

PPPPPPPPPPP	AAAAA	TTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH		
PPPPPPPPPPP	AAAAA	TTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH		
PPPPPPPPPPP	AAAAA	TTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH		
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	AAA	TTT	CCC	HHH	HHH
PPPPPPPPPPP	AAA	AAA	AAA	TTT	CCC	HHH	HHH
PPPPPPPPPPP	AAA	AAA	AAA	TTT	CCC	HHH	HHH
PPPPPPPPPPP	AAA	AAA	AAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCCCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCCCC	HHH	HHH
PPP	AAAAA	AAA	AAAAA	TTT	CCCCC	HHH	HHH

I
S
I
A
L
A
B
O
R
R
E
C
T
I
O
N
S

•
•
•
•

```

PPPPPPP      AAAAAA      TTTTTTTTTT      EEEEEEEEEEE      CCCCCCCC      000000
PPPPPPP      AAAAAA      TTTTTTTTTT      EEEEEEEEEEE      CCCCCCCC      000000
PP          PP      AA          AA      TT          TT          EE          CC          00          00
PP          PP      AA          AA      TT          TT          EE          CC          00          00
PP          PP      AA          AA      TT          TT          EE          CC          00          00
PP          PP      AA          AA      TT          TT          EE          CC          00          00
PPPPPPP      AA          AA      TT          TT          EEEEEEEEE      CC          00          00
PPPPPPP      AA          AA      TT          TT          EEEEEEEEE      CC          00          00
PP          AAAAAAAAAA      TT          TT          EE          CC          00          00
PP          AAAAAAAAAA      TT          TT          EE          CC          00          00
PP          AA          AA      TT          TT          EE          CC          00          00
PP          AA          AA      TT          TT          EE          CC          00          00
PP          AA          AA      TT          TT          EEEEEEEEEEE      CCCCCCCC      000000
PP          AA          AA      TT          TT          EEEEEEEEEEE      CCCCCCCC      000000

```

••••
••••
••••

```

LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SSSSSS
LL          II          SSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS

```

```

1  L 0001 0 MODULE PATECO (%IF %VARIANT EQL 1
2  0002 0      %THEN
3  0003 0          ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE,
4  0004 0          NONEXTERNAL = LONG_RELATIVE),
5  0005 0      %FI
6  0006 0      IDENT = 'V04-000'
7  0007 0      ) =
8  0008 1 BEGIN
9  0009 1
10 0010 1
11 0011 1 *****
12 0012 1 *
13 0013 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
14 0014 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
15 0015 1 *  ALL RIGHTS RESERVED.
16 0016 1 *
17 0017 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
18 0018 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
19 0019 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
20 0020 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
21 0021 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
22 0022 1 *  TRANSFERRED.
23 0023 1 *
24 0024 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
25 0025 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
26 0026 1 *  CORPORATION.
27 0027 1 *
28 0028 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
29 0029 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
30 0030 1 *
31 0031 1 *
32 0032 1 *****
33 0033 1
34 0034 1 **
35 0035 1 FACILITY:      PATCH
36 0036 1
37 0037 1 ABSTRACT:      THIS MODULE CONTAINS ROUTINES TO HANDLE ECO COMMANDS.
38 0038 1
39 0039 1 ENVIRONMENT:  PART OF THE IMAGE FILE PATCH UTILITY FOR VAX.
40 0040 1
41 0041 1 AUTHOR:      K.D. MORSE , CREATION DATE:      21-OCT-77
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1      V02-006 PCG0001      Peter George      02-FEB-1981
46 0046 1      Add require statement for LIB$:PATDEF.REQ
47 0047 1
48 0048 1      V02-005 KDM0027      KATHLEEN D. MORSE      04-DEC-1980
49 0049 1      Change action taken on CHECK [NOT] ECO commands to be:
50 0050 1      skip this ECO and scan until you find the next SET ECO command.
51 0051 1
52 0052 1      . : VERSION
53 0053 1  --

```

```

55 0054 1 |
56 0055 1 | TABLE OF CONTENTS:
57 0056 1 |
58 0057 1 |
59 0058 1 | FORWARD ROUTINE
60 0059 1 |     PAT$ECO_CMDS : NOVALUE;                | Handles ECO commands
61 0060 1 |
62 0061 1 |
63 0062 1 | INCLUDE FILES:
64 0063 1 |
65 0064 1 | LIBRARY 'SYS$LIBRARY:LIB.L32';            | System structure definitions
66 0065 1 | REQUIRE 'SRC$:PATPCT.REQ';                | Defines PSECTS
67 0105 1 | REQUIRE 'SRC$:PATGEN.REQ';                | Defines context bits
68 0327 1 | REQUIRE 'SRC$:PREFIX.REQ';                | Defines structure macros
69 0515 1 | REQUIRE 'SRC$:PATPRE.REQ';                | Defines PATCH structures
70 0678 1 | REQUIRE 'LIB$:PATDEF.REQ';                | Defines literals
71 0732 1 | REQUIRE 'LIB$:PATMSG.REQ';                | Defines error message codes
72 0906 1 | REQUIRE 'SRC$:BSTRUC.REQ';                | Defines basic structures
73 0982 1 | REQUIRE 'SRC$:LISTEL.REQ';                | Defines list structures
74 1024 1 | REQUIRE 'SRC$:VXSMAC.REQ';                | Defines TRUE and FALSE
75 1089 1 |
76 1090 1 |
77 1091 1 | MACROS:
78 1092 1 |
79 1093 1 |
80 1094 1 |
81 1095 1 | EQUATED SYMBOLS:
82 1096 1 |
83 1097 1 |
84 1098 1 |
85 1099 1 | OWN STORAGE:
86 1100 1 |
87 1101 1 |
88 1102 1 |
89 1103 1 | EXTERNAL REFERENCES:
90 1104 1 |
91 1105 1 | EXTERNAL
92 1106 1 |     PAT$GL_FLAGS,                          | CLI flags
93 1107 1 |     PAT$GL_ECO_UPD : BITVECTOR,            | /UPDATE qualifier ECO mask
94 1108 1 |     PAT$GB_EXEC_CMD : BYTE,                | Indicator whether or not to execute patch
95 1109 1 |     PAT$GB_ECOLVL : BYTE,                  | ECO level for current patch
96 1110 1 |     PAT$GL_IHPPTR : REF BLOCK[,BYTE],      | Pointer to patch section of image header
97 1111 1 |     PAT$GL_IMGHDR : REF BLOCK[,BYTE],      | Pointer to image header
98 1112 1 |     PAT$GL_CONTEXT : BITVECTOR,            | Context bits
99 1113 1 |     PAT$GL_HEAD_LST,                        | Listhead for command parameters
100 1114 1 |     PAT$GL_OLDNBK : BLOCK[,BYTE],          | Old image file name block
101 1115 1 |     PAT$GB_OLDNAME;                          | Name of old image file
102 1116 1 |

```

```

104 1117 1 GLOBAL ROUTINE PAT$ECO_CMDS : NOVALUE = ! HANDLES ALL ECO COMMANDS
105 1118 1
106 1119 1
107 1120 1 **
108 1121 1 FUNCTIONAL DESCRIPTION:
109 1122 1 THIS ROUTINE HANDLES THE COMMANDS--CHECK ECO, CHECK NOT ECO, AND SET ECO.
110 1123 1 THE ECO LEVEL NUMBERS HAVE ALREADY BEEN SCANNED AND INSERTED IN LIST ENTRIES.
111 1124 1 ONE ENTRY CONSISTS OF THREE LONG WORDS:
112 1125 1
113 1126 1 1.) THE FORWARD LINK
114 1127 1 2.) THE ECO LEVEL (OR THE START OF THE ECO RANGE)
115 1128 1 3.) ZERO (OR THE END OF THE ECO RANGE)
116 1129 1
117 1130 1 NOTE THAT THE SECOND AND THIRD LONGWORDS DESCRIBE A SINGLE ECO LEVEL OR
118 1131 1 A RANGE OF LEVELS, WITH THE SECOND LONGWORD BEING THE MINIMUM AND THE THIRD
119 1132 1 BEING THE MAXIMUM. IF THERE IS ONLY A SINGLE ECO LEVEL THEN THE THIRD
120 1133 1 LONGWORD IS A ZERO.
121 1134 1
122 1135 1 THIS ROUTINE SETS UP THE POINTERS TO ACCESS THE ECO BITS IN THE IMAGE HEADER
123 1136 1 AND THE ECO LEVELS FROM THE COMMAND. THE CONTEXT BITS, SET ECO AND
124 1137 1 SET NOT ECO, SPECIFY WHETHER THE COMMAND WAS "SET ECO" OR "CHECK NOT ECO",
125 1138 1 RESPECTIVELY. IF NEITHER BIT IS SET, THEN THE COMMAND MUST HAVE BEEN "CHECK ECO".
126 1139 1
127 1140 1 IF THE APPROPRIATE ECO BITS ARE ALREADY SET, OR NOT SET, THEN AN ERROR
128 1141 1 MESSAGE IS PRODUCED. THE "SET ECO" COMMAND CHECKS THAT THE ECO LEVEL WAS
129 1142 1 NOT ALREADY SET. IF THE BIT IS NOT SET, THEN ANOTHER CHECK IS MADE TO SEE IF
130 1143 1 A PREVIOUS "SET ECO" WAS SPECIFIED BEFORE AN "UPDATE" COMMAND. THIS IS DONE
131 1144 1 BY CHECKING PAT$GB_ECOLVL. THIS INDICATOR IS SET BY THE "SET ECO" COMMAND
132 1145 1 AND CLEARED BY THE "UPDATE" COMMAND. IF IT CONTAINS A NON-ZERO VALUE WHEN
133 1146 1 THIS ROUTINE EXECUTES, THEN TWO "SET ECO" COMMANDS WERE SPECIFIED FOR ONE
134 1147 1 PATCH.
135 1148 1
136 1149 1 FORMAL PARAMETERS:
137 1150 1
138 1151 1 NONE
139 1152 1
140 1153 1 IMPLICIT INPUTS:
141 1154 1
142 1155 1 THE CONTEXT BITS ARE SET AND THE COMMAND PARAMETERS ARE PARSED.
143 1156 1 THE IMAGE HEADER HAS BEEN READ AND EXPANDED TO INCLUDE A PATCH SECTION,
144 1157 1 IF NECESSARY.
145 1158 1
146 1159 1 IMPLICIT OUTPUTS:
147 1160 1
148 1161 1 THE APPROPRIATE ECO BITS ARE SET OR CHECKED IN THE IMAGE HEADER.
149 1162 1
150 1163 1 ROUTINE VALUE:
151 1164 1
152 1165 1 NONE
153 1166 1
154 1167 1 COMPLETION CODES:
155 1168 1
156 1169 1 NONE
157 1170 1
158 1171 1 SIDE EFFECTS:
159 1172 1
160 1173 1 AN ERROR MESSAGE AND EXIT OCCUR IF THE APPROPRIATE ECO BITS ARE NOT SET.

```

```

161 1174 1 1  !
162 1175 1 1  !--
163 1176 1 1  !
164 1177 2 2  BEGIN
165 1178 2 2  LOCAL
166 1179 2 2  LOOP_CNT,          ! LOOP COUNTER (MIN ECO LEVEL)
167 1180 2 2  LOOP_MAX,        ! LOOP MAX (MAX ECO LEVEL)
168 1181 2 2  LIST_PTR,       ! POINTER TO LIST ELEMENTS
169 1182 2 2  ECO_PTR : REF TVECTOR; ! POINTER TO HEADER ECO WORDS
170 1183 2 2  !
171 1184 2 2  !++
172 1185 2 2  ! INITIALIZE POINTERS TO THE ECO BITS IN THE IMAGE HEADER AND TO THE FIRST
173 1186 2 2  ! COMMAND PARAMETER, AN ECO LEVEL OR RANGE.
174 1187 2 2  !--
175 1188 2 2  ECO_PTR = CH$PTR(PAT$GL_IHPTR[IHP$L_ECO1], 0); ! POINT TO FIRST ECO LONGWORD
176 1189 2 2  LIST_PTR = CH$PTR(.PAT$GL_HEAD_LST, 0); ! POINT TO FIRST ECO LEVEL PARAMETER
177 1190 2 2  !
178 1191 2 2  !++
179 1192 2 2  ! LOOP TO HANDLE ALL ECO LEVELS AND RANGES FOR THIS COMMAND.
180 1193 2 2  !--
181 1194 2 2  WHILE .LIST_PTR NEQA 0
182 1195 2 2  DO
183 1196 2 2  BEGIN
184 1197 2 2  !++
185 1198 2 2  ! SET UP A LOOP TO HANDLE A SINGLE ECO LEVEL OR RANGE OF ECO LEVELS.
186 1199 2 2  !--
187 1200 2 2  LOOP_CNT = .LIST_ELEM_EXP1(.LIST_PTR) - 1; ! INITIALIZE LOOP COUNT
188 1201 2 2  IF .LIST_ELEM_EXP2(.LIST_PTR) EQ 0 ! CHECK IF MAXIMUM RANGE
189 1202 2 2  THEN ! EXISTS,
190 1203 2 2  LOOP_MAX = .LOOP_CNT ! IF NOT, SET MAX TO MIN AS SINGLE ECO LEVEL
191 1204 2 2  ELSE
192 1205 2 2  LOOP_MAX = .LIST_ELEM_EXP2(.LIST_PTR) - 1; ! IF SO, SET LOOP MAXIMUM (MAX ECO RANGE)
193 1206 2 2  !
194 1207 2 2  !++
195 1208 2 2  ! CHECK FOR ERRORS IN RANGE OF ECO LEVELS.
196 1209 2 2  !--
197 1210 2 2  IF .LOOP_MAX LSS .LOOP_CNT ! CHECK FOR REVERSED RANGE
198 1211 2 2  THEN
199 1212 2 2  SIGNAL(PAT$ EXARANGE); ! REPORT ERROR
200 1213 2 2  IF .LOOP_MAX GTR (PAT$K_MAX_ECO - 1) ! CHECK FOR ILLEGAL ECO LEVEL
201 1214 2 2  THEN ! IF > MAX, REPORT ERROR
202 1215 2 2  SIGNAL(PAT$ BADECO, 3, .LOOP_MAX+1, .PAT$GL_OLDNBK[NAM$B_RSL],
203 1216 2 2  PAT$GB_OLDNAME);
204 1217 2 2  IF .LOOP_CNT LSS (PAT$K_MIN_ECO - 1) ! CHECK FOR ILLEGAL ECO LEVEL
205 1218 2 2  THEN ! IF < MIN, REPORT ERROR
206 1219 2 2  SIGNAL(PAT$ BADECO, 3, .LOOP_CNT+1, .PAT$GL_OLDNBK[NAM$B_RSL],
207 1220 2 2  PAT$GB_OLDNAME);
208 1221 2 2  !
209 1222 2 2  WHILE .LOOP_CNT LEQ .LOOP_MAX ! LOOP FOR ONE RANGE
210 1223 2 2  DO
211 1224 2 2  BEGIN
212 1225 2 2  !
213 1226 2 2  !++
214 1227 2 2  ! NOW HANDLE THE "SET ECO" COMMANDS, BY TESTING FOR THE
215 1228 2 2  ! SET_ECO CONTEXT BIT. THEN TEST IF ANOTHER ECO LEVEL IS SET.
216 1229 2 2  !--
217 1230 2 2  IF .PAT$GL_CONTEXT[SET_ECO]

```

```

218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274

```

```

1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287

```

THEN

```

BEGIN
IF .ECO_PTR[.LOOP_CNT]          ! CHECK ECO BIT NOT ALREADY SET
THEN
    BEGIN
    SIGNAL(PAT$ ECOSET, 3, .LOOP_CNT+1, .PAT$GL_OLDNBK[NAM$B_RSL],
          PAT$GB_OLDNAME);      ! REPORT ERROR
    PAT$GB_ECOLVL = 0;
    PAT$GB_EXEC_CMD = FALSE;
    RETURN;
    END;
IF .PAT$GB_ECOLVL NEQ 0
THEN
    SIGNAL(PAT$ MULTECO, 2, .PAT$GB_ECOLVL, .LOOP_CNT+1);
    PAT$GB_ECOLVL = .LOOP_CNT+1;      ! REMEMBER ECO LEVEL FOR 'UPDATE' COMMAND
    PAT$GB_EXEC_CMD = TRUE;          ! FORCE COMMANDS TO BE EXECUTED
    IF (.PAT$GL_FLAGS AND PAT$M_UPDATE) NEQ 0
    THEN
        BEGIN
        IF NOT .PAT$GL_ECO_UPD<.LOOP_CNT, 1> ! IF NO /UPDATE
        THEN                                ! THEN RESET THE INDICATOR
            BEGIN                          ! FOR NO EXECUTION
            PAT$GB_ECOLVL = 0;
            PAT$GB_EXEC_CMD = FALSE;
            SIGNAL(PAT$ UPDATE, 1, .LOOP_CNT+1);
            END;
        END;
    END;

```

ELSE

```

++
COMMAND WAS NOT 'SET ECO', THEREFOR IT MUST BE EITHER
'CHECK NOT ECO' OR 'CHECK ECO'. THE SET_NOT_ECO CONTEXT
BIT WILL TELL WHICH COMMAND IT WAS.
--
IF .PAT$GL_CONTEXT[SET_NOT_ECO]    ! 'CHECK NOT ECO' COMMAND?
THEN                                ! IF SET THEN YES
    BEGIN
    IF .ECO_PTR[.LOOP_CNT]          ! CHECK ECO LEVEL BIT NOT SET
    THEN                            ! IF IT IS, REPORT ERROR
        BEGIN
        SIGNAL(PAT$ ECOSET, 3, .LOOP_CNT+1,
              .PAT$GL_OLDNBK[NAM$B_RSL], PAT$GB_OLDNAME);
        PAT$GB_ECOLVL = 0;
        PAT$GB_EXEC_CMD = FALSE;
        RETURN;
        END;
    END;
ELSE
    IF NOT .ECO_PTR[.LOOP_CNT]      ! COMMAND WAS 'CHECK ECO'
    THEN                            ! REPORT ERROR IF NOT SET
        BEGIN
        SIGNAL(PAT$ ECONOTSET, 3, .LOOP_CNT+1,
              .PAT$GL_OLDNBK[NAM$B_RSL], PAT$GB_OLDNAME);
        PAT$GB_ECOLVL = 0;
        PAT$GB_EXEC_CMD = FALSE;
        RETURN;
        END;

```

```

: 275      1288  4
: 276      1289  3
: 277      1290  3
: 278      1291  2
: 279      1292  2 RETURN
: 280      1293  1 END;

```

```

      LOOP_CNT = .LOOP_CNT + 1;
      END;
LIST_PTR = CHSPTR( .LIST_ELEM_FLINK(.LIST_PTR), 0);
      END;

```

! END OF PATSECO_CMDS

				.TITLE	PATECO				
				.IDENT	\V04-000\				
				ISE\$C_SIZE==	20				
				TXT\$C_SIZE==	4				
				PAL\$C_SIZE==	16				
				ASD\$C_SIZE==	9				
				FWR\$C_SIZE==	24				
				.EXTRN	PAT\$GL_FLAGS, PAT\$GL_ECO_UPD				
				.EXTRN	PAT\$GB_EXEC_CMD				
				.EXTRN	PAT\$GB_ECOLVL, PAT\$GL_IHPPTR				
				.EXTRN	PAT\$GL_IMGHDR, PAT\$GL_CONTEXT				
				.EXTRN	PAT\$GL_HEAD_LST				
				.EXTRN	PAT\$GL_OLDNBK, PAT\$GB_OLDNAME				
				.WEAK	ACCESS_CHECK				
				.PSECT	_PAT\$CODE, NOWRT, 2				
				.ENTRY	PAT\$ECO_CMDS, Save R2,R3,R4,R5,R6,R7,R8,R9,-;	1117			
					R10				
		5A	00000000G	EF	9E	00002	MOVAB	PAT\$GB_EXEC_CMD, R10	
		59	00000000G	EF	9E	00009	MOVAB	PAT\$GB_ECOLVL, R9	
		58	00000000G	EF	9E	00010	MOVAB	PAT\$GL_OLDNBK+3, R8	
		57	00000000G	EF	9E	00017	MOVAB	PAT\$GB_OLDNAME, R7	
		56	00000000G	00	9E	0001E	MOVAB	LIB\$SIGNAL, R6	
		55	00000000G	EF	D0	00025	MOVL	PAT\$GL_IHPPTR, ECO_PTR	1188
		53	00000000G	EF	D0	0002C	MOVL	PAT\$GL_HEAD_LST, LIST_PTR	1189
				01	12	00033	1\$: BNEQ	2\$	1194
					04	00035	RET		
52	04	A3		01	C3	00036	2\$: SUBL3	#1, 4(LIST_PTR), LOOP_CNT	1200
			08	A3	D5	00038	TSTL	8(LIST_PTR)	1201
				05	12	0003E	BNEQ	3\$	
		54		52	D0	00040	MOVL	LOOP_CNT, LOOP_MAX	1203
				05	11	00043	BRB	4\$	
54	08	A3		01	C3	00045	3\$: SUBL3	#1, 8(LIST_PTR), LOOP_MAX	1205
		52		54	D1	0004A	4\$: CMPL	LOOP_MAX, LOOP_CNT	1210
				09	18	0004D	BGEQ	5\$	
			006D80AA	8F	DD	0004F	PUSHL	#7176362	1212
		66		01	FB	00055	CALLS	#1, LIB\$SIGNAL	
			0000007F	8F	D1	00058	5\$: CMPL	LOOP_MAX, #127	1213
				13	15	0005F	BLEQ	6\$	
				57	DD	00061	PUSHL	R7	1215
		7E		68	9A	00063	MOVZBL	PAT\$GL_OLDNBK+3, -(SP)	
			01	A4	9F	00066	PUSHAB	1(Loop_max)	
				03	DD	00069	PUSHL	#3	
			006D809A	8F	DD	0006B	PUSHL	#7176346	
		66		05	FB	00071	CALLS	#5, LIB\$SIGNAL	
				52	D5	00074	6\$: TSTL	LOOP_CNT	1217
				13	18	00076	BGEQ	7\$	

		57	DD	00078	PUSHL	R7		1219
7E		68	9A	0007A	MOVZBL	PAT\$GL_OLDNBK+3, -(SP)		
	01	A2	9F	0007D	PUSHAB	1(LOOP_CNT)		
		03	DD	00080	PUSHL	#3		
	006D809A	8F	DD	00082	PUSHL	#7176346		
66		05	FB	00088	CALLS	#5, LIB\$SIGNAL		
54		52	D1	0008B	7\$:	CMPL	LOOP_CNT, LOOP_MAX	1222
		03	15	0008E	BLEQ	8\$		
		008D	31	00090	BRW	15\$		
46 00000000G	EF	02	E1	00093	8\$:	BBC	#2, PAT\$GL_CONTEXT+2, 10\$	1230
4E	65	52	E0	0009B	BBS	LOOP_CNT, (TECO_PTR), 11\$		1233
	50	69	9A	0009F	MOVZBL	PAT\$GB_ECOLVL, -R0		1242
		10	13	000A2	BEQL	9\$		
		01	A2	9F	000A4	PUSHAB	1(LOOP_CNT)	1244
		50	DD	000A7	PUSHL	R0		
		02	DD	000A9	PUSHL	#2		
	006D80F2	8F	DD	000AB	PUSHL	#7176434		
66		04	FB	000B1	CALLS	#4, LIB\$SIGNAL		
50	01	A2	9E	000B4	9\$:	MOVAB	1(R2), R0	1245
69		50	90	000B8	MOVB	R0, PAT\$GB_ECOLVL		
6A		01	90	000BB	MOVB	#1, PAT\$GB_EXEC_CMD		1246
55 00000000G	EF	04	E1	000BE	BBC	#4, PAT\$GL_FLAGS, 14\$		1247
4D 00000000G	EF	52	E0	000C6	BBS	LOOP_CNT, PAT\$GL_ECO_UPD, 14\$		1250
		69	94	000CE	CLRB	PAT\$GB_ECOLVL		1253
		6A	94	000D0	CLRB	PAT\$GB_EXEC_CMD		1254
		50	DD	000D2	PUSHL	R0		1255
		01	DD	000D4	PUSHL	#1		
	006D803B	8F	DD	000D6	PUSHL	#7176251		
66		03	FB	000DC	CALLS	#3, LIB\$SIGNAL		
		3A	11	000DF	BRB	14\$		1230
16 00000000G	EF	01	E1	000E1	10\$:	BBC	#1, PAT\$GL_CONTEXT, 12\$	1265
2E	65	52	E1	000E9	BBC	LOOP_CNT, (TECO_PTR), 14\$		1268
		57	DD	000ED	11\$:	PUSHL	R7	1271
	7E	68	9A	000EF	MOVZBL	PAT\$GL_OLDNBK+3, -(SP)		1272
		01	A2	9F	000F2	PUSHAB	1(LOOP_CNT)	1271
		03	DD	000F5	PUSHL	#3		
	006D804B	8F	DD	000F7	PUSHL	#7176267		
		14	11	000FD	BRB	13\$		
18	65	52	E0	000FF	12\$:	BBS	LOOP_CNT, (ECO_PTR), 14\$	1279
		57	DD	00103	PUSHL	R7		1282
	7E	68	9A	00105	MOVZBL	PAT\$GL_OLDNBK+3, -(SP)		1283
		01	A2	9F	00108	PUSHAB	1(LOOP_CNT)	1282
		03	DD	0010B	PUSHL	#3		
	006D8043	8F	DD	0010D	PUSHL	#7176259		
66		05	FB	00113	13\$:	CALLS	#5, LIB\$SIGNAL	
		69	94	00116	CLRB	PAT\$GB_ECOLVL		1284
		6A	94	00118	CLRB	PAT\$GB_EXEC_CMD		1285
		04	0011A		RET			1281
		52	D6	0011B	14\$:	INCL	LOOP_CNT	1288
		FF6B	31	0011D	BRW	7\$		1222
	53	63	D0	00120	15\$:	MOVL	(LIST_PTR), LIST_PTR	1290
		FF0D	31	00123	BRW	1\$		1194
		04	00126		RET			1293

; Routine Size: 295 bytes, Routine Base: _PAT\$CODE + 0000

: 282 1294 1 END
: 283 1295 0 ELUDOM

!End of module

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
_PAT\$CODE	295	NOVEC,NOWRT, RD, EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)
ABS	0	NOVEC,NOWRT,NORD,NOEXE,NOSHR, LCL, ABS, CON,NOPI,ALIGN(0)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	6	0	1000	00:01.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LIS\$:PATECO/OBJ=OBJ\$:PATECO MSRC\$:PATECO/UPDATE=(ENHS:PATECO)

: Size: 295 code + 0 data bytes
: Run Time: 00:19.0
: Elapsed Time: 00:54.9
: Lines/CPU Min: 4100
: Lexemes/CPU-Min: 47759
: Memory Used: 190 pages
: Compilation Complete

0300 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 100 terminal windows, arranged in 10 rows and 10 columns. Each window contains a different screen from the VAX/VMS operating system. The screens are densely packed with text, including headers, data tables, and status information. Several windows are clearly labeled with titles such as 'PATARI LIS', 'PATCMD LIS', 'PATECO LIS', 'PATCON LIS', 'PATENC LIS', 'PATBAS LIS', and 'PATBLD LIS'. The overall appearance is that of a multi-user environment where multiple users are simultaneously running various applications and viewing their results on a shared display.