	466	467	468	469	470
	471	472	473	474	475	476	477	478	479	483	536	573	575	580	585
	587	589	591	593	595	596	597	599	617	633	637	641	647	654	678
	680	682	686	688	690	694	697	699	701	702	706	709	711	713	717
	720	722	724	728	731	733	735	736	741	745	747	749	761	763	765
	779	781	783	799	802	804	806	821	824	826	828	829	839	841	843
	844	853	855	857	863	867	869	871	875	878	880	882	886	889	891
	893	898	902	904	906	910	913	915	917	922	925	927	929	930	935
	940	942	944	954	957	959	961	967	971	973	975	981	985	987	989
	995	998	1000	1002	1015	1026	1034	1036	1038	1051	1059	1061	1063	1081	1088
	1090	1092	1157	1160	1162	1164	1229	1232	1234	1236	1242	1247	1249	1251	1257
	1264	1266	1268	1269	1283	1291	1293	1295	1314	1320	1322	1324	1342	1344	1352
	1354	1356	1374	1380	1382	1384	1388	1403	1412	1414	1416	1417	1425	1433	1443
	1448	1450	1452	1453	1461	1469	1479	1481	1483	1487	1502	1512	1514	1516	1517
	1529	1531	1533	1548	1550	1552	1569	1577	1579	1581	1582	1591	1593	1595	1621
	1622	1623	1624	1625	1626	1627	1631	1637	1640	1642	1643	1645	1651	1653	1654
	1656	1756	1769	1772	1777	1782	1783	1795	1797	1798	1799	1801	1802	1803	1812
	1814	1822	1824	1829	1830	1831	1834	1837	1846	1849	1853	1855	1856	1858	1865
	1868	1875	1879	1880	1883	1886	1898	1922	1923	1978	1998	2006	2007	2011	2012
	2028	2029	2039	2054	2070	2088	2167	2177	2178	2183	2190	2191	2199	2201	2203
	2207	2219	2240	2300	2302	2305	2332	2347	2359	2365	2369	2373	2383	2384	2385
	2386	2387	2388	2389	2390										
.IFF	190	309	310	312	313	334	338	340	346	348	355	369	372	375	401
	411	415	574	576	585	679	680	681	682	687	688	689	690	695	698
	699	700	701	707	710	711	712	713	718	721	722	723	724	729	732
	733	734	735	741	746	747	748	749	762	763	764	765	780	781	782
	783	800	803	804	805	806	822	825	826	827	828	840	841	843	843
	854	855	856	857	864	868	869	870	871	876	879	880	881	892	897
	890	891	892	893	899	903	904	905	906	911	914	915	916	917	923

	926	927	928	929	936	941	942	943	944	955	958	959	960	961	968
	972	973	974	975	982	986	987	988	989	996	999	1000	1001	1002	1015
	1026	1035	1036	1037	1038	1051	1060	1061	1062	1063	1081	1089	1090	1091	1092
	1158	1161	1162	1163	1164	1230	1233	1234	1235	1236	1243	1248	1249	1250	1251
	1258	1265	1266	1267	1268	1284	1292	1293	1294	1295	1315	1321	1322	1323	1324
	1343	1345	1353	1354	1355	1356	1375	1381	1382	1383	1384	1399	1404	1413	1414
	1415	1416	1418	1425	1433	1443	1449	1450	1451	1452	1454	1461	1469	1480	1481
	1482	1483	1488	1503	1513	1514	1515	1516	1517	1530	1531	1532	1533	1549	1550
	1551	1552	1570	1578	1579	1580	1581	1582	1592	1593	1594	1595	1622	1626	1632
	1637	1640	1653	1770	1796	1799	1800	1803	1830	1831	1835	1837	1850	1875	1880
	1884	1923	1945	1979	1981	1986	1998	1999	2008	2012	2029	2040	2055	2084	2089
	2168	2184	2199	2220	2301	2360	2366								
.IFT	634	638	642	648	655	1757	1811	1856	1902	1918	1919	1923	1945	1981	1986
.IFTF	634	638	642	648	655	1757	1809	1855	1898	1902	1918	1937	1951	1979	1982
.IIF	177	182	187	306	307	308	309	312	313	321	410	415	586	589	595
	596	597	599	600	616	1624	1625	1632	1633	1653	1656	1773	1774	1775	1776
	1777	1778	1782	1810	1811	1827	1830	1831	1838	1839	1840	1841	1846	1868	1875
	1880	1923	1939	2021	2029	2035	2052	2077	2296	2382	2383	2384	2385	2386	2387
.IRP	2388	2389													
	536	678	686	697	709	720	731	745	761	779	802	824	839	853	867
	878	889	902	913	925	940	957	971	985	998	1034	1059	1088	1160	1232
	1247	1264	1291	1320	1352	1380	1412	1448	1479	1512	1529	1548	1577	1591	1782
.LIST	1893	1914	2171	2177	2190	2191	2306	2307	2328	2344	2345				
	177	298	312	321	401	403	404	405	411	415	536	601	634	638	642
	648	655	678	682	686	690	697	701	709	713	720	724	731	735	745
	749	761	765	779	783	802	806	824	828	839	843	853	857	867	871
	878	882	889	893	902	906	913	917	925	929	940	944	957	961	971
	975	985	989	998	1002	1034	1038	1059	1063	1088	1092	1160	1164	1232	1236
	1247	1251	1264	1268	1291	1295	1320	1324	1352	1356	1380	1384	1412	1416	1448
	1452	1479	1483	1512	1516	1529	1533	1548	1552	1577	1581	1591	1595	1632	1645
	1757	1777	1875	1998	2373	2382	2383	2384	2385	2386	2387	2388	2389	2390	
.MACRO	313	365	573	617	939	940	2373								
.MCALL	177	298	411	601											
.NLIST	177	298	312	321	401	403	404	405	411	415	536	601	634	638	642
	648	655	678	682	686	690	697	701	709	713	720	724	731	735	745
	749	761	765	779	783	802	806	824	828	839	843	853	857	867	871
	878	882	889	893	902	906	913	917	925	929	940	944	957	961	971
	975	985	989	998	1002	1034	1038	1059	1063	1088	1092	1160	1164	1232	1236
	1247	1251	1264	1268	1291	1295	1320	1324	1352	1356	1380	1384	1412	1416	1448
	1452	1479	1483	1512	1516	1529	1533	1548	1552	1577	1581	1591	1595	1632	1645
	1757	1777	1875	1998	2373	2382	2383	2384	2385	2386	2387	2388	2389	2390	
.PAGE	301	365	410												
.REM	1														
.REPT	321	403	785	808	1099	1171									
.SBTTL	188	302	315	325	331	343	366	412	484	678	686	697	709	720	731
	745	761	779	802	824	839	853	867	878	889	902	913	925	940	957
	971	985	998	1034	1059	1088	1160	1232	1247	1264	1291	1320	1352	1380	1412
	1448	1479	1512	1529	1548	1577	1591	1619	1750	1767	1832	1881	1920	2037	2086
	2165	2217	2298	2357	2374										
.TITLE	177														
.WORD	321	322	323	339	359	360	361	362	363	364	374	377	378	379	380
	383	384	385	386	387	388	389	390	391	392	401	403	404	417	418
	419	420	421	422	423	424	428	429	430	443	447	450	453	459	460
	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475
	476	477	478	1637	1640	1652	1918	2063	2068	2162	2200	2202	2246	2293	2330

G06

MAINDEC-11-DZLPI-B LPS-11-DRA DIAGNOSTIC MACY11 27(732) 17-SEP-76 14:21 PAGE 74
DZLPB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*.DZLPB.SEQ/SOL/CRF/DS:ERFZ=DZLPB.P11
RUN-TIME: 57 29 6 SECONDS
RUN-TIME RATIO: 224/93=2.3
CORE USED: 24K (47 PAGES)

Spooler runtime 10 Seconds, 307 KCS, 289 disk reads, 4 disk writes, 71 pages

~~SHOOTER FORTY TWO SECONDS OF RDS, EIGHT SHOTS, FOUR HITS~~

[illegible]