

LK11A

PUSH BUTTON MOD DIAG
MD-11-DZLKA-A

EP-DZLKA-A-DL-A

NOV 1976

COPYRIGHT © 1976

digital

FICHE 1 OF 1

MADE IN USA

The microfiche card contains a grid of frames. The first column of frames on the left contains technical diagrams, including circuit schematics and component layouts. The subsequent columns contain data tables and lists, likely providing component specifications, part numbers, and assembly instructions. The text within the frames is small and difficult to read due to the resolution of the scan.

.REM

%

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZLKA-A

PRODUCT NAME: LK11A PUSH BUTTON MODULE DIAGNOSTIC

DATE: AUG. 21, 1976

MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1976
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

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

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

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

1.0 ABSTRACT

THIS PROGRAM CONSISTS OF THREE SECTIONS. THE FIRST IS A SET OF 25 TESTS WHICH CHECK THE LOGIC OF THE PUSH BUTTON MODULE. THE SECOND SECTION IS ONE TEST WHICH CHECKS THE LAMP DATA REGISTER AND REQUIRES OPERATOR INTERVENTION. THE THIRD SECTION IS ONE TEST WHICH CHECKS THE ENCODER AND ALSO REQUIRES OPERATOR INTERVENTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 WITH 4K OF MEMORY
TELETYPE
LK11A MODULE
CABLE AND PUSH BUTTON BOX NEEDED FOR TESTS AT STARTING ADDRESSES 204 AND 210.

2.2 STORAGE

THIS PROGRAM USES 4K OF MEMORY.

3.0 LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TYPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW15 = 1 HALT ON ERROR
SW14 = 1 LOOP ON TEST
SW13 = 1 INHIBIT ERROR TYPEOUTS
SW12 = 1 HALT AT END OF LOGIC TESTS
SW11 = 1 INHIBIT ITERATIONS
SW10 = 1 BELL ON ERROR
SW9 = 1 LOOP ON ERROR
SW8 = 1 LOOP ON TEST IN SWR<7:0>

40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87

4.2 STARTING ADDRESS

200 IS STARTING ADDRESS OF THE 25 LOGIC TESTS.
204 IS STARTING ADDRESS OF THE MANUAL INTERVENTION TEST.
210 IS STARTING ADDRESS OF THE ENCODER TEST.

5.0 OPERATING PROCEDURE

THE OPERATOR MUST RUN THE 25 LOGIC TESTS BEFORE RUNNING EITHER OF THE TWO OPERATOR INTERVENTION TESTS.

WHEN STARTING AT 200, THE TEST RUNS WITHOUT OPERATOR INTERVENTION AND PRINTS AN END-OF-PASS MESSAGE WHEN COMPLETED, OR IT WILL HALT IF SW12 IS UP.

WHEN STARTING AT 204, THE OPERATOR IS REQUIRED TO PRESS ONE PUSH BUTTON AT A TIME. THE LIGHT ON THAT BUTTON ONLY SHOULD COMPLEMENT. NO CHANGE SHOULD OCCUR IN ANY OF THE OTHER LIGHTS.

WHEN STARTING AT 210, THE OPERATOR IS REQUIRED TO PRESS ONE PUSH BUTTON AT A TIME. THE LIGHT ON THAT BUTTON ONLY SHOULD TURN ON AND ALL OTHER LIGHTS SHOULD GO OFF.

6.0 ERRORS

THIS PROGRAM USES THE DIAGNOSTIC "SYSMAC" PACKAGE FOR ERROR REPORTING AND TYPEOUT. THE ERROR INFORMATION CONSISTS OF A DESCRIPTIVE MESSAGE ABOUT THE ERROR, THE LOCATION AT WHICH THE ERROR WAS DETECTED, THE ACTUAL DATA VALUE, AND THE EXPECTED DATA VALUE.

7.0 RESTRICTIONS

7.1 THE LOGIC TESTS MUST BE RUN AT LEAST ONE TIME BEFORE RUNNING THE TWO MANUAL INTERVENTION TESTS.

7.2 THIS PROGRAM SUPPORTS ONLY ONE LK11A.

88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

EXECUTION TIME FOR ONE PASS THROUGH THE LOGIC TESTS IS 30 SECONDS.

8.2 DEVICE ADDRESSES LOCATIONS

LOCATION "\$BASE" CONTAINS THE LK11A BASE DEVICE ADDRESS <160060>.
LOCATION "\$VECT1" CONTAINS THE LK11A INTERRUPT VECTOR <360>.
LOCATION "\$VECT1" CONTAINS THE LK11A INTERRUPT LEVEL <200>.

8.3 ACT/XXDP/APT NOTES

THE PROGRAM IS CHAINABLE UNDER "ACT-11" AND "XXDP".

THE SOFTWARE HOOKS FOR "APT" HAVE BEEN PROVIDED BUT HAVE NOT BEEN TESTED.

9.0 PROGRAM DESCRIPTION

9.1 LOGIC TESTS

THERE ARE 25 SEPARATE TESTS WHICH CHECK THE LOGIC OF THE PUSH-BUTTON MODULE. THEY RUN SEQUENTIALLY WITHOUT OPERATOR INTERVENTION AND AN END-OF-PASS MESSAGE IS PRINTED AFTER THE COMPLETION OF EACH PASS, OR A HALT WILL OCCUR IF SW12 IS UP.

9.2 PUSH-BUTTON TEST

THIS TEST REQUIRES OPERATOR INTERVENTION. ITS STARTING ADDRESS IS 204 AND THE TEST MUST BE RUN ONLY AFTER THE LOGIC TESTS HAVE BEEN COMPLETED. THE TEST ITSELF DEPENDS UPON THE OPERATOR TO PRESS A SINGLE BUTTON AND TAKE NOTICE OF THE RESULTS. THE LIGHT WITHIN THE BUTTON THAT WAS PRESSED SHOULD COMPLEMENT; IF IT WAS ON, IT SHOULD GO OFF AND IF IT WAS OFF, IT SHOULD GO ON. NO CHANGE SHOULD OCCUR IN ANY OF THE OTHER 15 LIGHTS.

9.3 ENCODER TEST

THIS TEST REQUIRES OPERATOR INTERVENTION AND ITS STARTING ADDRESS IS 210. THE TEST MUST ONLY BE RUN AFTER THE LOGIC TESTS HAVE BEEN RUN. THE OPERATOR MUST PUSH ONE BUTTON AND THE LIGHT WITHIN THAT BUTTON ONLY SHOULD BE TURNED ON AND WON'T GO OFF UNTIL THE TEST IS RESTARTED.

%
.TITLE MAINDEC-11-DZLKA-A
;#COPYRIGHT (C) 1976
;#DIGITAL EQUIPMENT CORP.
;#MAYNARD, MASS. 01754

131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
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
176
177
178
179
180
181
182
183
184
185
186

```

187      ;*
188      ;*PROGRAM BY VERA BREUER
189      ;*
190      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
191      ;*PACKAGE (MAINDEC-11-DZQAC-CD),MAR 21, 1976.
192      ;*
193      .SBTTL  BASIC DEFINITIONS
194
195      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
196      001100  STACK= 1100
197      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
198      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
199
200      ;*MISCELLANEOUS DEFINITIONS
201      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
202      000012  LF= 12          ;;CODE FOR LINE FEED
203      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
204      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
205      177776  PS= 177776      ;;PROCESSOR STATUS WORD
206      .EQUIV  PS,PSW
207      177774  STKLM= 177774    ;;STACK LIMIT REGISTER
208      177772  PIR= 177772     ;;PROGRAM INTERRUPT REQUEST REGISTER
209      177570  DSW= 177570     ;;HARDWARE SWITCH REGISTER
210      177570  DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
211
212      ;*GENERAL PURPOSE REGISTER DEFINITIONS
213      000000  R0= %0          ;;GENERAL REGISTER
214      000001  R1= %1          ;;GENERAL REGISTER
215      000002  R2= %2          ;;GENERAL REGISTER
216      000003  R3= %3          ;;GENERAL REGISTER
217      000004  R4= %4          ;;GENERAL REGISTER
218      000005  R5= %5          ;;GENERAL REGISTER
219      000006  R6= %6          ;;GENERAL REGISTER
220      000007  R7= %7          ;;GENERAL REGISTER
221      .EQUIV  R6,SP          ;;STACK POINTER
222      .EQUIV  R7,PC          ;;PROGRAM COUNTER
223
224      ;*PRIORITY LEVEL DEFINITIONS
225      000000  PR0= 0          ;;PRIORITY LEVEL 0
226      000040  PR1= 40         ;;PRIORITY LEVEL 1
227      000100  PR2= 100        ;;PRIORITY LEVEL 2
228      000140  PR3= 140        ;;PRIORITY LEVEL 3
229      000200  PR4= 200        ;;PRIORITY LEVEL 4
230      000240  PR5= 240        ;;PRIORITY LEVEL 5
231      000300  PR6= 300        ;;PRIORITY LEVEL 6
232      000340  PR7= 340        ;;PRIORITY LEVEL 7
233
234      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
235      100000  SW15= 100000
236      040000  SW14= 40000
237      020000  SW13= 20000
238      010000  SW12= 10000
239      004000  SW11= 4000
240      002000  SW10= 2000
241      001000  SW09= 1000
242      000400  SW08= 400

```


243	000200	SW07=	200
244	000100	SW06=	100
245	000040	SW05=	40
246	000020	SW04=	20
247	000010	SW03=	10
248	000004	SW02=	4
249	000002	SW01=	2
250	000001	SW00=	1
251		.EQUIV	SW09, SW9
252		.EQUIV	SW08, SW8
253		.EQUIV	SW07, SW7
254		.EQUIV	SW06, SW6
255		.EQUIV	SW05, SW5
256		.EQUIV	SW04, SW4
257		.EQUIV	SW03, SW3
258		.EQUIV	SW02, SW2
259		.EQUIV	SW01, SW1
260		.EQUIV	SW00, SW0
261			
262		:#DATA	BIT DEFINITIONS (BIT00 TO BIT15)
263	100000	BIT15=	100000
264	040000	BIT14=	40000
265	020000	BIT13=	20000
266	010000	BIT12=	10000
267	004000	BIT11=	4000
268	002000	BIT10=	2000
269	001000	BIT09=	1000
270	000400	BIT08=	400
271	000200	BIT07=	200
272	000100	BIT06=	100
273	000040	BIT05=	40
274	000020	BIT04=	20
275	000010	BIT03=	10
276	000004	BIT02=	4
277	000002	BIT01=	2
278	000001	BIT00=	1
279		.EQUIV	BIT09, BIT9
280		.EQUIV	BIT08, BIT8
281		.EQUIV	BIT07, BIT7
282		.EQUIV	BIT06, BIT6
283		.EQUIV	BIT05, BIT5
284		.EQUIV	BIT04, BIT4
285		.EQUIV	BIT03, BIT3
286		.EQUIV	BIT02, BIT2
287		.EQUIV	BIT01, BIT1
288		.EQUIV	BIT00, BIT0
289			
290		:#BASIC	"CPU" TRAP VECTOR ADDRESSES
291	000004	ERRVEC=	4
292	000010	RESVEC=	10
293	000014	TBITVEC=	14
294	000014	TRTVEC=	14
295	000014	BPTVEC=	14
296	000020	IOTVEC=	20
297	000024	PWRVEC=	24
298	000030	EMTVEC=	30

```

;; TIME OUT AND OTHER ERRORS
;; RESERVED AND ILLEGAL INSTRUCTIONS
;; "1" BIT
;; TRACE TRAP
;; BREAKPOINT TRAP (BPT)
;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
;; POWER FAIL
;; EMULATOR TRAP (EMT) **ERROR**

```

H01

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 8
DZLKAA.P11 BASIC DEFINITIONS

299
300
301
302
303
304
305

000034
000060
000064
000240
160060
100360
000200

TRAPVEC=34
TKVEC= 60
TPVEC= 64
PIRQVEC=240
ABASE=160060
AVECT1=100360
APRIOR=200

:: "TRAP" TRAP
:: TTY KEYBOARD VECTOR
:: TTY PRINTER VECTOR
:: PROGRAM INTERRUPT REQUEST VECTOR

306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331

000000

000174

000174

000000

000176

000000

000200

000137

001620

000204

000137

004654

000210

000137

004710

.SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
12	HALT AT END OF DIAGNOSTIC
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

.SBTTL TRAP CATCHER

```

.=0
; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
.=174
DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0      ;; SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP      J#BEGIN ;; JUMP TO STARTING ADDRESS OF PROGRAM
JMP      BEGIN1  ; TEST PBR TURNS LDR LIGHT ON AND OFF
JMP      BEGIN2  ; TEST ENCODER WITH PUSH BUTTON
    
```

```

332
333
334
335
336
337      000214
338      000046
339 000046 004616
340      000052
341 000052 000000
342      000214
343      001000
344
345
346
347
348
349      001000
350      000024
351 000024 000200
352      000044
353 000044 001000
354      001000
355
356
357
358
359 001000
360 001000 000000
361 001002 001202
362 001004 000074
363 001006 000170
364 001010 000264
365 001012 000031

.SBTTL ACT11 HOOKS
;*****
;HOOKS REQUIRED BY ACT11
      $SVPC=.          ;SAVE PC
      .-46
      $ENDAD           ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
      .-52
      .WORD 0          ;;2)SET LOC.52 TO ZERO
      .-$SVPC          ;; RESTORE PC
      .-1000
.SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
      .SX=.            ;SAVE CURRENT LOCATION
      .-24             ;SET POWER FAIL TO POINT TO START OF PROGRAM
      200              ;FOR APT START UP
      .-44             ;POINT TO APT INDIRECT ADDRESS PNTR.
      $APTHDR          ;POINT TO APT HEADER BLOCK
      .-$X             ;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

$APTHD:
$SHIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBAADR: .WORD $MAIL  ;; ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM:   .WORD 60     ;; RUN TIM OF LONGEST TEST
$PASTM:  .WORD 120    ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM:  .WORD 180    ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
      .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

```


K01

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 11
 DZLKA.P11 COMMON TAGS

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 000
 391 001135 000
 392 001136 000000
 393 001140 177570
 394 001142 177570
 395 001144 177560
 396 001146 177562
 397 001150 177564
 398 001152 177566
 399 001154 000
 400 001155 002
 401 001156 012
 402 001157 000
 403 001160 000000
 404
 405 001162 000000
 406 001164 000000
 407 001166 000000
 408 001170 000000
 409 001172 177607 000377
 410 001176 077
 411 001177 015
 412 001200 000012
 413
 414
 415
 416
 417
 418 001202
 419 001202 000000
 420 001204 000000
 421 001206 000000

.SBTTL COMMON TAGS

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

SCMTAG: .=1100

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
 SBDADR: .WORD 0
 SGDDAT: .WORD 0
 SBDDAT: .WORD 0
 SAUTOB: .WORD 0
 SINTAG: .BYTE 0
 SWR: .WORD 0
 DISPLAY: .WORD 0
 STKS: 177560
 STKB: 177562
 STPS: 177564
 STPB: 177566
 SNULL: .BYTE 0
 SFILLS: .BYTE 2
 SFILLC: .BYTE 12
 STPFLG: .BYTE 0
 SREGAD: .WORD 0

SREGO: .WORD 0
 SREG1: .WORD 0
 STIMES: 0
 SESCPE: 0
 SBELL: .ASCIZ <207><377><377>
 SQUES: .ASCII /?/
 SCRLF: .ASCII <15>
 SLF: .ASCIZ <12>

.SBTTL APT MAILBOX-ETABLE

 .EVEN
 SMAIL: .WORD
 SMSGTY: .WORD AMSGTY
 SFATAL: .WORD AFATAL
 STESTN: .WORD ATESTN

;; START OF COMMON TAGS

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

L01

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 12
 DZLKA.P11 APT MAILBOX-ETABLE

422	001210	000000	\$PASS:	.WORD	APASS	::PASS COUNT
423	001212	000000	\$DEVCT:	.WORD	ADEVCT	:::DEVICE COUNT
424	001214	000000	\$UNIT:	.WORD	AUNIT	:::I/O UNIT NUMBER
425	001216	000000	\$MSGAD:	.WORD	AMSGAD	:::MESSAGE ADDRESS
426	001220	000000	\$MSGLG:	.WORD	AMSGLG	:::MESSAGE LENGTH
427	001222		\$ETABLE:			:::APT ENVIRONMENT TABLE
428	001222	000	\$ENV:	.BYTE	AENV	:::ENVIRONMENT BYTE
429	001223	000	\$ENVM:	.BYTE	AENVM	:::ENVIRONMENT MODE BITS
430	001224	000000	\$SWREG:	.WORD	ASWREG	:::APT SWITCH REGISTER
431	001226	000000	\$USWR:	.WORD	AUSWR	:::USER SWITCHES
432	001230	000000	\$CPUOP:	.WORD	ACPUOP	:::CPU TYPE, OPTIONS
433			:::			BITS 15-11=CPU TYPE
434			:::			11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
435			:::			11/70=06, PDQ=07, Q=10
436			:::			BIT 10=REAL TIME CLOCK
437			:::			BIT 9=FLOATING POINT PROCESSOR
438			:::			BIT 8=MEMORY MANAGEMENT
439	001232	000	\$MAMS1:	.BYTE	AMAMS1	:::HIGH ADDRESS, M.S. BYTE
440	001233	000	\$MTYP1:	.BYTE	AMTYP1	:::MEM. TYPE, BLK#1
441			:::			MEM. TYPE BYTE -- (HIGH BYTE)
442			:::			900 NSEC CORE=001
443			:::			300 NSEC BIPOLAR=002
444			:::			500 NSEC MOS=003
445	001234	000000	\$MADR1:	.WORD	AMADR1	:::HIGH ADDRESS, BLK#1
446			:::			MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
447	001236	000	\$MAMS2:	.BYTE	AMAMS2	:::HIGH ADDRESS, M.S. BYTE
448	001237	000	\$MTYP2:	.BYTE	AMTYP2	:::MEM. TYPE, BLK#2
449	001240	000000	\$MADR2:	.WORD	AMADR2	:::MEM. LAST ADDRESS, BLK#2
450	001242	000	\$MAMS3:	.BYTE	AMAMS3	:::HIGH ADDRESS, M.S. BYTE
451	001243	000	\$MTYP3:	.BYTE	AMTYP3	:::MEM. TYPE, BLK#3
452	001244	000000	\$MADR3:	.WORD	AMADR3	:::MEM. LAST ADDRESS, BLK#3
453	001246	000	\$MAMS4:	.BYTE	AMAMS4	:::HIGH ADDRESS, M.S. BYTE
454	001247	000	\$MTYP4:	.BYTE	AMTYP4	:::MEM. TYPE, BLK#4
455	001250	000000	\$MADR4:	.WORD	AMADR4	:::MEM. LAST ADDRESS, BLK#4
456	001252	100360	\$VECT1:	.WORD	AVECT1	:::INTERRUPT VECTOR#1, BUS PRIORITY#1
457	001254	000000	\$VECT2:	.WORD	AVECT2	:::INTERRUPT VECTOR#2, BUS PRIORITY#2
458	001256	160060	\$BASE:	.WORD	ABASE	:::BASE ADDRESS OF EQUIPMENT UNDER TEST
459	001260	000000	\$DEVN:	.WORD	ADEVN	:::DEVICE MAP
460	001262	000000	\$CDW1:	.WORD	ACDW1	:::CONTROLLER DESCRIPTION WORD#1
461	001264		\$ETEND:			
462			.MEXIT			

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;	ITEM	1		
		EM1		;FAILED TO LOAD AND READ LDR
		DH1		;ERRPC LDR EXPECTED ACTUAL
		DT1		;SERRPC, LDR, \$GDDAT, \$BDDAT, 0
		DF1		
;	ITEM	2		
		EM2		;FAILED TO SET OR CLEAR PBR
		DH2		;ERRPC PBR EXPECTED ACTUAL
		DT2		;SERRPC, PBR, \$GDDAT, \$BDDAT, 0
		DF1		
;	ITEM	3		
		EM3		;FAILED TO CLEAR PB FLAG
		DH3		;ERRPC PBSR EXPECTED ACTUAL
		DT3		;SERRPC, PBSR, \$GDDAT, \$BDDAT, 0
		DF1		
;	ITEM	4		
		EM4		;FAILED TO LOAD AND READ PB FLAG
		DH3		;ERRPC PBSR EXPECTED ACTUAL
		DT3		;SERRPC, PBSR, \$GDDAT, \$BDDAT, 0
		DF1		
;	ITEM	5		
		EM5		;FAILED TO CLEAR INTERRUPT ENABLE
		DH3		;ERRPC PBSR EXPECTED ACTUAL
		DT3		;SERRPC, PBSR, \$GDDAT, \$BDDAT
		DF1		
;	ITEM	6		
		EM6		;FAILED TO SET INTERRUPT ENABLE
		DH3		;ERRPC PBSR EXPECTED ACTUAL
		DT3		;SERRPC, PBSR, \$GDDAT, \$BDDAT
		DF1		
;	ITEM	7		
		EM7		;ERRONEOUSLY CLEARED PB FLAG

463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477 001264
 478
 479
 480
 481 001264 005036
 482 001266 006565
 483 001270 006772
 484 001272 007040
 485
 486
 487 001274 005104
 488 001276 006625
 489 001300 007004
 490 001302 007040
 491
 492
 493 001304 005137
 494 001306 006665
 495 001310 007016
 496 001312 007040
 497
 498
 499 001314 005167
 500 001316 006665
 501 001320 007016
 502 001322 007040
 503
 504
 505 001324 005227
 506 001326 006665
 507 001330 007016
 508 001332 007040
 509
 510
 511 001334 005270
 512 001336 006665
 513 001340 007016
 514 001342 007040
 515
 516
 517
 518 001344 005327

519	001346	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
520	001350	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
521	001352	007040	DF1	
522				
523			;ITEM 10	
524	001354	005363	EM10	;FAILED TO GENERATE INTERRUPT WITH PB FLAG
525	001356	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
526	001360	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
527	001362	007040	DF1	
528				
529			;ITEM 11	
530	001364	005435	EM11	;CLR 2PBSR FAILED TO CLEAR PB FLAG
531	001366	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
532	001370	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
533	001372	007040	DF1	
534				
535			;ITEM 12	
536	001374	005476	EM12	;INTERRUPT OK FAILED TO CLEAR INTR FLOP
537	001376	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
538	001400	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
539	001402	007040	DF1	
540				
541			;ITEM 13	
542	001404	005545	EM13	;CLR 2PBSR FAILED TO CLEAR INTERRUPT ENABLE
543	001406	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
544	001410	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
545	001412	007040	DF1	
546				
547			;ITEM 14	
548	001414	005617	EM14	;UNEXPECTED INTERRUPT
549	001416	006726	DH4	;ERRPC PBSR LOCATION OF FAILURE
550	001420	007030	DT4	;SERRPC,PBSR,\$GDDAT
551	001422	007040	DF1	
552				
553			;ITEM 15	
554	001424	005644	EM15	;BIC FAILED TO CLEAR LDR
555	001426	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
556	001430	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
557	001432	007040	DF1	
558				
559			;ITEM 16	
560	001434	005675	EM16	;BIS FAILED TO SET LDR
561	001436	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
562	001440	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
563	001442	007040	DF1	
564				
565			;ITEM 17	
566	001444	005723	EM17	;BISB FAILED TO SET LOWER BYTE
567	001446	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
568	001450	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT
569	001452	007040	DF1	
570				
571			;ITEM 20	
572	001454	006016	EM20	;BISB FAILED TO SET UPPER BYTE
573	001456	006665	DH3	;ERRPC PBSR EXPECTED ACTUAL
574	001460	007016	DT3	;SERRPC,PBSR,\$GDDAT,\$BDDAT

575	001462	007040	DF1	
576				
577			; ITEM	21
578	001464	006111		EM21
579	001466	006665		DH3
580	001470	007016		DT3
581	001472	007040		DF1
582				
583			; ITEM	22
584	001474	006206		EM22
585	001476	006665		DH3
586	001500	007016		DT3
587	001502	007040		DF1
588				
589			; ITEM	23
590	001504	006303		EM23
591	001506	006665		DH3
592	001510	007016		DT3
593	001512	007040		DF1
594				
595			; ITEM	24
596	001514	006341		EM24
597	001516	006565		DH1
598	001520	006772		DT1
599	001522	007040		DF1
600				
601			; ITEM	25
602	001524	006377		EM25
603	001526	006625		DH2
604	001530	006772		DT1
605	001532	007040		DF1
606				
607			; ITEM	26
608	001534	006434		EM26
609	001536	006565		DH1
610	001540	006772		DT1
611	001542	007040		DF1
612				
613			; ITEM	27
614	001544	006471		EM27
615	001546	006625		DH2
616	001550	007004		DT2
617	001552	007040		DF1
618				
619			; ITEM	30
620	001554	006527		EM30
621	001556	006665		DH3
622	001560	007016		DT3
623	001562	007040		DF1

;BICB FAILED TO CLEAR LOWER BYTE
;ERRPC PBSR EXPECTED ACTUAL
;SERRPC,PBSR,\$GDDAT,\$BDDAT

;BICB FAILED TO CLEAR UPPER BYTE
;ERRPC PBSR EXPECTED ACTUAL
;SERRPC,PBSR,\$GDDAT,\$BDDAT

;LOADING LDR CHANGED PBSR DATA
;ERRPC PBSR EXPECTED ACTUAL
;SERRPC,PBSR,\$GDDAT,\$BDDAT

;LOADING PBSR CHANGED LDR DATA
;ERRPC LDR EXPECTED ACTUAL
;SERRPC,LDR,\$GDDAT,\$GDDAT

;LOADING LDR CHANGED PBR DATA
;ERRPC PBR EXPECTED ACTUAL
;SERRPC,PBSR,\$GDDAT,\$BDDAT

;LOADING PBR CHANGED LDR DATA
;ERRPC LDR EXPECTED ACTUAL
;SERRPC,LDR,\$GDDAT,\$BDDAT

;LOADING PBSR CHANGED PBR DATA
;ERRPC PBR EXPECTED ACTUAL
;SERRPC,PBR,\$GDDAT,\$BDDAT

;LOADING PBR CHANGED PBSR DATA
;ERRPC PBSR EXPECTED ACTUAL
;SERRPC,PBSR,\$GDDAT,\$BDDAT

624	001564	160060		PBSR:	ABASE		;PB STATUS REGISTER
625	001566	160062		PBR:	ABASE+2		;PUSH BUTTON REGISTER
626	001570	160064		LDR:	ABASE+4		;LAMP DATA REGISTER
627	001572	100360		VECT:	AECT1		;INTR. VECTOR
628	001574	100362		VECTR1:	AECT1+2		;VECT ADDRESS
629	001576	000000		TEMP:	0		
630	001600	000000		TEMP1:	0		
631	001602	000000		TEMP2:	0		
632	001604	000000		TEMPL:	0		
633	001606	000000		BITSAV:	0		
634							
635	001610	011637	001124	UNEXP:	MOV (SP),SGDDAT		;POP PC
636	001614	104014		ERROR	14		;UNEXPECTED INTR.
637	001616	000002		RTI			;RETURN FROM INTERRUPT
638							
639	001620	000005		BEGIN:	RESET		
640				.SBTTL	INITIALIZE THE COMMON TAGS		
641				;;CLEAR	THE COMMON TAGS (SCMTAG) AREA		
642	001622	012706	001100	MOV	#SCMTAG,R6		;FIRST LOCATION TO BE CLEARED
643	001626	005026		CLR	(R6)+		;CLEAR MEMORY LOCATION
644	001630	022706	001140	CMP	#SWR,R6 ;;DONE?		
645	001634	001374		BNE	-6		;LOOP BACK IF NO
646	001636	012706	001100	MOV	#STACK,SP		;SETUP THE STACK POINTER
647				;;INITIALIZE A FEW VECTORS			
648	001642	012737	010466	MOV	#SSCOPE,#IOTVEC		;IOT VECTOR FOR SCOPE ROUTINE
649	001650	012737	000340	MOV	#340,#IOTVEC+2		;LEVEL 7
650	001656	012737	010750	MOV	#SEERROR,#ENTVEC		;ENT VECTOR FOR ERROR ROUTINE
651	001664	012737	000340	MOV	#340,#ENTVEC+2		;LEVEL 7
652	001672	012737	012262	MOV	#STRAP,#TRAPVEC		;TRAP VECTOR FOR TRAP CALLS
653	001700	012737	000340	MOV	#340,#TRAPVEC+2		;LEVEL 7
654	001706	012737	012330	MOV	#SPWRDN,#PWAVEC		;POWER FAILURE VECTOR
655	001714	012737	000340	MOV	#340,#PWAVEC+2		;LEVEL 7
656	001722	013737	004564	MOV	SENDCT,SEOPCT		;SETUP END-OF-PROGRAM COUNTER
657	001730	005037	001166	CLR	STINES		;INITIALIZE NUMBER OF ITERATIONS
658	001734	005037	001170	CLR	SESCAPE		;CLEAR THE ESCAPE ON ERROR ADDRESS
659	001740	112737	000001	MOVB	#1,SEMAX		;ALLOW ONE ERROR PER TEST
660	001746	012737	001746	MOV	#.,SLPADR		;INITIALIZE THE LOOP ADDRESS FOR SCOPE
661	001754	012737	001754	MOV	#.,SLPERR		;SETUP THE ERROR LOOP ADDRESS
662				;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS			
663				;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.			
664	001762	013746	000004	MOV	#ERRVEC, -(SP)		;SAVE ERROR VECTOR
665	001766	012737	002022	MOV	#64S,#ERRVEC		;SET UP ERROR VECTOR
666	001774	012737	177570	MOV	#DSWR,SWR		;SETUP FOR A HARDWARE SWICH REGISTER
667	002002	012737	177570	MOV	#DDISP,DISPLAY		;AND A HARDWARE DISPLAY REGISTER
668	002010	022777	177777	CMP	#-1,#SWR		;TRY TO REFERENCE HARDWARE SWR
669	002016	001012		BNE	66S		;BRANCH IF NO TIMEOUT TRAP OCCURRED
670							;AND THE HARDWARE SWR IS NOT = -1
671	002020	000403		BR	65S		;BRANCH IF NO TIMEOUT
672	002022	012716	002030	64S:	MOV #65S,(SP)		;SET UP FOR TRAP RETURN
673	002026	000002		RTI			
674	002030	012737	000176	65S:	MOV #SWREG,SWR		;POINT TO SOFTWARE SWR
675	002036	012737	000174	MOV	#DISPREG,DISPLAY		
676	002044	012637	000004	66S:	MOV (SP)+,#ERRVEC		;RESTORE ERROR VECTOR
677							
678	002050	005037	001210	CLR	SPASS		;CLEAR PASS COUNT
679	002054	132737	000200	BITB	#APTSIZE,SENVN		;TEST USER SIZE UNDER APT

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 17
 DZLKA.P11 INITIALIZE THE COMMON TAGS

680	002062	001403			BEQ	675	::YES,USE NON-APT SWITCH
681	002064	012737	001224	001140	MOV	#SSWREG,SWR	::NO,USE APT SWITCH REGISTER
682	002072						
683	002072	005737	000042		675:	TST	
684	002076	001002				BNE	
685	002100	104400				TYPE	:TEST IF XXDP
686	002102	007237				HEAD1	:BRANCH IF YES
687	002104	013737	001256	001564	INIT:	MOV	\$BASE,PBSR
688	002112	013737	001256	001570		MOV	\$BASE,LDR
689	002120	013737	001256	001566		MOV	\$BASE,PBR
690	002126	062737	000002	001566		ADD	#2,PBR
691	002134	062737	000004	001570		ADD	#4,LDR
692	002142	013737	001252	001572		MOV	\$VECT1,VECT
693	002150	013737	001252	001574		MOV	\$VECT1,VECTR1
694	002156	062737	000002	001574		ADD	#2,VECTR1
695	002164	042737	160000	001572		BIC	#160000,VECT
696	002172	042737	160000	001574		BIC	#160000,VECTR1
697	002200	012777	001610	177364		MOV	#UNEXP,VECT
698	002206	005077	177362			CLR	\$VECTR1
699							:INITIAL BASE ADDRESS
							:INITIAL LDR ADDRESS
							:INITIAL PBR ADDRESS
							:CORRECT PBR ADDRESS
							:CORRECT LDR ADDRESS
							:INITIAL VECTERO ADDRESS
							:INITIAL VECTR1 ADDRESS
							:CORRECT VECTR1 ADDRESS
							:CLEAR PRIORITY BITS
							:INTERRUPT ADDRESS

```

700      ;*****
701      ;*TEST 1      LOAD ALL 1'S AND READ BACK LDR
702      ;*****
703      002212 000004
704      002214 012737 177777 001124
705      002222 013777 001124 177340
706      002230 017737 177334 001126
707      002236 023737 001124 001126
708      002244 001401
709      002246 104001
710
711      ;*****
712      ;*TEST 2      LOAD AND READ '25' PATTERN INTO LDR
713      ;*****
714      002250 000004
715      002252 012737 125252 001124
716      002260 013777 001124 177302
717      002266 017737 177276 001126
718      002274 023737 001124 001126
719      002302 001401
720      002304 104001
721
722      ;*****
723      ;*TEST 3      LOAD AND READ '52' PATTERN INTO LDR
724      ;*****
725      002306 000004
726      002310 012737 052525 001124
727      002316 013777 001124 177244
728      002324 017737 177240 001126
729      002332 023737 001124 001126
730      002340 001401
731      002342 104001
732
733      ;*****
734      ;*TEST 4      FLOAT A ONE THROUGH LDR
735      ;*****
736      002344 000004
737      002346 012737 000001 001124
738      002354 013777 001124 177206
739      002362 017737 177202 001126
740      002370 023737 001124 001126
741      002376 001401
742      002400 104001
743      002402 006137 001124
744      002406 103362
745
      TST1:  SCOPE
              MOV      #-1,SGDDAT      ;LOAD EXPECTED DATA
              MOV      SGDDAT,ALDR      ;LOAD ALL ONES INTO LDR
              MOV      ALDR,SBDDAT      ;READ ACTUAL DATA
              CMP      SGDDAT,SBDDAT    ;COMPARE RESULTS
              BEQ      TST2              ;;BRANCH IF EQUAL
              ERROR    1                  ;FAILED TO LOAD ALL ONES

      TST2:  SCOPE
              MOV      #125252,SGDDAT   ;LOAD EXPECTED DATA
              MOV      SGDDAT,ALDR      ;LOAD PATTERN INTO LDR
              MOV      ALDR,SBDDAT      ;READ ACTUAL DATA
              CMP      SGDDAT,SBDDAT    ;COMPARE RESULTS
              BEQ      TST3              ;;BRANCH IF EQUAL
              ERROR    1                  ;FAILED TO LOAD PATTERN

      TST3:  SCOPE
              MOV      #52525,SGDDAT    ;LOAD EXPECTED DATA
              MOV      SGDDAT,ALDR      ;LOAD PATTERN INTO LDR
              MOV      ALDR,SBDDAT      ;READ ACTUAL DATA
              CMP      SGDDAT,SBDDAT    ;COMPARE RESULTS
              BEQ      TST4              ;;BRANCH IF EQUAL
              ERROR    1                  ;FAILED TO LOAD PATTERN

      TST4:  SCOPE
              MOV      #BIT0,SGDDAT     ;LOAD EXPECTED DATA
              MOV      SGDDAT,ALDR      ;LOAD LDR BIT
              MOV      ALDR,SBDDAT      ;READ ACTUAL DATA
              CMP      SGDDAT,SBDDAT    ;COMPARE RESULTS
              BEQ      1S                ;;BRANCH IF EQUAL
              ERROR    1                  ;FAILED TO SET LDR BIT
      2S:      MOV      SGDDAT,ALDR      ;TEST NEXT BIT
              CMP      SGDDAT,SBDDAT    ;CONTINUE THROUGH LDR
              BEQ      1S
              BCC      2S
      1S:      ROL      SGDDAT
              BCC      2S

```



```

746                                     ;*****
747                                     ;*TEST 5      FLOAT ZERO THROUGH LDR
748                                     ;*****
749 002410 000004                                     †T5:  SCOPE
750 002412 012737 177776 001124 25:  MOV      #177776,$GDDAT      ;LOAD EXPECTED DATA
751 002420 013777 001124 177142 25:  MOV      $GDDAT,$LDR      ;LOAD LDR BIT
752 002426 017737 177136 001126 25:  MOV      $LDR,$BDDAT      ;READ ACTUAL DATA
753 002434 023737 001124 001126 25:  CMP      $GDDAT,$BDDAT      ;COMPARE RESULTS
754 002442 001401 15:  BEQ      15      ;;BRANCH IF EQUAL
755 002444 104001 15:  ERROR      1      ;FAILED TO CLEAR LDR BIT
756 002446 000261 15:  SEC      ;SET CARRY BIT
757 002450 006137 001124 15:  ROL      $GDDAT      ;TEST NEXT BIT
758 002454 103361 15:  BCC      25      ;CONTINUE THROUGH LDR
759
760                                     ;*****
761                                     ;*TEST 6      LOAD AND READ PBR USING BIT 0
762                                     ;*****
763 002456 000004                                     †T6:  SCOPE
764 002460 012737 177777 001124 25:  MOV      #-1,$GDDAT      ;LOAD EXPECTED DATA
765 002466 012777 000001 177072 25:  MOV      $BIT0,$PBR      ;SET BIT 0
766 002474 017737 177066 001126 25:  MOV      $PBR,$BDDAT      ;READ ACTUAL DATA
767 002502 023737 001124 001126 25:  CMP      $GDDAT,$BDDAT      ;COMPARE RESULTS
768 002510 001401 25:  BEQ      TST7      ;;BRANCH IF EQUAL
769 002512 104002 25:  ERROR      2      ;FAILED TO SET PBR
770
771                                     ;*****
772                                     ;*TEST 7      CLEAR PBR USING BIT 0
773                                     ;*****
774 002514 000004                                     †T7:  SCOPE
775 002516 005037 001124 25:  CLR      $GDDAT      ;EXPECTED DATA
776 002522 005077 177040 25:  CLR      $PBR      ;CLEAR BIT 0
777 002526 017737 177034 001126 25:  MOV      $PBR,$BDDAT      ;READ ACTUAL DATA
778 002534 023737 001124 001126 25:  CMP      $GDDAT,$BDDAT      ;COMPARE RESULTS
779 002542 001401 25:  BEQ      TST10      ;;BRANCH IF EQUAL
780 002544 104002 25:  ERROR      2      ;FAILED TO CLEAR PBR
781
782                                     ;*****
783                                     ;*TEST 10     LOAD AND READ BACK PB FLAG
784                                     ;*****
785 002546 000004                                     †T10: SCOPE
786 002550 005077 177010 25:  CLR      $PBSR      ;CLEAR STATUS REGISTER
787 002554 012737 100000 001124 25:  MOV      $BIT15,$GDDAT      ;LOAD EXPECTED DATA
788 002562 013777 001124 176774 25:  MOV      $GDDAT,$PBSR      ;LOAD STATUS REG.
789 002570 017737 176770 001126 25:  MOV      $PBSR,$BDDAT      ;READ ACTUAL DATA
790 002576 023737 001124 001126 25:  CMP      $GDDAT,$BDDAT      ;COMPARE RESULTS
791 002604 001401 25:  BEQ      TST11      ;;BRANCH IF EQUAL
792 002606 104004 25:  ERROR      4      ;FAILED TO SET BIT 15
793

```

```

794
795
796
797 002610 000004
798 002612 005077 176746
799 002616 005037 001124
800 002622 052777 100000 176734
801 002630 005077 176730
802 002634 017737 176724 001126
803 002642 023737 001124 001126
804 002650 001401
805 002652 104011
806
807
808
809
810 002654 000004
811 002656 012737 000100 001166
812 002664 005077 176674
813 002670 005037 001124
814 002674 012777 100000 176662
815 002702 000005
816 002704 017737 176654 001126
817 002712 023737 001124 001126
818 002720 001401
819 002722 104003
820
821
822
823
824 002724 000004
825 002726 005077 176632
826 002732 005037 001124
827 002736 012777 100000 176620
828 002744 017737 176616 001576
829 002752 017737 176606 001126
830 002760 023737 001124 001126
831 002766 001401
832 002770 104003
833
834
835
836
837 002772 000004
838 002774 005077 176564
839 003000 005037 001124
840 003004 012777 100000 176552
841 003012 017737 176546 001576
842 003020 017737 176540 001126
843 003026 023737 001124 001126
844 003034 001401
845 003036 104003

;*****
;TEST 11 TEST PB FLAG CLEARED BY CLR PBSR
;*****
TST11: SCOPE
CLR 2PBSR ;CLEAR STATUS REGISTER
CLR SGDDAT ;EXPECTED DATA
BIS #BIT15,2PBSR ;SET PB FLAG
CLR 2PBSR ;CLEAR STATUS REGISTER
MOV 2PBSR,$BDDAT ;READ ACTUAL DATA
CMP SGDDAT,$BDDAT ;COMPARE RESULTS
BEQ TST12 ;;BRANCH IF EQUAL
ERROR 11 ;FAILED TO CLEAR PB FLAG

;*****
;TEST 12 TEST INIT CLEARS PB FLAG
;*****
TST12: SCOPE
MOV #100,$TIMES ;;DO 100 ITERATIONS
CLR 2PBSR ;CLEAR STATUS REG.
CLR SGDDAT ;LOAD EXPECTED DATA
MOV #BIT15,2PBSR ;SET PB FLAG
RESET ;INITIALIZE
MOV 2PBSR,$BDDAT ;READ ACTUAL DATA
CMP SGDDAT,$BDDAT ;COMPARE RESULTS
BEQ TST13 ;;BRANCH IF EQUAL
ERROR 3 ;FAILED TO CLEAR PB FLAG

;*****
;TEST 13 TEST PB FLAG CLEARED WHEN READ PBR H
;*****
TST13: SCOPE
CLR 2PBSR ;CLEAR STATUS REGISTER
CLR SGDDAT ;LOAD EXPECTED DATA
MOV #BIT15,2PBSR ;SET PB FLAG
MOV 2PBR,TEMP ;READ PB REGISTER
MOV 2PBSR,$BDDAT ;READ ACTUAL DATA
CMP SGDDAT,$BDDAT ;COMPARE RESULTS
BEQ TST14 ;;BRANCH IF EQUAL
ERROR 3 ;FAILED TO CLEAR PB FLAG

;*****
;TEST 14 TEST PB FLAG IS CLEARED BY READ STATUS WHEN INTR. ENABLE IS CLEARED
;*****
TST14: SCOPE
CLR 2PBSR ;CLEAR STATUS REG
CLR SGDDAT ;EXPECTED DATA
MOV #BIT15,2PBSR ;SET PB FLAG
MOV 2PBSR,TEMP ;READ SR
MOV 2PBSR,$BDDAT ;READ ACTUAL DATA
CMP SGDDAT,$BDDAT ;COMPARE RESULTS
BEQ TST15 ;;BRANCH IF EQUAL
ERROR 3 ;READ SR FAILED TO CLEAR PB FLAG

```



```

846                                     ;*****
847                                     ;*TEST 15      LOAD AND READ INTERRUPT ENABLE BIT, ALSO UNEXPECTED INTRPT TEST
848                                     ;*****
849 003040 000004                       TST15: SCOPE
850 003042 005077 176516                CLR      2PBSR                      ;CLEAR PB STATUS REGISTER
851 003046 012737 040000 001124        MOV      2BIT14,SGDDAT          ;LOAD EXPECTED DATA
852 003054 013777 001124 176502        MOV      SGDDAT,2PBSR          ;LOAD STATUS REGISTER
853 003062 017737 176476 001126        MOV      2PBSR,$BDDAT          ;READ ACTUAL DATA
854 003070 023737 001124 001126        CMP      SGDDAT,$BDDAT        ;COMPARE RESULTS
855 003076 001401                       BEQ      TST16                  ;;BRANCH IF EQUAL
856 003100 104006                       ERROR 6                          ;FAILED TO LOAD INTERRUPT ENABLE
857                                     ;*****
858                                     ;*TEST 16      TEST INTERRUPT ENABLE CLEARED BY INIT
859                                     ;*****
860 003102 000004                       TST16: SCOPE
861 003104 012737 000100 001166        MOV      2100,2TIMES          ;;DO 100 ITERATIONS
862 003112 005037 001124                CLR      SGDDAT              ;LOAD EXPECTED DATA
863 003116 005077 176442                CLR      2PBSR              ;ENSURE CLEAR REG
864 003122 012777 040000 176434        MOV      2BIT14,2PBSR          ;SET BIT 14
865 003130 000005                       RESET                          ;INITIALIZE
866 003132 017737 176426 001126        MOV      2PBSR,$BDDAT          ;READ ACTUAL DATA
867 003140 023737 001124 001126        CMP      SGDDAT,$BDDAT        ;COMPARE RESULTS
868 003146 001401                       BEQ      TST17                  ;;BRANCH IF EQUAL
869 003150 104005                       ERROR 5                          ;FAILED TO CLEAR INTRPT ENABLE
870                                     ;*****
871                                     ;*TEST 17      TEST INTERRUPT ENABLE CLEARED BY CLR PBSR
872                                     ;*****
873 003152 000004                       TST17: SCOPE
874 003154 005077 176404                CLR      2PBSR                      ;CLEAR STATUS REGISTER
875 003160 005037 001124                CLR      SGDDAT              ;EXPECTED DATA
876 003164 052777 040000 176372        BIS      2BIT14,2PBSR          ;SET INTERRUPT ENABLE
877 003172 005077 176366                CLR      2PBSR              ;CLEAR STATUS REGISTER
878 003176 017737 176362 001126        MOV      2PBSR,$BDDAT          ;READ ACTUAL DATA
879 003204 023737 001124 001126        CMP      SGDDAT,$BDDAT        ;COMPARE RESULTS
880 003212 001401                       BEQ      TST20                  ;;BRANCH IF EQUAL
881 003214 104013                       ERROR 13                         ;FAILED TO CLEAR INTERRUPT ENABLE
882                                     ;*****
883                                     ;*TEST 20      GENERATE INTERRUPT WITH PB FLAG AND INTRPT EN
884                                     ;*****
885 003216 000004                       TST20: SCOPE
886 003220 005077 176340                CLR      2PBSR                      ;CLEAR STATUS REGISTER
887 003224 012777 003274 176340        MOV      225,2VECT          ;INTERRUPT ADDRESS
888 003232 012737 140000 001124        MOV      2BIT15!2BIT14,SGDDAT ;EXPECTED DATA
889 003240 005037 001126                CLR      $BDDAT
890 003244 012777 040000 176312        MOV      2BIT14,2PBSR          ;SET INTERRUPT ENABLE
891 003252 012777 140000 176304        MOV      2BIT15!2BIT14,2PBSR ;SET PB FLAG
892 003260 005037 177776                CLR      PSW                      ;LOWER PSW
893 003264 000240                       15:  NOP
894 003266 000240                       NOP
895 003270 104010                       ERROR 10                         ;PB FLAG FAILED TO GENERATE INTERRUPT
896 003272 000401                       BR      35                      ;AND CLEAR I.E.
897 003274 022626                       25:  CMP      (SP)+,(SP)+      ;POP STACK
898                                     35:  CLR      2PBSR              ;CLEAR I.E.
899 003276 005077 176262

```

MAINDEC-11-DZLKA-A
DZLKA.P11 T20

MACY11 27(732) 25-SEP-76 09:47 PAGE 22
GENERATE INTERRUPT WITH PB FLAG AND INTRPT EN

```

900
901
902
903
904 003302 000004
905 003304 005077 176254
906 003310 113737 001253 001576
907 003316 042737 177437 001576
908 003324 013737 001576 001600
909 003332 162737 000040 001600
910 003340 012777 003404 176224
911 003346 012777 000340 176220
912 003354 013737 001576 177776
913 003362 012777 040000 176174
914 003370 012777 140000 176166
915 003376 000240
916 003400 000240
917 003402 000403
918 003404 022626
919 003406 104010
920 003410 000425
921 003412 012777 003450 176152
922 003420 012777 000340 176146
923 003426 013737 001600 177776
924 003434 000240
925 003436 000240
926 003440 005037 001126
927 003444 104010
928 003446 000406
929 003450 022626
930 003452 012777 001610 176112
931 003460 005077 176100
932
933
934
935
936 003464 000004
937 003466 012737 000340 177776
938 003474 005077 176066
939 003500 005077 176060
940 003504 012737 140000 001124
941 003512 013777 001124 176044
942 003520 017737 176040 001126
943 003526 017737 176032 001126
944 003534 023737 001124 001126
945 003542 001401
946 003544 104007

;*****
;TEST 21 TEST FOR CORRECTNESS OF PRIORITY OF INTERRUPT
;*****
TST21: SCOPE
        CLR      2PBSR      ;CLEAR STATUS REGISTER
        MOVB     $VECT1+1,TEMP ;GET PRIORITY LEVEL
        BIC      #177437,TEMP ;MASK TO VECTOR BITS
        MOV      TEMP,TEMP1  ;SAVE IT
        SUB      #40,TEMP1   ;LOWER PRIORITY
        MOV      #15,2VECT   ;INTERRUPT ADDRESS
        MOV      #PR7,2VECTR1 ;LOAD VECTOR ADDRESS PSW
        MOV      TEMP,PSW    ;LOWER PSW
        MOV      #BIT14,2PBSR ;SET INTERRUPT ENABLE
        MOV      #BIT15:BIT14,2PBSR ;SET PB FLAG
        NOP
        NOP
        BR       2S          ;INTERRUPT NOT SERVICED
1S:      CMP      (SP)+,(SP)+ ;POP STACK
        ERROR    10          ;INTERRUPT LEVEL NOT CORRECT
        BR       TST22       ;;BRANCH TO NEXT TEST
2S:      MOV      #35,2VECT   ;INTERRUPT ADDRESS
        MOV      #PR7,2VECTR1 ;SET PRIORITY
        MOV      TEMP1,PSW    ;LOAD PSW
        NOP
        NOP
        CLR      $BDDAT
        ERROR    10          ;PB FLAG FAILED TO GENERATE INTERRUPT
        BR       TST22       ;;BRANCH TO NEXT TEST
3S:      CMP      (SP)+,(SP)+ ;POP STACK
        MOV      #UNEXP,2VECT ;UNEXPECTED INTERRUPT ADDRESS
        CLR      2PBSR      ;CLEAR OUT PBSR
;*****
;TEST 22 TEST PB FLAG IS NOT CLEARED BY READ STATUS WHEN INTR ENABLE IS SET.
;*****
TST22: SCOPE
        MOV      #340,PSW
        CLR      2PBR      ;CLEAR BUTTON REGISTER
        CLR      2PBSR     ;CLEAR STATUS REGISTER
        MOV      #BIT15:BIT14,$GDDAT ;LOAD EXPECTED DATA
        MOV      $GDDAT,2PBSR ;LOAD STATUS
        MOV      2PBSR,$BDDAT ;READ STATUS REG
        MOV      2PBSR,$BDDAT ;READ STATUS REG AGAIN
        CMP      $GDDAT,$BDDAT ;COMPARE REGISTERS
        BEQ      TST23       ;;BR IF PB FLAG CLEARED IN ERROR
        ERROR    7           ;ERRONEOUSLY CLEARED PB FLAG

```



```

947
948
949
950 003546 000004
951 003550 005077 176014 001606
952 003554 012737 000001 001606
953 003562 012777 177777 176000
954 003570 012737 177777 001124
955 003576 043737 001606 001124
956 003604 043777 001606 175756
957 003612 017737 175752 001126
958 003620 023737 001124 001126
959 003626 001401
960 003630 104015
961 003632 006137 001606
962 003636 103351
963
964
965
966 003640 000004
967 003642 012737 000001 001606
968 003650 005077 175714
969 003654 013737 001606 001124
970 003662 053777 001606 175700
971 003670 017737 175674 001126
972 003676 023737 001124 001126
973 003704 001401
974 003706 104016
975 003710 006137 001606
976 003714 103355
977
978
979
980
981 003716 000004
982 003720 005077 175644
983 003724 012777 052525 175636
984 003732 012737 052777 001124
985 003740 152777 177777 175622
986 003746 017737 175616 001126
987 003754 023737 001124 001126
988 003762 001401
989 003764 104017
990 003766 012777 052525 175574
991 003774 013737 001570 001604
992 004002 005237 001604
993 004006 012737 177525 001124
994 004014 152777 177777 175562
995 004022 017737 175542 001126
996 004030 023737 001124 001126
997 004036 001401
998 004040 104020
999

*****
;TEST 23 TEST BIC INSTRUCTION CLEARS LDR
*****
TST23: SCOPE
CLR JLDL ;CLEAR LAMP DATA REGISTER
MOV #BIT0,BITSAV ;INITIALIZE BITSAV
2S: MOV #-1,JLDL ;SET REGISTER TO ALL 1'S
MOV #-1,$GDDAT ;EXPECTED
BIC BITSAV,$GDDAT ;DATA
BIC BITSAV,JLDL ;CLEAR PBSR BIT
MOV JLDL,$BDDAT ;READ ACTAL DATA
CMP $GDDAT,$BDDAT ;COMPARE RESULTS
BEQ 15 ;BRANCH IF EQUAL
ERROR 15 ;FAILED TO CLEAR LDR BIT
1S: ROL BITSAV ;SHIFT TO TEST NEXT BIT
BCC 2S ;BRANCH IF NOT DONE ALL BITS
*****
;TEST 24 TEST BIS INTRUCTION SETS LDR
*****
TST24: SCOPE
MOV #BIT0,BITSAV ;INITIAIALIZE BITSAV
2S: CLR JLDL ;CLEAR LAMP DATA REGISTER
MOV BITSAV,$GDDAT ;DATA
BIS BITSAV,JLDL ;SET LDR BIT
MOV JLDL,$BDDAT ;READ ACTUAL DATA
CMP $GDDAT,$BDDAT ;COMPARE RESULTS
BEQ 15 ;BRANCH IF EQUAL
ERROR 16 ;FAILED TO SET LDR BIT
1S: ROL BITSAV ;SHIFT TO TEST NEXT BIT
BCC 2S ;BRANCH BACK IF NOT DONE ALL BITS
*****
;TEST 25 TEST BISB INSTRUCTION SETS LDR
*****
TST25: SCOPE
CLR JLDL ;CLEAR LAMP DATA REGISTER
MOV #52525,JLDL ;LOAD ALTERNATING 1'S IN LDR
MOV #52777,$GDDAT ;EXPECTED DATA
BISB #-1,JLDL ;SET LOWER BYTE TO ALL 1'S
MOV JLDL,$BDDAT ;READ ACTUAL DATA
CMP $GDDAT,$BDDAT ;COMPARE RESULTS
BEQ 15 ;BRANCH IF EQUAL
ERROR 17 ;FAILED TO SET LOWER BYTE & LEAVE UPPER BYTE THE
1S: MOV #52525,JLDL ;LOAD ALTERNATING 1'S IN LDR
MOV LDR,TEMPL ;GET ADDRESS OF LDR
INC TEMPL ;GET ADDRESS OF UPPER BYTE OF LDR
MOV #177525,$GDDAT ;EXPECTED DATA
BISB #-1,JTEMPL ;SET UPPER BYTE TO ALL 1'S
MOV JLDL,$BDDAT ;READ ACTUAL DATA
CMP $GDDAT,$BDDAT ;COMPARE RESULTS
BEQ TST26 ;BRANCH IF EQUAL
ERROR 20 ;FAILED TO SET UPPER BYTE &
; LEAVE LOWER BYTE THE SAME

```

```

1000
1001
1002
1003 004042 000004
1004 004044 005077 175520
1005 004050 012777 052525 175512
1006 004056 012737 052400 001124
1007 004064 142777 177777 175476
1008 004072 017737 175472 001126
1009 004100 023737 001124 001126
1010 004106 001401
1011 004110 104021
1012 004112 012777 052525 175450 1S:
1013 004120 013737 001570 001604
1014 004126 005237 001604
1015 004132 012737 000125 001124
1016 004140 142777 177777 175436
1017 004146 017737 175416 001126
1018 004154 023737 001124 001126
1019 004162 001401
1020 004164 104022
1021
1022
1023
1024
1025
1026 004166 000004
1027 004170 005077 175370
1028 004174 005037 001124
1029 004200 012777 052525 175362
1030 004206 017737 175352 001126
1031 004214 023737 001124 001126
1032 004222 001401
1033 004224 104023
1034 004226 005077 175336 1S:
1035 004232 012777 100000 175324
1036 004240 017737 175324 001126
1037 004246 023737 001124 001126
1038 004254 001401
1039 004256 104024
1040

*****
*TEST 26 TEST BICB INSTRUCTION CLEARS LDR
*****
TST26: SCOPE
CLR 2LDR ;CLEAR LAMP DATA REGISTER
MOV #52525, 2LDR ;LOAD ALTERNATING 1'S IN LSR
MOV #52400, $GDDAT ;EXPECTED DATA
BICB #-1, 2LDR ;CLEAR LOWER BYTE OF LDR
MOV 2LDR, $BDDAT ;READ ACTUAL DTA
CMP $GDDAT, $BDDAT ;COMPARE RESULTS
BEQ 1S ;BRANCH IF EQUAL
ERROR 21 ;FAILED TO CLEAR LOWER BYTE & LEAVE UPPER BYTE T
MOV #52525, 2LDR ;LOAD ALTERNATING 1'S IN LDR
MOV LDR, TEMPL ;GET ADDRESS OF LDR
INC TEMPL ;GET ADDRESS OF UPPER BYTE OF LDR
MOV #125, $GDDAT ;EXPECTED DATA
BICB #-1, 2TEMPL ;CLEAR UPPER BYTE OF LDR
MOV 2LDR, $BDDAT ;READ ACTUAL DATA
CMP $GDDAT, $BDDAT ;COMPARE RESULTS
BEQ TST27 ;;BRANCH IF EQUAL
ERROR 22 ;FAILED TO CLEAR UPPER BYTE &
; LEAVE LOWER BYTE THE SAME

*****
*TEST 27 TEST THAT PBSR AND LDR DON'T CHANGE
*****
TST27: SCOPE
CLR 2PBSR ;CLEAR STATUS REGISTER
CLR $GDDAT ;LOAD EXPECTED DATA
MOV #52525, 2LDR ;LOAD PATTERN INTO LDR
MOV 2PBSR, $BDDAT ;READ STATUS REGISTER
CMP $GDDAT, $BDDAT ;COMPARE RESULTS
BEQ 1S ;;GO TO NEXT SECTION
ERROR 23
CLR 2LDR ;CLEAR LDR
MOV #BIT15, 2PBSR ;SET BIT 15 IN PBSR
MOV 2LDR, $BDDAT ;READ LDR DATA
CMP $GDDAT, $BDDAT ;COMPARE RESULTS
BEQ TST30 ;;GO TO NEXT TEST
ERROR 24

```



```

1041                                     ;*****
1042                                     ;*TEST 30      TEST THAT PBR AND LDR DON'T CHANGE
1043                                     ;*****
1044 TST30:  SCOPE
1045          CLR      @PBR                      ;CLEAR PBR
1046          MOV      #BIT0,@PBR                ;SET ALL 1'S IN PBR
1047          MOV      #177777,$GDDAT            ;LOAD EXPECTED DATA
1048          MOV      #52525,@LDR               ;LOAD PATTERN INTO LDR
1049          MOV      @PBR,$BDDAT              ;READ PBR
1050          CMP      $GDDAT,$BDDAT             ;COMPARE RESULTS
1051          BEQ      1$                          ;;GO TO NEXT SECTION
1052          ERROR    25
1053          CLR      @LDR                      ;CLEAR LDR
1054          CLR      @PBR                      ;CLEAR PBR
1055          CLR      $GDDAT                    ;LOAD EXPECTED DATA
1056          MOV      #BIT0,@PBR                ;SET PBR TO ALL ONES
1057          MOV      @LDR,$BDDAT              ;READ LDR
1058          CMP      $GDDAT,$BDDAT             ;READ LDR
1059          CMP      $GDDAT,$BDDAT             ;COMPARE RESULTS
1060          BEQ      TST31                      ;;GO TO NEXT TEST
1061          ERROR    26
1062
1063                                     ;*****
1064                                     ;*TEST 31      TEST THAT PBR AND PBR DON'T CHANGE
1065                                     ;*****
1066 TST31:  SCOPE
1067          CLR      @PBR                      ;CLEAR PBR
1068          CLR      $GDDAT                    ;LOAD EXPECTED DATA
1069          MOV      #52525,@PBR               ;LOAD PATTERN INTO PBR
1070          MOV      @PBR,$BDDAT              ;READ PBR
1071          CMP      $GDDAT,$BDDAT             ;COMPARE RESULTS
1072          BEQ      1$                          ;;GO TO NEXT SECTION
1073          ERROR    27
1074          CLR      @PBR                      ;CLEAR PBR
1075          MOV      #BIT0,@PBR                ;SET PBR TO ALL ONES
1076          MOV      @PBR,$BDDAT              ;READ PBR
1077          BIC      #36,$BDDAT               ;MASK TO "ENCODER" BITS
1078          CMP      $GDDAT,$BDDAT             ;COMPARE RESULTS
1079          BEQ      2$                          ;;BR IF CORRECT
1080          ERROR    30
1081          CLR      @PBR                      ;PBR CHANGED IN ERROR
1082          CLR      @LDR                      ;CLEAR BP REGISTER
1083          CLR      @PBR                      ;CLEAR LAMPS
1084                                     ;CLEAR STATUS

```

MAINDEC-11-DZLKA-A
DZLKA.P11 T31

MACY11 27(732) 25-SEP-76 09:47 PAGE 26
TEST THAT PBSR AND PBR DON'T CHANGE

```

1085
1086
1087
1088
1089 004514 000004
1090 004516 032777 010000 174414
1091 004524 001401
1092 004526 000000
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102 004530
1103 004530 000004
1104 004532 005037 001102
1105 004536 005037 001166
1106 004542 005237 001210
1107 004546 042737 100000 001210
1108 004554 005327
1109 004556 000001
1110 004560 003022
1111 004562 012737
1112 004564 000001
1113 004566 004556
1114 004570 104400 004635
1115 004574 013746 001210
1116 004600 104404
1117 004602 104400 004632
1118 004606 013700 000042
1119 004612 001405
1120 004614 000005
1121 004616 004710
1122 004620 000240
1123 004622 000240
1124 004624 000240
1125 004626
1126 004626 000137
1127 004630 002104
1128 004632 377 377 000
1129 004635 015 042412 042116
1130 004642 050040 051501 020123
1131 004650 000043
1132

;*****
;TEST 32 HALT AT END OF DIAGNOSTIC
;*****
TST32: SCOPE
        BIT #BIT12,DSWR ;IS BIT 12 SET?
        BEQ SEOP ;;NO-PRINT END-OF-PASS MESSAGE
        HALT ;HALT OT END OF LOGIC TESTS

.SBTTL END OF PASS ROUTINE

;*****
;INCREMENT THE PASS NUMBER ($PASS)
;TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;IF THERES A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO INIT
SEOP:
        SCOPE
        CLR $STNM ;;ZERO THE TEST NUMBER
        CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
        INC $PASS ;;INCREMENT THE PASS NUMBER
        BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
        DEC (PC)+ ;;LOOP?
SEOPCT: .WORD 1
        BGT $DOAGN ;;YES
        MOV (PC)+,2(PC)+ ;;RESTORE COUNTER
SENDCT: .WORD 1
        SEOPCT
        TYPE $SENDMG ;;TYPE "END PASS #"
        MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
        TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
        TYPE $ENULL ;;TYPE A NULL CHARACTER
SGET42: MOV 2#42,R0 ;;GET MONITOR ADDRESS
        BEQ $DOAGN ;;BRANCH IF NO MONITOR
        RESET ;;CLEAR THE WORLD
SENDAD: JSR PC,(R0) ;;GO TO MONITOR
        NOP ;;SAVE ROOM
        NOP ;;FOR
        NOP ;;ACT11
$DOAGN: JMP 2(PC)+ ;;RETURN
SRTNAD: .WORD INIT
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
$SENDMG: .ASCIIZ <15><12>/END PASS #/

```


MAINDEC-11-DZLKA-A
DZLKAA.P11 T33

MACY11 27(732) 25-SEP-76 09:47 PAGE 27
PUSH BUTTON TURNS LDR LIGHT ON AND OFF LOOP

```

1133      ;*****
1134      ;*TEST 33      PUSH BUTTON TURNS LDR LIGHT ON AND OFF LOOP
1135      ;*****
1136      004652 000004
1137      004654 012706 001100
1138      004660 005077 174700
1139      004664 104400
1140      004666 007042
1141      004670 005777 174670
1142      004674 100375
1143      004676 017777 174664 174664
1144      004704 000771
1145
1146      ;*****
1147      ;*TEST 34      TEST ENCODER WITH PUSH BUTTON
1148      ;*****
1149      004706 000004
1150      004710 012706 001100
1151      004714 005077 174644
1152      004720 104400
1153      004722 007140
1154      004724 005077 174636
1155      004730 005777 174630
1156      004734 100375
1157      004736 017737 174622 001600
1158      004744 042737 177741 001600
1159      004752 012737 004776 001602
1160      004760 063737 001600 001602
1161      004766 017777 174610 174574
1162      004774 000753
1163
1164      004776 000001
1165      005000 000002
1166      005002 000004
1167      005004 000010
1168      005006 000020
1169      005010 000040
1170      005012 000100
1171      005014 000200
1172      005016 000400
1173      005020 001000
1174      005022 002000
1175      005024 004000
1176      005026 010000
1177      005030 020000
1178      005032 040000
1179      005034 100000

      TST33: SCOPE
      BEGIN1: MOV      #STACK,SP      ;LOAD STACK POINTER
              CLR      @PBSR          ;CLEAR STATUS REGISTER
              TYPE     ;TYPE OUT EXPLANATION
              MESS1    ;OF TEST
      1S:      TST      @PBSR          ;STALL
              BPL      1S             ;TIME
              MOV      @PBR,@LDR      ;SET OR TURN OFF LIGHT
              BR       1S             ;BRANCH BACK

      ;*****
      ;*TEST 34      TEST ENCODER WITH PUSH BUTTON
      ;*****
      TST34: SCOPE
      BEGIN2: MOV      #STACK,SP      ;LOAD STACK POINTER
              CLR      @PBSR          ;CLEAR STATUS REGISTER
              TYPE     ;TYPE OUT EXPLANATION
              MESS3    ;OF TEST
      1S:      CLR      @PBR          ;CLEAR PUSH BUTTON REGISTER
      2S:      TST      @PBSR          ;STALL
              BPL      2S             ;TIME
              MOV      @PBSR,TEMP1    ;READ STATUS REGISTER
              BIC      #177741,TEMP1  ;GET ENCODER BITS
              MOV      #TABLE,TEMP2   ;GET STARTING ADDRESS
              ADD      TEMP1,TEMP2     ;ADD OFFSET TO ADDRESS
              MOV      @TEMP2,@LDR     ;SET OR TURN OFF LAMP
              BR       1S             ;BRANCH BACK

      TABLE: BIT0
              BIT1
              BIT2
              BIT3
              BIT4
              BIT5
              BIT6
              BIT7
              BIT8
              BIT9
              BIT10
              BIT11
              BIT12
              BIT13
              BIT14
              BIT15

```

1180						
1181						
1182	005036	040506	046111	042105	EM1:	.SBTTL ASCII MESSAGES
1183	005044	052040	020117	047514		.ASCIZ /FAILED TO LOAD AND READ LDR CORRECTLY/
1184	005052	042101	040440	042116		
1185	005060	051040	040505	020104		
1186	005066	042114	020122	047503		
1187	005074	051122	041505	046124		
1188	005102	000131				
1189	005104	040506	046111	042105	EM2:	.ASCIZ /FAILED TO SET OR CLEAR PBR/
1190	005112	052040	020117	042523		
1191	005120	020124	051117	041440		
1192	005126	042514	051101	050040		
1193	005134	051102	000			
1194	005137	106	044501	042514	EM3:	.ASCIZ /FAILED TO CLEAR PB FLAG/
1195	005144	020104	047524	041440		
1196	005152	042514	051101	050040		
1197	005160	020102	046106	043501		
1198	005166	000				
1199	005167	106	044501	042514	EM4:	.ASCIZ /FAILED TO LOAD AND READ PB FLAG/
1200	005174	020104	047524	046040		
1201	005202	040517	020104	047101		
1202	005210	020104	042522	042101		
1203	005216	050040	020102	046106		
1204	005224	043501	000			
1205	005227	106	044501	042514	EM5:	.ASCIZ /FAILED TO CLEAR INTERRUPT ENABLE/
1206	005234	020104	047524	041440		
1207	005242	042514	051101	044440		
1208	005250	052116	051105	052522		
1209	005256	052120	042440	040516		
1210	005264	046102	000105			
1211	005270	040506	046111	042105	EM6:	.ASCIZ /FAILED TO SET INTERRUPT ENABLE/
1212	005276	052040	020117	042523		
1213	005304	020124	047111	042524		
1214	005312	051122	050125	020124		
1215	005320	047105	041101	042514		
1216	005326	000				
1217	005327	105	051122	047117	EM7:	.ASCIZ /ERRONEOUSLY CLEARED PB FLAG/
1218	005334	047505	051525	054514		
1219	005342	041440	042514	051101		
1220	005350	042105	050040	020102		
1221	005356	046106	043501	000		
1222	005363	106	044501	042514	EM10:	.ASCIZ /FAILED TO GENERATE INTERRUPT WITH PB FLAG/
1223	005370	020104	047524	043440		
1224	005376	047105	051105	052101		
1225	005404	020105	047111	042524		
1226	005412	051122	050125	020124		
1227	005420	044527	044124	050040		
1228	005426	020102	046106	043501		
1229	005434	000				
1230	005435	103	051114	050040	EM11:	.ASCIZ /CLR PBSR FAILED TO CLEAR PB FLAG/
1231	005442	051502	020122	040506		
1232	005450	046111	042105	052040		
1233	005456	020117	046103	040505		
1234	005464	020122	041120	043040		
1235	005472	040514	000107			

C03

 MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 29
 DZLKA.P11 ASCII MESSAGES

1236	005476	047111	042524	051122	EM12: .ASCIZ /INTERRUPT OK FAILED TO CLEAR INTR FLOP/
1237	005504	050125	020124	045517	
1238	005512	043040	044501	042514	
1239	005520	020104	047524	041440	
1240	005526	042514	051101	044440	
1241	005534	052116	020122	046106	
1242	005542	050117	000		
1243	005545	103	051114	050040	EM13: .ASCIZ /CLR PBSR FAILED TO CLEAR INTERRUPT ENABLE/
1244	005552	051502	020122	040506	
1245	005560	046111	042105	052040	
1246	005566	020117	046103	040505	
1247	005574	020122	047111	042524	
1248	005602	051122	050125	020124	
1249	005610	047105	041101	042514	
1250	005616	000			
1251	005617	125	042516	050130	EM14: .ASCIZ /UNEXPECTED INTERRUPT/
1252	005624	041505	042524	020104	
1253	005632	047111	042524	051122	
1254	005640	050125	000124		
1255	005644	044502	020103	040506	EM15: .ASCIZ /BIC FAILED TO CLEAR LDR /
1256	005652	046111	042105	052040	
1257	005660	020117	046103	040505	
1258	005666	020122	042114	020122	
1259	005674	000			
1260	005675	102	051511	043040	EM16: .ASCIZ /BIS FAILED TO SET LDR/
1261	005702	044501	042514	020104	
1262	005710	047524	051440	052105	
1263	005716	046040	051104	000	
1264	005723	102	051511	020102	EM17: .ASCIZ /BISB FAILED TO SET LOWER BYTE OR LEAVE UPPER BYTE THE SAME/
1265	005730	040506	046111	042105	
1266	005736	052040	020117	042523	
1267	005744	020124	047514	042527	
1268	005752	020122	054502	042524	
1269	005760	047440	020122	042514	
1270	005766	053101	020105	050125	
1271	005774	042520	020122	054502	
1272	006002	042524	052040	042510	
1273	006010	051440	046501	000105	
1274	006016	044502	041123	043040	EM20: .ASCIZ /BISB FAILED TO SET UPPER BYTE OR LEAVE LOWER BYTE THE SAME/
1275	006024	044501	042514	020104	
1276	006032	047524	051440	052105	
1277	006040	052440	050120	051105	
1278	006046	041040	052131	020105	
1279	006054	051117	046040	040505	
1280	006062	042526	046040	053517	
1281	006070	051105	041040	052131	
1282	006076	020105	044124	020105	
1283	006104	040523	042515	000	
1284	006111	102	041511	020102	EM21: .ASCIZ /BICB FAILED TO CLEAR LOWER BYTE OR LEAVE UPPER BYTE THE SAME/
1285	006116	040506	046111	042105	
1286	006124	052040	020117	046103	
1287	006132	040505	020122	047514	
1288	006140	042527	020122	054502	
1289	006146	042524	047440	020122	
1290	006154	042514	053101	020105	
1291	006162	050125	042520	020122	

D03

 MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 30
 DZLKA.P11 ASCII MESSAGES

1292	006170	054502	042524	052040	
1293	006176	042510	051440	046501	
1294	006204	000105			
1295	006206	044502	041103	043040	EM22: .ASCIZ /BICB FAILED TO CLEAR UPPER BYTE OR LEAVE LOWER BYTE THE SAME/
1296	006214	044501	042514	020104	
1297	006222	047524	041440	042514	
1298	006230	051101	052440	050120	
1299	006236	051105	041040	052131	
1300	006244	020105	051117	046040	
1301	006252	040505	042526	046040	
1302	006260	053517	051105	041040	
1303	006266	052131	020105	044124	
1304	006274	020105	040523	042515	
1305	006302	000			
1306	006303	114	040517	044504	EM23: .ASCIZ /LOADING LDR CHANGED PBSR DATA/
1307	006310	043516	046040	051104	
1308	006316	041440	040510	043516	
1309	006324	042105	050040	051502	
1310	006332	020122	040504	040524	
1311	006340	000			
1312	006341	114	040517	044504	EM24: .ASCIZ /LOADING PBSR CHANGED LDR DATA/
1313	006346	043516	050040	051502	
1314	006354	020122	044103	047101	
1315	006362	042507	020104	042114	
1316	006370	020122	040504	040524	
1317	006376	000			
1318	006377	114	040517	044504	EM25: .ASCIZ /LOADING LDR CHANGED PBR DATA/
1319	006404	043516	046040	051104	
1320	006412	041440	040510	043516	
1321	006420	042105	050040	051102	
1322	006426	042040	052101	000101	
1323	006434	047514	042101	047111	EM26: .ASCIZ /LOADING PBR CHANGED LDR DATA/
1324	006442	020107	041120	020122	
1325	006450	044103	047101	042507	
1326	006456	020104	042114	020122	
1327	006464	040504	040524	000	
1328	006471	114	040517	044504	EM27: .ASCIZ /LOADING PBSR CHANGED PBR DATA/
1329	006476	043516	050040	051502	
1330	006504	020122	044103	047101	
1331	006512	042507	020104	041120	
1332	006520	020122	040504	040524	
1333	006526	000			
1334	006527	114	040517	044504	EM30: .ASCIZ /LOADING PBR CHANGED PBSR DATA/
1335	006534	043516	050040	051102	
1336	006542	041440	040510	043516	
1337	006550	042105	050040	051502	
1338	006556	020122	040504	040524	
1339	006564	000			
1340	006565	105	051122	041520	DH1: .ASCIZ /ERRPC LDR EXPECTED ACTUAL/
1341	006572	020040	046040	051104	
1342	006600	020040	042440	050130	
1343	006606	041505	042524	020104	
1344	006614	020040	041501	052524	
1345	006622	046101	000		
1346	006625	105	051122	041520	DH2: .ASCIZ /ERRPC PBR EXPECTED ACTUAL/
1347	006632	020040	050040	051102	

E03

 MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 31
 DZLKA.P11 ASCII MESSAGES

1348	006640	020040	042440	050130	
1349	006646	041505	042524	020104	
1350	006654	020040	041501	052524	
1351	006662	046101	000		
1352	006665	105	051122	041520	DH3: .ASCIZ /ERRPC PBSR EXPECTED ACTUAL/
1353	006672	020040	050040	051502	
1354	006700	020122	020040	054105	
1355	006706	042520	052103	042105	
1356	006714	020040	040440	052103	
1357	006722	040525	000114		
1358	006726	051105	050122	020103	DH4: .ASCIZ /ERRPC PBSR LOCATION OF FAILURE/
1359	006734	050040	051502	020122	
1360	006742	020040	046040	041517	
1361	006750	052101	047511	020116	
1362	006756	043117	043040	044501	
1363	006764	052514	042522	000	
1364		006772			.EVEN
1365					
1366	006772	001116	001570	001124	DT1: SERRPC, LDR, SGDDAT, SBDDAT, 0
1367	007000	001126	000000		
1368	007004	001116	001566	001124	DT2: SERRPC, PBR, SGDDAT, SBDDAT, 0
1369	007012	001126	000000		
1370	007016	001116	001564	001124	DT3: SERRPC, PBSR, SGDDAT, SBDDAT, 0
1371	007024	001126	000000		
1372	007030	001116	001564	001124	DT4: SERRPC, PBSR, SGDDAT, 0
1373	007036	000000			
1374	007040	000000			DF1: 0
1375					
1376	007042	005015	051120	051505	MESS1: .ASCIZ <15><12>/PRESS PUSH BUTTON. CORRESPONDING LIGHT SHOULD COMPLEMENT./<15><
1377	007050	020123	052520	044123	
1378	007056	041040	052125	047524	
1379	007064	027116	041440	051117	
1380	007072	042522	050123	047117	
1381	007100	044504	043516	046040	
1382	007106	043511	052110	051440	
1383	007114	047510	046125	020104	
1384	007122	047503	050115	042514	
1385	007130	042515	052116	006456	
1386	007136	000012			
1387	007140	005015	051120	051505	MESS3: .ASCIZ <15><12>/PRESS PUSH BUTTON. ENCODER SHOULD SET CORRESPONDING LIGHT./<15>
1388	007146	020123	052520	044123	
1389	007154	041040	052125	047524	
1390	007162	027116	042440	041516	
1391	007170	042117	051105	051440	
1392	007176	047510	046125	020104	
1393	007204	042523	020124	047503	
1394	007212	051122	051505	047520	
1395	007220	042116	047111	020117	
1396	007226	044514	044107	027124	
1397	007234	005015	000		
1398	007237	015	046412	044501	HEAD1: .ASCIZ <15><12>/MAINDEC-11-DZLKA-A: PUSH BUTTON OPTION DIAGNOSTIC/<15><12>
1399	007244	042116	041505	030455	
1400	007252	026461	055104	045514	
1401	007260	026501	035101	050040	
1402	007266	051525	020110	052502	
1403	007274	052124	047117	047440	

1404 007302 052120 047511 020116
 1405 007310 044504 043501 047516
 1406 007316 052123 041511 005015
 1407 007324 000
 1408 007326

.EVEN

.SBTTL TTY INPUT ROUTINE

::*****
 .ENABL LSB

::*****
 ::SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 ::ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 ::SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 ::WHEN OPERATING IN TTY FLAG MODE.

1420 007326 022737 000176 001140
 1421 007334 001074
 1422 007336 105777 171602
 1423 007342 100071
 1424 007344 117746 171576
 1425 007350 042716 177600
 1426 007354 022726 000007
 1427 007360 001062
 1428 007362 123727 001134 000001
 1429 007370 001456

\$CKSWR: CMP #SWREG,SWR ::IS THE SOFT-SWR SELECTED?
 BNE 15\$::BRANCH IF NO
 TSTB \$STKS ::CHAR THERE?
 BPL 15\$::IF NO, DON'T WAIT AROUND
 MOVB \$STKB,-(SP) ::SAVE THE CHAR
 BIC #1C177,(SP) ::STRIP-OFF THE ASCII
 CMP #7,(SP)+ ::IS IT A CONTROL G?
 BNE 15\$::NO, RETURN TO USER
 CMPB \$AUTOB,#1 ::ARE WE RUNNING IN AUTO-MODE?
 BEQ 15\$::BRANCH IF YES

1431 007372 104400 010053
 1432 007376 104400 010060
 1433 007402 013746 000176
 1434 007406 104401
 1435 007410 104400 010071
 1436 007414 005046
 1437 007416 005046
 1438 007420 105777 171520
 1439 007424 100375

SGTSWR: TYPE ,SCNTLG ::ECHO THE CONTROL-G (1G)
 TYPE ,SMSWR ::TYPE CURRENT CONTENTS
 MOV SWREG,-(SP) ::SAVE SWREG FOR TYPEOUT
 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
 TYPE ,SMNEW ::PROMPT FOR NEW SWR
 19\$: CLR -(SP) ::CLEAR COUNTER
 CLR -(SP) ::THE NEW SWR
 7\$: TSTB \$STKS ::CHAR THERE?
 BPL 7\$::IF NOT TRY AGAIN

1441 007426 117746 171514
 1442 007432 042716 177600

MOVB \$STKB,-(SP) ::PICK UP CHAR
 BIC #1C177,(SP) ::MAKE IT 7-BIT ASCII

1443
 1444
 1445
 1446 007436 021627 000025
 1447 007442 001005
 1448 007444 104400 010046
 1449 007450 062706 000006
 1450 007454 000757

9\$: CMP (SP),#25 ::IS IT A CONTROL-U?
 BNE 10\$::BRANCH IF NOT
 TYPE ,SCNTLU ::YES, ECHO CONTROL-U (1U)
 20\$: ADD #6,SP ::IGNORE PREVIOUS INPUT
 BR 19\$::LET'S TRY IT AGAIN

1451
 1452
 1453 007456 021627 000015
 1454 007462 001022
 1455 007464 005766 000004
 1456 007470 001403
 1457 007472 016677 000002 171440
 1458 007500 062706 000006
 1459 007504 104400 001177

10\$: CMP (SP),#15 ::IS IT A <CR>?
 BNE 16\$::BRANCH IF NO
 TST 4(SP) ::YES, IS IT THE FIRST CHAR?
 BEQ 11\$::BRANCH IF YES
 MOV 2(SP), \$SWR ::SAVE NEW SWR
 11\$: ADD #6,SP ::CLEAR UP STACK
 14\$: TYPE ,SCRLF ::ECHO <CR> AND <LF>


```

1460 007510 123727 001135 000001      CMPB    $INTAG,#1      ;;RE-ENABLE TTY KBD INTERRUPTS?
1461 007516 001003                      BNE     15$           ;;BRANCH IF NOT
1462 007520 012777 000100 171416      MOV     #100,$STKS    ;;RE-ENABLE TTY KBD INTERRUPTS
1463 007526 000002                      RTI                     RETURN
1464 007530 004737 011516      15$:    JSR     PC,$TYPEC    ECHO CHAR
1465 007534 021627 000060      16$:    CMP     (SP),#60    CHAR < 0?
1466 007540 002420                      BLT     18$           BRANCH IF YES
1467 007542 021627 000067      CMP     (SP),#67    CHAR > 7?
1468 007546 003015                      BGT     18$           BRANCH IF YES
1469 007550 042726 000060      BIC     #60,(SP)+    STRIP-OFF ASCII
1470 007554 005766 000002      TST     2(SP)        IS THIS THE FIRST CHAR
1471 007560 001403                      BEQ     17$           BRANCH IF YES
1472 007562 006316                      ASL     (SP)        NO, SHIFT PRESENT
1473 007564 006316                      ASL     (SP)        CHAR OVER TO MAKE
1474 007566 006316                      ASL     (SP)        ROOM FOR NEW ONE.
1475 007570 005266 000002      17$:    INC     2(SP)        KEEP COUNT OF CHAR
1476 007574 056616 177776      BIS     -2(SP),(SP)  SET IN NEW CHAR
1477 007600 000707                      BR      7$           GET THE NEXT ONE
1478 007602 104400 001176      18$:    TYPE    $QUES    TYPE ?<CR><LF>
1479 007606 000720                      BR      20$        SIMULATE CONTROL-U
1480                                .DSABL  LSB

```

```

*****

```

```

;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

```

```

;CALL:

```

```

;RDCHR
;RETURN HERE

```

```

;;INPUT A SINGLE CHARACTER FROM THE TTY
;;CHARACTER IS ON THE STACK
;;WITH PARITY BIT STRIPPED OFF

```

```

1491 007610 011646                      SRDCHR: MOV    (SP),-(SP)    ;;PUSH DOWN THE PC
1492 007612 016666 000004 000002      MOV    4(SP),2(SP)    ;;SAVE THE PS
1493 007620 105777 171320      1$:    TSTB   $STKS    ;;WAIT FOR
1494 007624 100375                      BPL     1$           A CHARACTER
1495 007626 117766 171314 000004      MOVB   $STKB,4(SP)    ;;READ THE TTY
1496 007634 042766 177600 000004      BIC     #177,4(SP)    ;;GET RID OF JUNK IF ANY
1497 007642 026627 000004 000023      CMP     4(SP),#23    ;;IS IT A CONTROL-S?
1498 007650 001013                      BNE     3$           BRANCH IF NO
1499 007652 105777 171266      2$:    TSTB   $STKS    ;;WAIT FOR A CHARACTER
1500 007656 100375                      BPL     2$           LOOP UNTIL ITS THERE
1501 007660 117746 171262      MOVB   $STKB,-(SP)    ;;GET CHARACTER
1502 007664 042716 177600      BIC     #177,(SP)    ;;MAKE IT 7-BIT ASCII
1503 007670 022627 000021      CMP     (SP)+,#21    ;;IS IT A CONTROL-Q?
1504 007674 001366                      BNE     2$           IF NOT DISCARD IT
1505 007676 000750                      BR      1$           YES, RESUME
1506 007700 026627 000004 000140      3$:    CMP     4(SP),#140  ;;IS IT UPPER CASE?
1507 007706 002407                      BLT     4$           BRANCH IF YES
1508 007710 026627 000004 000175      CMP     4(SP),#175    ;;IS IT A SPECIAL CHAR?
1509 007716 003003                      BGT     4$           BRANCH IF YES
1510 007720 042766 000040 000004      BIC     #40,4(SP)    ;;MAKE IT UPPER CASE
1511 007726 000002      4$:    RTI                     GO BACK TO USER

```

```

*****

```

```

;THIS ROUTINE WILL INPUT A STRING FROM THE TTY

```

```

;CALL:

```

```

;RDLIN

```

```

;;INPUT A STRING FROM THE TTY

```

```

1516                                     ;*      RETURN HERE
1517                                     ;*
1518                                     ;: ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1519                                     ;: TERMINATOR WILL BE A BYTE OF ALL 0'S
1519 007730 010346 SRDLIN: MOV      R3, -(SP)      ;: SAVE R3
1520 007732 012703 010036 15:      MOV      #STTYIN, R3 ;: GET ADDRESS
1521 007736 022703 010046 25:      CMP      #STTYIN+8., R3 ;: BUFFER FULL?
1522 007742 101405      BLOS      45 ;: BR IF YES
1523 007744 104407      ROCHR      ;: GO READ ONE CHARACTER FROM THE TTY
1524 007746 112613      MOV      (SP)+, (R3) ;: GET CHARACTER
1525 007750 122713 000177 105:     CMP      #177, (R3) ;: IS IT A RUBOUT
1526 007754 001003      BNE      35 ;: SKIP IF NOT
1527 007756 104400 001176 45:      TYPE      $QUES ;: TYPE A '?'
1528 007762 000763      BR      15 ;: CLEAR THE BUFFER AND LOOP
1529 007764 111337 010034 35:      MOV      (R3), 95 ;: ECHO THE CHARACTER
1530 007770 104400 010034      TYPE      95
1531 007774 122723 000015      CMP      15, (R3)+ ;: CHECK FOR RETURN
1532 010000 001356      BNE      25 ;: LOOP IF NOT RETURN
1533 010002 105063 177777      CLRB      -1(R3) ;: CLEAR RETURN (THE 15)
1534 010006 104400 001200      TYPE      $LF ;: TYPE A LINE FEED
1535 010012 012603      MOV      (SP)+, R3 ;: RESTORE R3
1536 010014 011646      MOV      (SP), -(SP) ;: ADJUST THE STACK AND PUT ADDRESS OF THE
1537 010016 016666 000004 000002 MOV      4(SP), 2(SP) ;: FIRST ASCII CHARACTER ON IT
1538 010024 012766 010036 000004 MOV      #STTYIN, 4(SP)
1539 010032 000002      RTI ;: RETURN
1540 010034      000      95:      .BYTE      0 ;: STORAGE FOR ASCII CHAR. TO TYPE
1541 010035      000      .BYTE      0 ;: TERMINATOR
1542 010036 000010      STTYIN: .BLKB      8. ;: RESERVE 8 BYTES FOR TTY INPUT
1543 010046 052536 005015 000      SCNTLU: .ASCIZ /1U/15>12> ;: CONTROL "U"
1544 010053 136 006507 000012      SCNTLG: .ASCIZ /1G/15>12> ;: CONTROL "G"
1545 010060 005015 053523 020122 SMSWR: .ASCIZ <15><12>/SWR = /
1546 010066 020075 000      SMNEW: .ASCIZ / NEW = /
1547 010071 040 047040 053505
1548 010076 036440 000040

```


MAINDEC-11-DZLKA-A
DZLKA.P11

MACY11 27(732) 25-SEP-76 09:47 PAGE 35
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561 010102
1562 010102 010046
1563 010104 010146
1564 010106 010246
1565 010110 010346
1566 010112 010546
1567 010114 012746 020200
1568 010120 016605 000020
1569 010124 100004
1570 010126 005405
1571 010130 112766 000055 000001
1572 010136 005000
1573 010140 012703 010316
1574 010144 112723 000040
1575 010150 005002
1576 010152 016001 010306
1577 010156 160105
1578 010160 002402
1579 010162 005202
1580 010164 000774
1581 010166 060105
1582 010170 005702
1583 010172 001002
1584 010174 105716
1585 010176 100407
1586 010200 106316
1587 010202 103003
1588 010204 116663 000001 177777
1589 010212 052702 000060
1590 010216 052702 000040
1591 010222 110223
1592 010224 005720
1593 010226 020027 000010
1594 010232 002746
1595 010234 003002
1596 010236 010502
1597 010240 000764
1598 010242 105726
1599 010244 100003
1600 010246 116663 177777 177776
1601 010254 105013
1602 010256 012605
1603 010260 012603
1604 010262 012602

;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;REPLACED WITH SPACES.
;CALL:
;*   MOV   NUM,-(SP)           ;;PUT THE BINARY NUMBER ON THE STACK
;*   TYPDS ;;GO TO THE ROUTINE

STYPDS:
MOV   R0,-(SP)           ;;PUSH R0 ON STACK
MOV   R1,-(SP)           ;;PUSH R1 ON STACK
MOV   R2,-(SP)           ;;PUSH R2 ON STACK
MOV   R3,-(SP)           ;;PUSH R3 ON STACK
MOV   R5,-(SP)           ;;PUSH R5 ON STACK
MOV   #20200,-(SP)       ;;SET BLANK SWITCH AND SIGN
MOV   20(SP),R5          ;;GET THE INPUT NUMBER
BPL   1$                 ;;BR IF INPUT IS POS.
NEG   R5                 ;;MAKE THE BINARY NUMBER POS.
MOVB  #'-,1(SP)          ;;MAKE THE ASCII NUMBER NEG.
1$:   CLR   R0            ;;ZERO THE CONSTANTS INDEX
MOV   #SDBLK,R3          ;;SETUP THE OUTPUT POINTER
MOVB  #' ,(R3)+          ;;SET THE FIRST CHARACTER TO A BLANK
2$:   CLR   R2            ;;CLEAR THE BCD NUMBER
MOV   SDBLK(R0),R1       ;;GET THE CONSTANT
3$:   SUB   R1,R5         ;;FORM THIS BCD DIGIT
BLT   4$                 ;;BR IF DONE
INC   R2                 ;;INCREASE THE BCD DIGIT BY 1
4$:   ADD   R1,R5         ;;ADD BACK THE CONSTANT
TST   R2                 ;;CHECK IF BCD DIGIT=0
BNE   5$                 ;;FALL THROUGH IF 0
TSTB  (SP)               ;;STILL DOING LEADING 0'S?
BMI   7$                 ;;BR IF YES
ASLB  (SP)               ;;MSD?
BCC   6$                 ;;BR IF NO
MOVB  1(SP),-1(R3)       ;;YES--SET THE SIGN
BIS   #'0,R2             ;;MAKE THE BCD DIGIT ASCII
BIS   #' ,R2             ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB  R2,(R3)+          ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST   (R0)+              ;;JUST INCREMENTING
CMP   R0,#10            ;;CHECK THE TABLE INDEX
BLT   2$                 ;;GO DO THE NEXT DIGIT
BGT   8$                 ;;GO TO EXIT
MOV   R5,R2             ;;GET THE LSD
BR    6$                 ;;GO CHANGE TO ASCII
8$:   TSTB  (SP)+         ;;WAS THE LSD THE FIRST NON-ZERO?
BPL   9$                 ;;BR IF NO
MOVB  -1(SP),-2(R3)     ;;YES--SET THE SIGN FOR TYPING
CLRB  (R3)               ;;SET THE TERMINATOR
MOV   (SP)+,R5           ;;POP STACK INTO R5
MOV   (SP)+,R3           ;;POP STACK INTO R3
MOV   (SP)+,R2           ;;POP STACK INTO R2

```

J03

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 36
 DZLKA.P11 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

1605	010264	012601			MOV	(SP)+,R1	::POP STACK INTO R1
1606	010266	012600			MOV	(SP)+,R0	::POP STACK INTO R0
1607	010270	104400	010316		TYPE	\$DBLK	::NOW TYPE THE NUMBER
1608	010274	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
1609	010302	012616			MOV	(SP)+,(SP)	
1610	010304	000002			RTI		::RETURN TO USER
1611	010306	023420			\$DTBL:	10000.	
1612	010310	001750				1000.	
1613	010312	000144				100.	
1614	010314	000012				10.	
1615	010316	000004			\$DBLK:	.BLKW 4	

K03

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 37
 DZLKA.P11 READ AN OCTAL NUMBER FROM THE TTY

```

1616 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
1617
1618 *****
1619 *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
1620 *CHANGE IT TO BINARY.
1621 *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
1622 *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
1623 *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
1624 *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
1625 *CALL:
1626 *      RDOCT
1627 *      RETURN HERE
1628 *
1629
1630 010326 011646 000004 000002 SRDOCT: MOV (SP),-(SP)
1631 010330 016666 000004 000002 MOV 4(SP),2(SP)
1632 010336 010046 MOV RO,-(SP)
1633 010340 010146 MOV R1,-(SP)
1634 010342 010246 MOV R2,-(SP)
1635 010344 104410 1S: RDLIN
1636 010346 012600 MOV (SP)+,RO
1637 010350 010037 010454 MOV RO,5S
1638 010354 005001 CLR R1
1639 010356 005002 CLR R2
1640 010360 112046 2S: MOVB (RO)+,-(SP)
1641 010362 001420 BEQ 3S
1642 010364 122716 000060 CMPB #'0,(SP)
1643 010370 003026 BGT 4S
1644 010372 122716 000067 CMPB #'7,(SP)
1645 010376 002423 BLT 4S
1646 010400 006301 ASL R1
1647 010402 006102 ROL R2
1648 010404 006301 ASL R1
1649 010406 006102 ROL R2
1650 010410 006301 ASL R1
1651 010412 006102 ROL R2
1652 010414 042716 177770 BIC #'C7,(SP)
1653 010420 062601 ADD (SP)+,R1
1654 010422 000756 BR 2S
1655 010424 005726 3S: TST (SP)+
1656 010426 010166 000012 MOV R1,12(SP)
1657 010432 010237 010464 MOV R2,SHIOCT
1658 010436 012602 MOV (SP)+,R2
1659 010440 012601 MOV (SP)+,R1
1660 010442 012600 MOV (SP)+,RO
1661 010444 000002 RTI
1662 010446 005726 4S: TST (SP)+
1663 010450 105010 CLRB (RO)
1664 010452 104400 TYPE
1665 010454 000000 5S: .WORD 0
1666 010456 104400 001176 TYPE $QUES
1667 010462 000730 BR 1S
1668 010464 000000 SHIOCT: .WORD 0

```

;; READ AN OCTAL NUMBER
 ;; LOW ORDER BITS ARE ON TOP OF THE STACK
 ;; HIGH ORDER BITS ARE IN SHIOCT
 ;; PROVIDE SPACE FOR THE
 ;; INPUT NUMBER
 ;; PUSH RO ON STACK
 ;; PUSH R1 ON STACK
 ;; PUSH R2 ON STACK
 ;; READ AN ASCII LINE
 ;; GET ADDRESS OF 1ST CHARACTER
 ;; AND SAVE IT
 ;; CLEAR DATA WORD
 ;; PICKUP THIS CHARACTER
 ;; IF ZERO GET OUT
 ;; MAKE SURE THIS CHARACTER
 ;; IS AN OCTAL DIGIT
 ;; *2
 ;; *4
 ;; *8
 ;; STRIP THE ASCII JUNK
 ;; ADD IN THIS DIGIT
 ;; LOOP
 ;; CLEAN TERMINATOR FROM STACK
 ;; SAVE THE RESULT
 ;; POP STACK INTO R2
 ;; POP STACK INTO R1
 ;; POP STACK INTO RO
 ;; RETURN
 ;; CLEAN PARTIAL FROM STACK
 ;; SET A TERMINATOR
 ;; TYPE UP THRU THE BAD CHAR.
 ;; "?" "CR" & "LF"
 ;; TRY AGAIN
 ;; HIGH ORDER BITS GO HERE

```

1669 .SBTTL SCOPE HANDLER ROUTINE
1670
1671 ;*****
1672 ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1673 ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1674 ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1675 ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1676 ;$SW14=1 LOOP ON TEST
1677 ;$SW11=1 INHIBIT ITERATIONS
1678 ;$SW09=1 LOOP ON ERROR
1679 ;$SW08=1 LOOP ON TEST IN SWR<7:0>
1680 ;CALL
1681 ;* SCOPE ;;SCOPE=IOT
1682
1683 $SCOPE:
1684 010466 104406 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
1685 010470 104406 CKSWR
1686 010472 032777 040000 170440 1$: BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
1687 010500 001114 BNE $OVER ;;YES IF SW14=1
1688 ;*****START OF CODE FOR THE XOR TESTER*****
1689 010502 000416 $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1690 ;THIS INSTRUCTION TO A "NOP" (NOP=240)
1691 010504 013746 000004 MOV $#ERRVEC,-($P) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1692 010510 012737 010530 000004 MOV $S,$#ERRVEC ;;SET FOR TIMEOUT
1693 010516 005737 177060 TST $#177060 ;;TIME OUT ON XOR?
1694 010522 012637 000004 MOV ($P)+,$#ERRVEC ;;RESTORE THE ERROR VECTOR
1695 010526 000463 BR $SVLAD ;;GO TO THE NEXT TEST
1696 010530 022626 5$: CMP ($P)+,($P)+ ;;CLEAR THE STACK AFTER A TIME OUT
1697 010532 012637 000004 MOV ($P)+,$#ERRVEC ;;RESTORE THE ERROR VECTOR
1698 010536 000423 BR 7$ ;;LOOP ON THE PRESENT TEST
1699 010540 6$:;*****END OF CODE FOR THE XOR TESTER*****
1700 010540 032777 000400 170372 BIT #BIT08,$SWR ;;LOOP ON SPEC. TEST?
1701 010546 001404 BEQ 2$ ;;BR IF NO
1702 010550 127737 170364 001102 CMPB $SWR,$TSTNM ;;ON THE RIGHT TEST? SWR<7:0>
1703 010556 001465 BEQ $OVER ;;BR IF YES
1704 010560 105737 001103 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
1705 010564 001421 BEQ 3$ ;;BR IF NO
1706 010566 123737 001115 001103 CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
1707 010574 101015 BHI 3$ ;;BR IF NO
1708 010576 032777 001000 170334 BIT #BIT09,$SWR ;;LOOP ON ERROR?
1709 010604 001404 BEQ 4$ ;;BR IF NO
1710 010606 013737 001110 001106 7$: MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
1711 010614 000446 BR $OVER
1712 010616 105037 001103 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
1713 010622 005037 001166 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1714 010626 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
1715 010630 032777 004000 170302 3$: BIT #BIT11,$SWR ;;INHIBIT ITERATIONS?
1716 010636 001011 BNE 1$ ;;BR IF YES
1717 010640 005737 001210 TST $PASS ;;IF FIRST PASS OF PROGRAM
1718 010644 001406 BEQ 1$ ;;INHIBIT ITERATIONS
1719 010646 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
1720 010652 023737 001166 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
1721 010660 002024 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
1722 010662 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
1723 010670 013737 010746 001166 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
1724 010676 105237 001102 $SVLAD: INCB $TSTNM ;;COUNT TEST NUMBERS

```


M03

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 39
 DZLKA.P11 SCOPE HANDLER ROUTINE

1725	010702	113737	001102	001206		MOVB	\$TSTNM,\$TESTN	:::SET TEST NUMBER IN APT MAILBOX
1726	010710	011637	001106			MOV	(SP),\$LPADR	:::SAVE SCOPE LOOP ADDRESS
1727	010714	011637	001110			MOV	(SP),\$LPERR	:::SAVE ERROR LOOP ADDRESS
1728	010720	005037	001170			CLR	\$ESCAPE	:::CLEAR THE ESCAPE FROM ERROR ADDRESS
1729	010724	112737	000001	001115		MOVB	#1,\$ERMAX	:::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1730	010732	013777	001102	170202	\$OVER:	MOV	\$TSTNM,\$DISPLAY	:::DISPLAY TEST NUMBER
1731	010740	013716	001106			MOV	\$LPADR,(SP)	:::FUDGE RETURN ADDRESS
1732	010744	000002				RTI		:::FIXES PS
1733	010746	003720			\$MXCNT: 2000.			:::MAX. NUMBER OF ITERATIONS

```

1734 .SBTTL ERROR HANDLER ROUTINE
1735
1736 *****
1737 *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1738 *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1739 *AND GO TO $ERRTYP ON ERROR
1740 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1741 *SW15=1 HALT ON ERROR
1742 *SW13=1 INHIBIT ERROR TYPEOUTS
1743 *SW10=1 BELL ON ERROR
1744 *SW09=1 LOOP ON ERROR
1745 *CALL
1746 *
1747 *      ERROR      N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1748 $ERROR:
1749 010750 104406 001103 7$: CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
1750 010752 105237 001102 170154 INCB          $ERFLG          ;;SET THE ERROR FLAG
1751 010756 001775 002000 170144 BEQ          7$          ;;DON'T LET THE FLAG GO TO ZERO
1752 010760 013777 001102 170154 MOV          $STNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
1753 010766 032777 002000 170144 BIT          #BIT10,$SWR    ;;BELL ON ERROR?
1754 010774 001402 001172 1$: BEQ          1$          ;;NO - SKIP
1755 010776 104400 001112 170144 TYPE          $BELL          ;;RING BELL
1756 011002 005237 001116 170144 INC          $ERRTL        ;;COUNT THE NUMBER OF ERRORS
1757 011006 011637 001116 170144 MOV          (SP),$ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
1758 011012 162737 000002 001116 SUB          #2,$ERRPC
1759 011020 117737 170072 001114 MOVB         $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
1760 011026 032777 020000 170104 BIT          #BIT13,$SWR    ;;SKIP TYPEOUT IF SET
1761 011034 001004 011150 20$: BNE          20$          ;;SKIP TYPEOUTS
1762 011036 004737 001177 20$: JSR          PC,$ERRTYP    ;;GO TO USER ERROR ROUTINE
1763 011042 104400 001177 20$: TYPE          $CRLF
1764 011046 122737 000001 001222 CMPB          #APTENV,$ENV    ;;RUNNING IN APT MODE
1765 011054 001007 001114 011070 BNE          2$          ;;NO SKIP APT ERROR REPORT
1766 011056 113737 011604 21$: MOVB         $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
1767 011064 004737 001177 21$: JSR          PC,$ATY4        ;;REPORT FATAL ERROR TO APT
1768 011070 000 22$: .BYTE          0
1769 011071 000 22$: .BYTE          0
1770 011072 000777 170040 25$: BR          22$          ;;APT ERROR LOOP
1771 011074 005777 170040 25$: TST          $SWR          ;;HALT ON ERROR
1772 011100 100002 001110 3$: BPL          3$          ;;SKIP IF CONTINUE
1773 011102 000000 001170 4$: HALT          ;;HALT ON ERROR!
1774 011104 104406 001000 170024 3$: CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
1775 011106 032777 001110 170024 3$: BIT          #BIT09,$SWR    ;;LOOP ON ERROR SWITCH SET?
1776 011114 001402 001110 170024 3$: BEQ          4$          ;;BR IF NO
1777 011116 013716 001170 4$: MOV          $LPERR,(SP)    ;;FUDGE RETURN FOR LOOPING
1778 011122 005737 001170 4$: TST          $ESCAPE        ;;CHECK FOR AN ESCAPE ADDRESS
1779 011126 001402 001170 5$: BEQ          5$          ;;BR IF NONE
1780 011130 013716 004616 000042 5$: MOV          $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
1781 011134 022737 001001 000002 6$: CMP          #SENDAD,$42    ;;ACT-11 AUTO-ACCEPT?
1782 011142 001001 000000 6$: BNE          6$          ;;BRANCH IF NO
1783 011144 000000 6$: HALT          ;;YES
1784 011146 000002 6$: RTI          ;;RETURN
1785
1786
1787

```


804

```

1788
1789
1790
1791
1792
1793
1794
1795
1796 011150
1797 011150 104400 001177
1798 011154 010046
1799 011156 005000
1800 011160 153700 001114
1801 011164 001004
1802
1803 011166 013746 001116
1804
1805 011172 104401
1806 011174 000426
1807 011176 005300
1808 011200 006300
1809 011202 006300
1810 011204 006300
1811 011206 062700 001264
1812 011212 012037 011222
1813 011216 001404
1814 011220 104400
1815 011222 000000
1816 011224 104400 001177
1817 011230 012037 011240
1818 011234 001404
1819 011236 104400
1820 011240 000000
1821 011242 104400 001177
1822 011246 011000
1823 011250 001004
1824 011252 012600
1825 011254 104400 001177
1826 011260 000207
1827 011262
1828 011262 013046
1829 011264 104401
1830 011266 005710
1831 011270 001770
1832 011272 104400 011300
1833 011276 000771
1834 011300 020040 000
1835 011304

```

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

```

C04

1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891

.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$:      MOV       RD,-(SP)    ;; SAVE RD
        MOV       32(SP),RD   ;; GET ADDRESS OF ASCIZ STRING
        CMPB      #APTENV,SENV ;; RUNNING IN APT MODE
        BNE       62$        ;; NO, GO CHECK FOR APT CONSOLE
        BITB      #APTSPool,SENVM ;; SPOOL MESSAGE TO APT
        BEQ       62$        ;; NO, GO CHECK FOR CONSOLE
        MOV       RD,61$      ;; SETUP MESSAGE ADDRESS FOR APT
        JSR       PC,$ATY3    ;; SPOOL MESSAGE TO APT
        .WORD     0           ;; MESSAGE ADDRESS
        BITB      #APTCSUP,SENVM ;; APT CONSOLE SUPPRESSED
        BNE       60$        ;; YES, SKIP TYPE OUT
        MOVB      (RD)+,-(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
        MOV       (SP)+,RD    ;; RESTORE RD
        ADD       #2,(SP)     ;; ADJUST RETURN PC
        RTI              ;; RETURN
        CMPB      #HT,(SP)    ;; BRANCH IF <HT>
        BEQ       8$         ;; BRANCH IF NOT <CRLF>
        CMPB      #CRLF,(SP)
        BNE       5$         ;; POP <CR><LF> EQUIV
        TST       (SP)+       ;; TYPE A CR AND LF
        TYPE
        $CRLF
        CLRB      $CHARCNT    ;; CLEAR CHARACTER COUNT
        BR        2$         ;; GET NEXT CHARACTER
        JSR       PC,$TYPEC    ;; GO TYPE THIS CHARACTER
        CMPB      $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
        BNE       2$         ;; IF NO GO GET NEXT CHAR.
        MOV       $NULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
                                ;; AND THE NULL CHAR.
        DECB      1(SP)       ;; DOES A NULL NEED TO BE TYPED?
        BLT       6$         ;; BR IF NO--GO POP THE NULL OFF OF STACK
        JSR       PC,$TYPEC    ;; GO TYPE A NULL
        JSR       PC,$TYPEC

```


1892	011464	105337	011562	DECB	\$CHARCNT	::DO NOT COUNT AS A COUNT
1893	011470	000770		BR	7\$	::LOOP
1894						
1895				;HORIZONTAL TAB PROCESSOR		
1896						
1897	011472	112716	000040	8\$:	MOVB	8' (SP)
1898	011476	004737	011516	9\$:	JSR	PC,\$TYPEC
1899	011502	132737	000007		BITB	87,\$CHARCNT
1900	011510	001372			BNE	9\$
1901	011512	005726			TST	(SP)+
1902	011514	000724			BR	2\$
1903	011516	105777	167426	\$TYPEC:	TSTB	2\$TPS
1904	011522	100375			BPL	\$TYPEC
1905	011524	116677	000002		MOVB	2(SP),2\$TPB
1906	011532	122766	000015		CMPB	8CR,2(SP)
1907	011540	001003			BNE	1\$
1908	011542	105037	011562		CLRB	\$CHARCNT
1909	011546	000406			BR	\$TYPEX
1910	011550	122766	000012	1\$:	CMPB	8LF,2(SP)
1911	011556	001402			BEQ	\$TYPEX
1912	011560	105227			INCB	(PC)+
1913	011562	000000		\$CHARCNT:	WORD	0
1914	011564	000207		\$TYPEX:	RTS	PC
1915						

::REPLACE TAB WITH SPACE
 ::TYPE A SPACE
 ::BRANCH IF NOT AT
 ::TAB STOP
 ::POP SPACE OFF STACK
 ::GET NEXT CHARACTER
 ::WAIT UNTIL PRINTER IS READY
 ::LOAD CHAR TO BE TYPED INTO DATA REG.
 ::IS CHARACTER A CARRIAGE RETURN?
 ::BRANCH IF NO
 ::YES--CLEAR CHARACTER COUNT
 ::EXIT
 ::IS CHARACTER A LINE FEED?
 ::BRANCH IF YES
 ::COUNT THE CHARACTER
 ::CHARACTER COUNT STORAGE

```

1916
1917
1918
1919
1920 011566 112737 000001 012032 SATY1: MOV      #1,SFFLG      ;;TO REPORT FATAL ERROR
1921 011574 112737 000001 012030 SATY3: MOV      #1,SMFLG      ;;TO TYPE A MESSAGE
1922 011602 000403
1923 011604 112737 000001 012032 SATY4: MOV      #1,SFFLG      ;;TO ONLY REPORT FATAL ERROR
1924 011612 SATYC:
1925 011612 010046      MOV      RO,-(SP)      ;;PUSH RO ON STACK
1926 011614 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
1927 011616 105737 012030      TSTB     SMFLG      ;;SHOULD TYPE A MESSAGE?
1928 011622 001450      BEQ       5$      ;;IF NOT: BR
1929 011624 122737 000001 001222 CMPB     #APTENV,SENV      ;;OPERATING UNDER APT?
1930 011632 001031      BNE       3$      ;;IF NOT: BR
1931 011634 132737 000100 001223 BITB     #APTPOOL,SENV      ;;SHOULD SPOOL MESSAGES?
1932 011642 001425      BEQ       3$      ;;IF NOT: BR
1933 011644 017600 000004      MOV      24(SP),RO      ;;GET MESSAGE ADDR.
1934 011650 062766 000002 000004 ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
1935 011656 005737 001202      1$: TST      MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
1936 011662 001375      BNE       1$      ;;IF NOT: WAIT
1937 011664 010037 001216      MOV      RO,MSGAD      ;;PUT ADDR IN MAILBOX
1938 011670 105720      2$: TSTB     (RO)+      ;;FIND END OF MESSAGE
1939 011672 001376      BNE       2$
1940 011674 163700 001216      SUB      MSGAD,RO      ;;SUB START OF MESSAGE
1941 011700 006200      ASR      RO      ;;GET MESSAGE LGTH IN WORDS
1942 011702 010037 001220      MOV      RO,MSGGLT      ;;PUT LENGTH IN MAILBOX
1943 011706 012737 000004 001202 MOV      #4,MSGTYPE      ;;TELL APT TO TAKE MSG.
1944 011714 000413      BR       5$
1945 011716 017637 000004 011742 3$: MOV      24(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
1946 011724 062766 000002 000004 ADD      #2,4(SP)      ;;BUMP RETURN ADDRESS
1947 011732 013746 177776      MOV      177776,-(SP)      ;;PUSH 177776 ON STACK
1948 011736 004737 011304      JSR      PC,STYPE      ;;CALL TYPE MACRO
1949 011742 000000      4$: .WORD      0
1950 011744      5$:
1951 011744 105737 012032      10$: TSTB     SFFLG      ;;SHOULD REPORT FATAL ERROR?
1952 011750 001416      BEQ      12$      ;;IF NOT: BR
1953 011752 005737 001222      TST      SENV      ;;RUNNING UNDER APT?
1954 011756 001413      BEQ      12$      ;;IF NOT: BR
1955 011760 005737 001202      11$: TST      MSGTYPE      ;;FINISHED LAST MESSAGE?
1956 011764 001375      BNE      11$      ;;IF NOT: WAIT
1957 011766 017637 000004 001204      MOV      24(SP),SFATAL      ;;GET ERROR #
1958 011774 062766 000002 000004 ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
1959 012002 005237 001202      INC      MSGTYPE      ;;TELL APT TO TAKE ERROR
1960 012006 105037 012032      12$: CLRB     SFFLG      ;;CLEAR FATAL FLAG
1961 012012 105037 012031      CLRB     SLFLG      ;;CLEAR LOG FLAG
1962 012016 105037 012030      CLRB     SMFLG      ;;CLEAR MESSAGE FLAG
1963 012022 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
1964 012024 012600      MOV      (SP)+,RO      ;;POP STACK INTO RO
1965 012026 000207      RTS      PC      ;;RETURN
1966 012030      000      SMFLG: .BYTE      0      ;;MSG. FLAG
1967 012031      000      SLFLG: .BYTE      0      ;;LOG FLAG
1968 012032      000      SFFLG: .BYTE      0      ;;FATAL FLAG
1969      012034      .EVEN
1970      000200      APTSIZE=200
1971      000001      APTENV=001

```



```

1972      000100      APTSPool=100
1973      000040      APTCSUP=040
1974      .SBTTL      BINARY TO OCTAL (ASCII) AND TYPE
1975
1976      ;*****
1977      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1978      ;OCTAL (ASCII) NUMBER AND TYPE IT.
1979      ;STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1980      ;CALL:
1981      ;      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
1982      ;      TYPOS      ;:CALL FOR TYPEOUT
1983      ;      .BYTE      N      ;:N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1984      ;      .BYTE      M      ;:M=1 OR 0
1985      ;      ;:1=TYPE LEADING ZEROS
1986      ;      ;:0=SUPPRESS LEADING ZEROS
1987
1988      ;STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1989      ;STYPOS OR STYPOC
1990      ;CALL:
1991      ;      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
1992      ;      TYPON      ;:CALL FOR TYPEOUT
1993
1994      ;STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1995      ;CALL:
1996      ;      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
1997      ;      TYPOC      ;:CALL FOR TYPEOUT
1998
1999      012034 017646 000000      STYPOS: MOV      2(SP),-(SP)      ;:PICKUP THE MODE
2000      012040 116637 000001 012257      MOV      1(SP),SOFILL      ;:LOAD ZERO FILL SWITCH
2001      012046 112637 012261      MOV      (SP)+,SOMODE+1      ;:NUMBER OF DIGITS TO TYPE
2002      012052 062716 000002      ADD      #2,(SP)      ;:ADJUST RETURN ADDRESS
2003      012056 000406      BR      STYPON
2004      012060 112737 000001 012257      STYPOC: MOV      #1,SOFILL      ;:SET THE ZERO FILL SWITCH
2005      012066 112737 000006 012261      MOV      #6,SOMODE+1      ;:SET FOR SIX(6) DIGITS
2006      012074 112737 000005 012256      STYPON: MOV      #5,SOCNT      ;:SET THE ITERATION COUNT
2007      012102 010346      MOV      R3,-(SP)      ;:SAVE R3
2008      012104 010446      MOV      R4,-(SP)      ;:SAVE R4
2009      012106 010546      MOV      R5,-(SP)      ;:SAVE R5
2010      012110 113704 012261      MOV      SOMODE+1,R4      ;:GET THE NUMBER OF DIGITS TO TYPE
2011      012114 005404      NEG      R4
2012      012116 062704 000006      ADD      #6,R4      ;:SUBTRACT IT FOR MAX. ALLOWED
2013      012122 110437 012260      MOV      R4,SOMODE      ;:SAVE IT FOR USE
2014      012126 113704 012257      MOV      SOFILL,R4      ;:GET THE ZERO FILL SWITCH
2015      012132 016605 000012      MOV      12(SP),R5      ;:PICKUP THE INPUT NUMBER
2016      012136 005003      CLR      R3      ;:CLEAR THE OUTPUT WORD
2017      012140 006105      1$: ROL      R5      ;:ROTATE MSB INTO "C"
2018      012142 000404      BR      3$      ;:GO DO MSB
2019      012144 006105      2$: ROL      R5      ;:FORM THIS DIGIT
2020      012146 006105      ROL      R5
2021      012150 006105      ROL      R5
2022      012152 010503      MOV      R5,R3
2023      012154 006103      3$: ROL      R3      ;:GET LSB OF THIS DIGIT
2024      012156 105337 012260      DECB      SOMODE      ;:TYPE THIS DIGIT?
2025      012162 100016      BPL      7$      ;:BR IF NO
2026      012164 042703 177770      BIC      #177770,R3      ;:GET RID OF JUNK
2027      012170 001002      BNE      4$      ;:TEST FOR 0

```

G04

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 46
 DZLKA.P11 BINARY TO OCTAL (ASCII) AND TYPE

2028	012172	005704		TST	R4	:: SUPPRESS THIS 0?
2029	012174	001403		BEQ	5\$	:: BR IF YES
2030	012176	005204		4\$: INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
2031	012200	052703	000060	BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
2032	012204	052703	000040	5\$: BIS	#' ,R3	:: MAKE ASCII IF NOT ALREADY
2033	012210	110337	012254	MOVB	R3,8\$	:: SAVE FOR TYPING
2034	012214	104400	012254	TYPE	8\$	:: GO TYPE THIS DIGIT
2035	012220	105337	012256	7\$: DECB	\$OCNT	:: COUNT BY 1
2036	012224	003347		BGT	2\$	:: BR IF MORE TO DO
2037	012226	002402		BLT	6\$	:: BR IF DONE
2038	012230	005204		INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
2039	012232	000744		BR	2\$	:: GO DO THE LAST DIGIT
2040	012234	012605		6\$: MOV	(SP)+,R5	:: RESTORE R5
2041	012236	012604		MOV	(SP)+,R4	:: RESTORE R4
2042	012240	012603		MOV	(SP)+,R3	:: RESTORE R3
2043	012242	016666	000002 000004	MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
2044	012250	012616		MOV	(SP)+,(SP)	
2045	012252	000002		RTI		:: RETURN
2046	012254	000		8\$: .BYTE	0	:: STORAGE FOR ASCII DIGIT
2047	012255	000		.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
2048	012256	000		\$OCNT: .BYTE	0	:: OCTAL DIGIT COUNTER
2049	012257	000		\$OFILL: .BYTE	0	:: ZERO FILL SWITCH
2050	012260	000000		\$OMODE: .WORD	0	:: NUMBER OF DIGITS TO TYPE

2051
 2052
 2053
 2054
 2055
 2056
 2057
 2058
 2059
 2060
 2061
 2062
 2063
 2064
 2065
 2066
 2067
 2068
 2069
 2070
 2071
 2072
 2073
 2074
 2075
 2076
 2077
 2078
 2079
 2080
 2081
 2082
 2083
 2084
 2085
 2086
 2087

.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.

```

```

STRAP:  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    STRPAD(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
-----
STRPAD:
STYPE   ;;CALL=TYPE      TRAP+0(104400)  TTY TYPEOUT ROUTINE
STYPOC  ;;CALL=TYPOC     TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
STYPOS  ;;CALL=TYPOS     TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
STYPON  ;;CALL=TYPON     TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
STYPDS  ;;CALL=TYPDS     TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)

SGTSWR  ;;CALL=GTSMR     TRAP+5(104405)  GET SOFT-SWR SETTING

SCKSWR  ;;CALL=CKSMR     TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
SRDCHR  ;;CALL=RDCHR     TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
SRDLIN  ;;CALL=RDLIN     TRAP+10(104410) TTY TYPEIN STRING ROUTINE
SRDOCT  ;;CALL=RDOCT     TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY

```

```

012262 010046
012264 016600 000002
012270 005740
012272 111000
012274 006300
012276 016000 012304
012302 000200

012304
012304 011304
012306 012060
012310 012034
012312 012074
012314 010102

012316 007376

012320 007326
012322 007610
012324 007730
012326 010326

```

.SBTTL POWER DOWN AND UP ROUTINES

```

*****
:POWER DOWN ROUTINE

```

```

$PWRDN: MOV    $SILLUP, $PWRVEC    ;;SET FOR FAST UP
        MOV    $340, $PWRVEC+2    ;;PRIO:7
        MOV    R0, -(SP)           ;;PUSH R0 ON STACK
        MOV    R1, -(SP)           ;;PUSH R1 ON STACK
        MOV    R2, -(SP)           ;;PUSH R2 ON STACK
        MOV    R3, -(SP)           ;;PUSH R3 ON STACK
        MOV    R4, -(SP)           ;;PUSH R4 ON STACK
        MOV    R5, -(SP)           ;;PUSH R5 ON STACK
        MOV    $SWR, -(SP)         ;;PUSH $SWR ON STACK
        MOV    SP, $SAVR6         ;;SAVE SP
        MOV    $PWRUP, $PWRVEC    ;;SET UP VECTOR
        HALT
        BR      .-2                ;;HANG UP

```

```

*****
:POWER UP ROUTINE

```

```

$PWRUP: MOV    $SILLUP, $PWRVEC    ;;SET FOR FAST DOWN
        MOV    $SAVR6, SP          ;;GET SP
        CLR    $SAVR6             ;;WAIT LOOP FOR THE TTY
        15: INC    $SAVR6          ;;WAIT FOR THE INC
        BNE    15                 ;;OF WORD
        MOV    (SP)+, $SWR         ;;POP STACK INTO $SWR
        MOV    (SP)+, R5           ;;POP STACK INTO R5
        MOV    (SP)+, R4           ;;POP STACK INTO R4
        MOV    (SP)+, R3           ;;POP STACK INTO R3
        MOV    (SP)+, R2           ;;POP STACK INTO R2
        MOV    (SP)+, R1           ;;POP STACK INTO R1
        MOV    (SP)+, R0           ;;POP STACK INTO R0
        MOV    $PWRDN, $PWRVEC    ;;SET UP THE POWER DOWN VECTOR
        MOV    $340, $PWRVEC+2    ;;PRIO:7
        TYPE    $PWRMG: .WORD $POWER    ;;REPORT THE POWER FAILURE
        RTI                          ;;POWER FAIL MESSAGE POINTER
        SILLUP: HALT
        BR      .-2                ;;THE POWER UP SEQUENCE WAS STARTED
        $SAVR6: 0                    ;;BEFORE THE POWER DOWN WAS COMPLETE
        $POWER: .ASCIZ <15><12>"POWER" ;;PUT THE SP HERE

```

```

.EVEN
.END

```

```

2088
2089
2090
2091
2092
2093 012330 012737 012470 000024
2094 012336 012737 000340 000026
2095 012344 010046
2096 012346 010146
2097 012350 010246
2098 012352 010346
2099 012354 010446
2100 012356 010546
2101 012360 017746 166554
2102 012364 010637 012474
2103 012370 012737 012402 000024
2104 012376 000000
2105 012400 000776
2106
2107
2108
2109 012402 012737 012470 000024
2110 012410 013706 012474
2111 012414 005037 012474
2112 012420 005237 012474
2113 012424 001375
2114 012426 012677 166506
2115 012432 012605
2116 012434 012604
2117 012436 012603
2118 012440 012602
2119 012442 012601
2120 012444 012600
2121 012446 012737 012330 000024
2122 012454 012737 000340 000026
2123 012462 104400
2124 012464 012476
2125 012466 000002
2126 012470 000000
2127 012472 000776
2128 012474 000000
2129 012476 005015 047520 042527
2130 012504 000122
2131
2132 000001

```


ABASE = 160060	303#	417	458	624	625	626
ACDW1 = 000000	417	460				
ACDW2 = 000000	417					
ACPUOP= 000000	417	432				
ADDW0 = 000000	417					
ADDW1 = 000000	417					
ADDW10= 000000	417					
ADDW11= 000000	417					
ADDW12= 000000	417					
ADDW13= 000000	417					
ADDW14= 000000	417					
ADDW15= 000000	417					
ADDW2 = 000000	417					
ADDW3 = 000000	417					
ADDW4 = 000000	417					
ADDW5 = 000000	417					
ADDW6 = 000000	417					
ADDW7 = 000000	417					
ADDW8 = 000000	417					
ADDW9 = 000000	417					
ADEVCT= 000000	417	423				
ADEVN = 000000	417	459				
RENV = 000000	417	428				
REVM = 000000	417	429				
AFATAL= 000000	417	420				
AMADR1= 000000	417	445				
AMADR2= 000000	417	449				
AMADR3= 000000	417	452				
AMADR4= 000000	417	455				
AMAMS1= 000000	417	439				
AMAMS2= 000000	417	447				
AMAMS3= 000000	417	450				
AMAMS4= 000000	417	453				
AMSGAD= 000000	417	425				
AMSGLG= 000000	417	426				
AMSGTY= 000000	417	419				
AMTYP1= 000000	417	440				
AMTYP2= 000000	417	448				
AMTYP3= 000000	417	451				
AMTYP4= 000000	417	454				
APASS = 000000	417	422				
APRIOR= 000200	305#	417				
APTCU= 000040	1867	1973#				
APTEV= 000001	1765	1860	1929	1971#		
APTSIZ= 000200	679	1970#				
APTSPO= 000100	1862	1931	1972#			
ASWREG= 000000	417	430				
ATESTN= 000000	417	421				
AUNIT = 000000	417	424				
AUSW = 000000	417	431				
AVECT1= 100360	304#	417	456	627	628	
AVECT2= 000000	417	457				
BEGIN 001620	329	639#				
BEGIN1 004654	330	1137#				
BEGIN2 004710	331	1150#				
BITSAY 001606	633#	952#	955	956	961#	967# 969 970 975#

BIT0	=	000001	288#	737	765	952	967	1046	1056	1075	1164								
BIT00	=	000001	278#	288															
BIT01	=	000002	277#	287															
BIT02	=	000004	276#	286															
BIT03	=	000010	275#	285															
BIT04	=	000020	274#	284															
BIT05	=	000040	273#	283															
BIT06	=	000100	272#	282															
BIT07	=	000200	271#	281															
BIT08	=	000400	270#	280	1700														
BIT09	=	001000	269#	279	1708	1776													
BIT1	=	000002	287#	1165															
BIT10	=	002000	268#	1174	1753														
BIT11	=	004000	267#	1175	1715														
BIT12	=	010000	266#	1090	1176														
BIT13	=	020000	265#	1177	1760														
BIT14	=	040000	264#	851	864	876	889	891	892	913	914	940	1178	1686					
BIT15	=	100000	263#	787	800	814	827	840	889	892	914	940	1035	1179					
BIT2	=	000004	286#	1166															
BIT3	=	000010	285#	1167															
BIT4	=	000020	284#	1168															
BIT5	=	000040	283#	1169															
BIT6	=	000100	282#	1170															
BIT7	=	000200	281#	1171															
BIT8	=	000400	280#	1172															
BIT9	=	001000	279#	1173															
BPTVEC=		000014	295#																
CKSMR	=	104406	1684	1685	1749	1775	2084#												
CR	=	000015	203#	1906	1916														
CRLF	=	000200	204#	1877	1916														
DDISP	=	177570	210#	394	667														
DF1		007040	484	490	496	502	508	514	521	527	533	539	545	551	557				
			563	569	575	581	587	593	599	605	611	617	623	1374#					
DH1		006565	482	597	609	1340#													
DH2		006625	488	603	615	1346#													
DH3		006665	494	500	506	512	519	525	531	537	543	555	561	567	573				
			579	585	591	621	1352#												
DH4		006726	549	1358#															
DISPLA		001142	394#	667#	675#	1730#	1752#												
DISPRE		000174	326#	675															
DSMR	=	177570	209#	393	666														
DT1		006772	483	598	604	610	1366#												
DT2		007004	489	616	1368#														
DT3		007016	495	501	507	513	520	526	532	538	544	556	562	568	574				
			580	586	592	622	1370#												
DT4		007030	550	1372#															
EMTVEC=		000030	298#	650#	651#														
EM1		005036	481	1182#															
EM10		005363	524	1222#															
EM11		005435	530	1230#															
EM12		005476	536	1236#															
EM13		005545	542	1243#															
EM14		005617	548	1251#															
EM15		005644	554	1255#															
EM16		005675	560	1260#															
EM17		005723	566	1264#															

[illegible]

[illegible]

[illegible]

SATY4	011604	1768	1923#											
SAUT08	001134	390#	1428	1549										
SBASE	001256	458#	687	688	689									
SBDADR	001122	385#												
SBDAT	001126	387#	706*	707	717*	718	728*	729	739*	740	752*	753	766*	767
		777*	778	789*	790	802*	803	816*	817	829*	830	842*	843	853*
		854	866*	867	878*	879	890*	926*	942*	943*	944	957*	958	971*
		972	986*	987	995*	996	1008*	1009	1017*	1018	1030*	1031	1036*	1037
		1049*	1050	1057*	1058	1059	1070*	1071	1076*	1077*	1078	1366	1368	1370
		409#	1755	1788										
		460#												
SBELL	001172	1882*	1892*	1899	1908*	1913#								
SCDW1	001262	1420#	2084											
SCHARC	011562	373#	641	642	650	656	657	658						
SCKSMR	007326	405#	406#	407#										
SCNTAG	001100	405#	406#	407#										
SCN1 =	000002	403#	405											
SCN2 =	000004	1431	1544#											
SCN3 =	000002	1448	1543#											
SCNTLG	010053	432#												
SCNTLU	010046	411#	1459	1543	1669	1763	1788	1797	1816	1821	1825	1881	1916	
SCPUOP	001230	1573	1607	1615#										
SCRIF	001177	423#												
SDBLK	010316	459#												
SDEVCT	001212	1110	1119	1125#										
SDEVN	001260	1576	1611#											
SDOAGN	004626	339	1121#	1783										
SOTBL	010306	656	1112#											
SENDAD	004616	1114	1129#											
SENDCT	004564	1117	1128#											
SENDMG	004635	428#	1765	1860	1929	1953								
SENULL	004632	429#	679	1862	1867	1931								
SENV	001222	1091	1102#											
SENVN	001223	656*	1109#	1113										
SEOP	004530	376#	1674	1704	1706	1712*	1734	1750*	1788					
SEOPCT	004556	382#	659*	1706	1729*	1734								
SERFLG	001103	650	1748#											
SERMAX	001115	383#	1366	1368	1370	1372	1757*	1758*	1759	1788	1803			
SERROR	010750	477#	1811											
SERRPC	001116	1762	1796#											
SERRTB	001264	380#	1756*	1788										
SERTY	011150	408#	658*	1728*	1779	1781	1788							
SERTTL	001112	427#												
SESCAP	001170	365	461#											
SETABL	001222	420#	1957*											
SETEND	001264	1920*	1923*	1951	1960*	1968#								
SFATAL	001204	401#	1885	1916										
SFFLG	012032	400#	1916											
SFILLC	001156	384#												
SFILLS	001155	386#	635*	704*	705	707	715*	716	718	726*	727	729	737*	738
SGDADR	001120	740	743*	750*	751	753	757*	764*	767	775*	778	787*	788	790
SGDOAT	001124	799*	803	813*	817	826*	830	839*	843	851*	852	854	862*	867
		875*	879	889*	940*	941	944	954*	955*	958	969*	972	984*	987
		993*	996	1006*	1009	1015*	1018	1028*	1031	1037	1047*	1050	1055*	1058
		1059	1068*	1071	1078	1366	1368	1370	1372					
		1118#												
SGET42	004606													

[illegible]

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 57
DZLKA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

SR2A = ***** U	2088																
SSAVRE= ***** U	2088																
SSAVR6 012474	2102*	2110	2111*	2112*	2128#												
SSCOPE 010466	648	1683#															
SSETUP= 000137	479#	647	648	650	652	654	656	657	658	660	1104	1415	1549				
	1684	1749	1775	1783													
SSTUP = 177777	479#																
SSVLAD 010676	1695	1724#															
SSVPC = 000214	337#	342															
SSWR = 167400	183#	193	311	312	313	314	315	316	317	318	407	408	409				
	657	658	660	661	704	715	726	737	750	764	775	786	798				
	811	825	838	850	861	874	887	905	937	951	967	982	1004				
	1027	1045	1067	1090	1099	1105	1120	1126	1128	1137	1150	1675	1676				
	1677	1678	1679	1686	1698	1700	1701	1704	1705	1706	1713	1714	1715				
	1727	1730	1733	1740	1741	1742	1743	1744	1753	1760	1772	1776	1788				
	2125																
SSWREG 001224	430#	681															
SSWRNK= 000000	318	319	1679	1680	1702												
STESTN 001206	421#	1725*															
STINES 001166	407#	657*	811*	861*	1105*	1713*	1720	1723*	1733								
STKB 001146	396#	1413	1424	1441	1495	1501											
STKS 001144	395#	1413	1422	1438	1462*	1493	1499										
STN = 000035	183#	193	700	704#	708	711	715#	719	722	726#	730	733	737#				
	746	750#	760	764#	768	771	775#	779	782	786#	791	794	798#				
	804	807	811#	818	821	825#	831	834	838#	844	846	850#	855				
	857	861#	868	870	874#	880	883	887#	901	905#	920	928	933				
	937#	945	947	951#	963	967#	978	982#	997	1000	1004#	1019	1023				
	1027#	1038	1041	1045#	1060	1063	1067#	1086	1090#	1133	1137#	1146	1150#				
	398#	1905*	1916														
STPB 001152	402#	1854	1916														
STPFLG 001157	397#	1903	1916														
STPS 001150	652	2060#															
STRAP 012262	2068#	2077#	2078#	2079#	2080#	2081#	2082	2083#	2084	2085#	2086#	2087#	2088#				
STRP = 000012	2065	2075#															
STRPAD 012304	362#																
STSTM 001004	375#	1104#	1674	1702	1724*	1725	1730	1734	1752	1788							
STSTM 001102	1520	1521	1538	1542#													
STTYIN 010036	2081																
STYPB= ***** U	1561#	2080															
STYPDS 010102	1854#	1948	2068	2076													
STYPE 011304	1464	1884	1891	1898	1903#	1904											
STYPEC 011516	1909	1911	1914#														
STYPEX 011564	2004#	2077															
STYPOC 012060	2003	2006#	2079														
STYPON 012074	1999#	2078															
STYPOS 012034	424#																
SUNIT 001214	364#																
SUNITM 001010	431#																
SUSWR 001226	456#	692	693	906													
SVECT1 001252	457#																
SVECT2 001254	1689#																
SXTSTR 010502	1120#																
SSGET4= 000000	2000#	2004#	2014	2049#													
SOFILL 012257	1686	1762															
S4OCAT= ***** U	321#	325#	337	338#	340#	342#	343#	349	350#	352#	354#	372#	413				
	645	660	661	1128	1132	1364#	1408#	1413	1542#	1543	1549	1615#	1669				

E05

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 58
DZLKAA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

.SASTA= ***** U
.SX = 001000

1733	1734	1788	1835#	1916	1969#	2105	2127
1921	1924						
349#	354						

COMMON	18	303#														
ENDCOM	18	303#														
ERROR	197#	636	709	720	731	742	755	769	780	792	805	819	832	845	856	
	869	881	896	919	927	946	960	974	989	998	1011	1020	1033	1039	1052	
	1061	1073	1080													
ESCAPE	18	303#														
GETPRI	18	303#														
GETSMR	18	303#														
MULT	18	303#														
NEWST	18	303#	700	711	722	733	746	760	771	782	794	807	821	834	846	
	857	870	883	901	933	947	963	978	1000	1023	1041	1063	1086	1133	1146	
POP	18	303#	1602	1658	1963	1964	2114	2115								
PUSH	18	303#	1561	1632	1924	1926	1947	2095	2101							
REPORT	18	303#														
SCOPE	198#	703	714	725	736	749	763	774	785	797	810	824	837	849	860	
	873	886	904	936	950	966	981	1003	1026	1044	1066	1089	1103	1136	1149	
SETPRI	18	303#														
SETTRA	2068#	2077	2078	2079	2080	2082	2084	2085	2086	2087						
SETUP	18	303#	640													
SKIP	18	303#	708	719	730	741	754	768	779	791	804	818	831	844	855	
	868	880	897	920	928	945	997	1019	1032	1038	1051	1060	1072	1079	1091	
SLASH	18	303#														
SPACE	303#															
STARS	18	303#	335	346	348	355	368	413	416	700	702	711	713	722	724	
	733	735	746	748	760	762	771	773	782	784	794	796	807	809	821	
	823	834	836	846	848	857	859	870	872	883	885	901	903	933	935	
	947	949	963	965	978	980	1000	1002	1023	1025	1041	1043	1063	1065	1086	
	1088	1096	1133	1135	1146	1148	1412	1415	1483	1512	1551	1618	1671	1736	1791	
	1839	1919	1976	2054	2091	2107										
SMRSU	18	303#	662#													
TRMTRP	2068#															
TYPBIN	18	303#														
TYPDEC	18	303#	1115													
TYPNAM	18	303#														
TYPNUM	18	303#														
TYPOCS	18	303#														
TYPOCT	18	303#	1433	1803	1827											
TYPTXT	18	303#														
\$\$CHRE	366#	405	406													
\$\$CHTH	366#															
\$\$ESCA	18	303#														
\$\$NEMT	18	303#	700	711	722	733	746	760	771	782	794	807	821	834	846	
	857	870	883	901	933	947	963	978	1000	1023	1041	1063	1086	1133	1146	
\$\$SET	2068#	2077	2078	2079	2080	2082	2084	2085	2086	2087						
\$\$SETH	678#															
\$\$SKIP	18	303#	708	719	730	768	779	791	804	818	831	844	855	868	880	
	920	928	945	997	1019	1038	1060									
.EQUAT	18	183#	193													
.HEADE	18	183#														
.KT11	18															
.SETUP	18	183#	479													
.SMRHI	18	183#	307													
.SMRLO	319#															
.SACT1	18	183#	333													
.SAPT8	18	183#	414#													
.SAPTH	18	183#	344													

.SAPTY	18	183#	1917
.SASTA	18		
.SCATC	18	183#	319
.SCMTA	18	183#	366
.SDB2D	18		
.SDB20	18		
.SDIV	18		
.SEOP	18	183#	1094
.SERRO	18	183#	1734
.SERRT	18	183#	1789
.SMULT	18		
.SPARM	183#		
.SPOWE	18	183#	2089
.SRAND	18		
.SRDE	18		
.SRDOC	18	183#	1616
.SREAD	18	183#	1410
.SR2AZ	18		
.SSAVE	18	183#	
.SSB2D	18		
.SSB20	18		
.SSCOP	18	183#	1669
.SSIZE	18		
.SSPAC	183#		
.SSUPR	18		
.SSMDO	183#		
.STRAP	18	183#	2052
.STYPB	18		
.STYPD	18	183#	1549
.STYPE	18	183#	1837
.STYPO	18	183#	1974
.S4OCA	18		
.1170	18		

H05

 MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 63
 DZLKA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ADD	690	691	694	1160	1449	1458	1581	1653	1811	1873	1934	1946	1958	2002	2012
ASL	1472	1473	1474	1646	1648	1650	1808	1809	1810	2064					
ASLB	1586														
ASR	1941														
BCC	744	758	962	976	1587										
BEQ	680	708	719	730	741	754	768	779	791	804	818	831	844	855	868
	880	945	959	973	988	997	1010	1019	1032	1038	1051	1060	1072	1079	1091
	1119	1429	1456	1471	1641	1701	1703	1705	1709	1718	1751	1754	1777	1780	1813
	1818	1831	1863	1876	1911	1928	1932	1952	1954	2029					
BGE	1721														
BGT	1110	1468	1509	1595	1643	2036									
BHI	1707														
BIC	695	696	907	955	956	1077	1107	1158	1425	1442	1469	1496	1502	1510	1652
	2026														
BICB	1007	1016													
BIS	800	876		1476	1589	1590	2031	2032							
BISB	985	994	1800												
BIT	1090	1686	1700	1708	1715	1753	1760	1776							
BITB	679	1862	1867	1899	1931										
BLOS	1522														
BLT	1466	1507	1578	1594	1645	1890	2037								
BMI	1585														
BNE	645	669	684	1421	1427	1447	1454	1461	1498	1504	1526	1532	1583	1687	1716
	1761	1766	1784	1801	1823	1861	1868	1870	1878	1886	1900	1907	1930	1936	1939
	1956	2027	2113												
BPL	1142	1156	1423	1439	1494	1500	1569	1599	1773	1855	1904	2025			
BR	671	897	917	920	928	1144	1162	1450	1477	1479	1505	1528	1580	1597	1654
	1667	1689	1695	1698	1711	1714	1771	1806	1833	1857	1883	1893	1902	1909	1922
	1944	2003	2018	2039	2105	2127									
CLR	643	657	658	678	698	775	776	786	798	799	801	812	813	825	826
	838	839	850	862	863	874	875	877	887	890	893	899	905	926	931
	938	939	951	968	982	1004	1027	1028	1034	1045	1053	1054	1055	1067	1068
	1074	1081	1082	1083	1104	1105	1138	1151	1154	1436	1437	1572	1575	1638	1639
	1713	1728	1799	2016	2111										
CLRB	1533	1601	1663	1712	1882	1908	1960	1961	1962						
CMP	644	668	707	718	729	740	753	767	778	790	803	817	830	843	854
	867	879	898	918	929	944	958	972	987	996	1009	1018	1031	1037	1050
	1058	1059	1071	1078	1420	1426	1446	1453	1465	1467	1497	1503	1506	1508	1521
	1593	1696	1720	1783											
CMPB	1428	1460	1525	1531	1642	1644	1702	1706	1765	1860	1875	1877	1885	1906	1910
	1929														
DEC	1108	1807													
DECB	1889	1892	2024	2035											
ENT	197														
HALT	325	1092	1774	1785	1856	2104	2126								
INC	992	1014	1106	1475	1579	1719	1756	1959	2030	2038	2112				
INCB	1724	1750	1912												
IOT	198														
JMP	329	330	331	1126											
JSR	1121	1464	1762	1768	1865	1884	1891	1898	1948						
MOV	635	642	646	648	649	650	651	652	653	654	655	656	660	661	664
	665	666	667	672	674	675	676	681	687	688	689	692	693	697	704
	705	706	715	716	717	726	727	728	737	738	739	750	751	752	764
	765	766	777	787	788	789	802	811	814	816	827	828	829	840	841
	842	851	852	853	861	864	866	878	888	889	891	892	908	910	911
	912	913	914	921	922	923	930	937	940	941	942	943	952	953	954

	957	967	969	971	983	984	986	990	991	993	995	1005	1006	1008	1012
	1013	1015	1017	1029	1030	1035	1036	1046	1047	1048	1049	1056	1057	1069	1070
	1075	1076	1111	1115	1118	1137	1143	1150	1157	1159	1161	1433	1457	1462	1491
	1492	1519	1520	1535	1536	1537	1538	1562	1563	1564	1565	1566	1567	1568	1573
	1576	1596	1602	1603	1604	1605	1606	1608	1609	1630	1631	1632	1633	1634	1636
	1637	1656	1657	1658	1659	1660	1691	1692	1694	1697	1710	1722	1723	1726	1727
	1730	1731	1752	1757	1778	1781	1798	1803	1812	1817	1822	1824	1828	1858	1859
	1864	1872	1887	1925	1926	1933	1937	1942	1943	1945	1947	1957	1963	1964	1999
	2007	2008	2009	2015	2022	2040	2041	2042	2043	2044	2060	2061	2065	2093	2094
	2095	2096	2097	2098	2099	2100	2101	2102	2103	2109	2110	2114	2115	2116	2117
	2118	2119	2120	2121	2122										
NOVB	659	906	1424	1441	1495	1501	1524	1529	1571	1574	1588	1591	1600	1640	1725
	1729	1759	1767	1869	1897	1905	1920	1921	1923	2000	2001	2004	2005	2006	2010
	2013	2014	2033	2063											
NEG	1570	2011													
NOP	894	895	915	916	924	925	1122	1123	1124						
RESET	639	815	865	1120											
ROL	743	757	961	975	1647	1649	1651	2017	2019	2020	2021	2023			
RTI	637	673	1463	1511	1539	1610	1661	1732	1787	1874	2045	2125			
RTS	1826	1914	1965	2066											
SEC	756														
SUB	909	1577	1758	1940											
TRAP	2068	2077	2078	2079	2080	2082	2084	2085	2086	2087					
TST	683	1141	1155	1455	1470	1582	1592	1655	1662	1693	1717	1772	1779	1830	1871
	1879	1901	1935	1953	1955	2028	2062								
TSTB	1422	1438	1493	1499	1584	1598	1704	1854	1903	1927	1938	1951			
.ASCII	410	411													
.ASCIIZ	409	412	1129	1182	1189	1194	1199	1205	1211	1217	1222	1230	1236	1243	1251
	1255	1260	1264	1274	1284	1295	1306	1312	1318	1323	1328	1334	1340	1346	1352
	1358	1376	1387	1398	1543	1544	1545	1547	1834	2129					
.BLKB	1542														
.BLKW	1615														
.BYTE	375	376	381	382	390	391	399	400	401	402	428	429	439	440	447
	448	450	451	453	454	1128	1540	1541	1769	1770	1966	1967	1968	2046	2047
	2048	2049													
.DSABL	1480														
.ENABL	1	183	1413												
.END	2132														
.ENDC	188	197	289	303	315	317	318	319	330	336	340	342	347	349	356
	369	373	375	403	407	408	409	410	414	417	439	447	450	453	456
	457	458	459	460	463	479	646	647	650	652	654	656	657	658	660
	662	683	701	702	703	704	709	712	713	714	715	720	723	724	725
	726	731	734	735	736	737	742	747	748	749	750	755	761	762	763
	764	769	772	773	774	775	780	783	784	785	786	792	795	796	797
	798	805	808	809	810	811	812	819	822	823	824	825	832	835	836
	837	838	845	847	848	849	850	856	858	859	860	861	862	869	871
	872	873	874	881	884	885	886	887	898	902	903	904	905	921	929
	934	935	936	937	946	948	949	950	951	964	965	966	967	979	980
	981	982	998	1001	1002	1003	1004	1020	1024	1025	1026	1027	1033	1039	1042
	1043	1044	1045	1052	1061	1064	1065	1066	1067	1073	1080	1087	1088	1089	1090
	1092	1097	1098	1099	1101	1104	1110	1113	1114	1118	1120	1126	1128	1129	1132
	1134	1135	1136	1137	1147	1148	1149	1150	1413	1414	1416	1444	1480	1484	1512
	1513	1520	1522	1525	1527	1543	1549	1552	1619	1625	1669	1672	1675	1680	1686
	1688	1699	1702	1703	1704	1706	1708	1715	1719	1724	1726	1730	1733	1734	1737
	1740	1750	1757	1762	1763	1764	1772	1783	1787	1788	1792	1807	1836	1840	1869
	1920	1921	1924	1951	1966	1977	2055	2061	2064	2076	2077	2078	2079	2080	2081

	2082	2083	2084	2085	2086	2087	2088	2092	2101	2102	2108	2114	2115	2125	2132
.EQUIV	197	198	206	221	222	251	252	253	254	255	256	257	258	259	260
	279	280	281	282	283	284	285	286	287	288					
.EVEN	417	1364	1408	1835	1969	2131									
.IF	184	195	261	289	314	316	317	318	319	328	335	338	340	346	348
	355	368	372	374	403	407	408	409	413	414	416	439	447	450	453
	456	457	458	459	460	461	463	479	641	646	648	650	652	654	656
	657	658	660	678	700	702	704	708	711	713	715	719	722	724	726
	730	733	735	737	741	746	748	750	754	760	762	764	768	771	773
	775	779	782	784	786	791	794	796	798	804	807	809	811	812	818
	821	823	825	831	834	836	838	844	846	848	850	855	857	859	861
	862	868	870	872	874	880	883	885	887	897	901	903	905	920	928
	933	935	937	945	947	949	951	963	965	967	978	980	982	997	1000
	1002	1004	1019	1023	1025	1027	1032	1038	1041	1043	1045	1051	1060	1063	1065
	1067	1072	1079	1086	1088	1090	1091	1096	1097	1098	1099	1100	1101	1103	1109
	1112	1114	1118	1120	1126	1128	1129	1133	1135	1137	1146	1148	1150	1412	1414
	1415	1416	1444	1483	1484	1512	1520	1521	1525	1526	1542	1543	1549	1551	1618
	1621	1637	1671	1674	1679	1685	1686	1698	1700	1701	1702	1704	1705	1706	1715
	1717	1725	1727	1732	1733	1734	1736	1739	1750	1753	1760	1762	1763	1765	1772
	1776	1783	1787	1788	1791	1806	1822	1839	1860	1919	1921	1924	1951	1966	1976
	2054	2060	2064	2068	2077	2078	2079	2080	2081	2082	2084	2085	2086	2087	2088
.IFF	2091	2101	2102	2107	2114	2115	2123	2125	2129						
	195	314	317	318	319	336	340	342	347	349	356	369	372	375	403
	414	417	646	701	702	703	704	709	712	713	714	715	720	723	724
	725	726	731	734	735	736	737	741	747	748	749	750	754	761	762
	763	764	769	772	773	774	775	780	783	784	785	786	792	795	796
	797	798	805	808	809	810	811	819	822	823	824	825	832	835	836
	837	838	845	847	848	849	850	856	858	859	860	861	869	871	872
	873	874	881	884	885	886	887	897	902	903	904	905	921	929	934
	935	936	937	946	948	949	950	951	964	965	966	967	979	980	981
	982	998	1001	1002	1003	1004	1020	1024	1025	1026	1027	1032	1039	1042	1043
	1044	1045	1051	1061	1064	1065	1066	1067	1072	1079	1087	1088	1089	1090	1091
	1097	1100	1104	1110	1113	1128	1134	1135	1136	1137	1147	1148	1149	1150	1413
	1416	1484	1486	1491	1512	1513	1522	1526	1543	1552	1619	1672	1699	1702	1703
	1706	1733	1734	1737	1739	1753	1783	1788	1792	1807	1836	1840	1920	1977	2055
.IFT	2061	2092	2108	2125											
.IFTF	1486	1491	1642	1662	1669	1714	1763								
.IIF	1431	1484	1487	1638	1646	1668	1712	1762							
	183	188	193	311	312	313	315	318	319	325	413	417	647	650	656
	657	658	660	661	1098	1104	1105	1116	1128	1132	1413	1434	1535	1543	1549
	1669	1675	1676	1677	1678	1679	1680	1684	1713	1714	1730	1733	1734	1740	1741
	1742	1743	1744	1749	1775	1783	1788	1804	1829	1916	2076	2077	2078	2079	2080
.IRP	2082	2084	2085	2086	2087										
	479	700	711	722	733	746	760	771	782	794	807	821	834	846	857
	870	883	901	933	947	963	978	1000	1023	1041	1063	1086	1133	1146	1562
.LIST	1602	1632	1658	1685	1925	1926	1947	1963	1964	2095	2101	2114	2115		
	1	183	303	318	325	403	405	406	407	414	417	479	662	700	704
	711	715	722	726	733	737	746	750	760	764	771	775	782	786	794
	798	807	811	821	825	834	838	846	850	857	861	870	874	883	887
	901	905	933	937	947	951	963	967	978	982	1000	1004	1023	1027	1041
	1045	1063	1067	1086	1090	1104	1120	1133	1137	1146	1150	1512	1679	1783	2068
	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088		
.MACRO	1	319	366	678	2068										
.MCALL	183	303	414	662											
.MEXIT	462														
.NLIST	1	183	303	318	325	403	405	406	407	414	417	479	662	700	704

K05

MAINDEC-11-DZLKA-A MACY11 27(732) 25-SEP-76 09:47 PAGE 66
 DZLKAA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

	711	715	722	726	733	737	746	750	760	764	771	775	782	786	794
	798	807	811	821	825	834	838	846	850	857	861	870	874	883	887
	901	905	933	937	947	951	963	967	978	982	1000	1004	1023	1027	1041
	1045	1063	1067	1086	1090	1104	1120	1133	1137	1146	1150	1512	1679	1783	2068
.PAGE	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088		
.REM	366	463													
.REPT	1														
.SBTTL	325	405													
	193	307	319	328	333	344	366	414	463	640	700	711	722	733	746
	760	771	782	794	807	821	834	846	857	870	883	901	933	947	963
	978	1000	1023	1041	1063	1086	1094	1133	1146	1181	1410	1549	1616	1669	1734
.TITLE	1789	1837	1917	1974	2052	2068	2089								
.WORD	183														
	325	326	327	341	360	361	362	363	364	365	374	377	378	379	380
	383	384	385	386	387	388	389	392	393	394	403	405	406	419	420
	421	422	423	424	425	426	430	431	432	445	449	452	455	456	457
	458	459	460	1109	1112	1127	1665	1668	1815	1820	1866	1913	1949	2050	2124

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*,DZLKAA.SEG/SOL/CRF/PAGNUM/NL:TOC=DZLKAA.SML,DZLKAA.P11
 RUN-TIME: 36 44 5 SECONDS
 RUN-TIME RATIO: 156/86=1.8
 CORE USED: 33K (65 PAGES)

Spooler runtime 9 Seconds, 43 KCS, 247 disk reads, 3 disk writes, 62 pages

CONFIDENTIAL - SECURITY INFORMATION

[illegible]