

PPPPPPPPPPPP		AAAAAAAAAA	TTTTTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH
PPPPPPPPPPPP		AAAAAAAAAA	TTTTTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH
PPPPPPPPPPPP		AAAAAAAAAA	TTTTTTTTTTTTTTTT	CCCCCCCCCCCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPP	PPP	AAA	TTT	CCC	HHH	HHH
PPPPPPPPPPPP		AAA	TTT	CCC	HHH	HHH
PPPPPPPPPPPP		AAA	TTT	CCC	HHHHHHHHHHHHHHHH	HHHHHHHHHHHHHHHH
PPPPPPPPPPPP		AAA	TTT	CCC	HHHHHHHHHHHHHHHH	HHHHHHHHHHHHHHHH
PPP		AAAAAAAAAAAAAAAA	TTT	CCC	HHH	HHH
PPP		AAAAAAAAAAAAAAAA	TTT	CCC	HHH	HHH
PPP		AAAAAAAAAAAAAAAA	TTT	CCC	HHH	HHH
PPP		AAA	TTT	CCC	HHH	HHH
PPP		AAA	TTT	CCC	HHH	HHH
PPP		AAA	TTT	CCC	HHH	HHH
PPP		AAA	TTT	CCC	HHH	HHH
PPP		AAA	TTT	CCCCCCCCCCCC	HHH	HHH
PPP		AAA	TTT	CCCCCCCCCCCC	HHH	HHH
PPP		AAA	TTT	CCCCCCCCCCCC	HHH	HHH


```
1 L 0001 0 MODULE PATERR ( %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: ERROR HANDLER
38 0038 1
39 0039 1 ENVIRONMENT: IMAGE FILE PATCH UTILITY
40 0040 1
41 0041 1 AUTHOR: K.D. MORSE , CREATION DATE: 5-OCT-77
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1 V03-002 MTR0025 Mike Rhodes 11-Aug-1983
46 0046 1 Add OVERLAY message definition to shared message declarations.
47 0047 1 This message is issued when a file is being updated, as a
48 0048 1 result of the user's request /ABSOLUTE and /NONEW_VERSION
49 0049 1 qualifiers on the DCL command line.
50 0050 1
51 0051 1 V03-001 MTR0008 Mike Rhodes 15-Jun-1982
52 0052 1 Modify JNL_ERRMSG to return a value (to indicate
53 0053 1 whether or not the message has been written) to
54 0054 1 correspond with $PUTMSG protocol.
55 0055 1
56 0056 1 A new GLOBAL ROUTINE has been added to support the use of the
57 0057 1 shared messages. GETFILDSC (fab) has been added to generate
```


PATERR
V04-000

K 2
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1 Page 2 (1)

: 58
: 59
: 60
: 61
: 62
: 63
: 64
: 65
: 66
: 67
: 68
: 69
: 70

0058 1
0059 1
0060 1
0061 1
0062 1
0063 1
0064 1
0065 1
0066 1
0067 1
0068 1
0069 1
0070 1

and return the address of a file descriptor containing the file
name of the problem file.

Add shared message definitions.

V02-008 PCG0001 Peter George 02-FEB-1981
Add require statement for LIB\$:PATDEF.REQ

V02-007 KDM0027 Kathleen D. Morse 04-Dec-1980
Remove special case for ECOSET error. It is now an
informational message and handled in PAT\$ECO_CMDS.


```

72 0071 1 |
73 0072 1 | TABLE OF CONTENTS:
74 0073 1 |
75 0074 1 |
76 0075 1 | FORWARD ROUTINE
77 0076 1 | PATERR HANDLER,
78 0077 1 | JNL_ERRMSG,
79 0078 1 | PATERROR_EXIT;
80 0079 1 |
81 0080 1 |
82 0081 1 | INCLUDE FILES:
83 0082 1 |
84 0083 1 |
85 0084 1 | LIBRARY 'SYSSLIBRARY:STARLET.L32';
86 0085 1 | REQUIRE 'SRC$:VXSMAC.REQ';
87 0150 1 | REQUIRE 'SRC$:PATPCT.REQ';
88 0190 1 | REQUIRE 'LIB$:PATDEF.REQ';
89 0244 1 | REQUIRE 'LIB$:PATMSG.REQ';
90 0418 1 | REQUIRE 'SRC$:PREFIX.REQ';
91 0606 1 | REQUIRE 'SRC$:PATPRE.REQ';
92 0769 1 |
93 0770 1 |
94 0771 1 | MACROS:
95 0772 1 |
96 0773 1 |
97 0774 1 |
98 0775 1 |
99 0776 1 | Define shared messages and associated severities.
100 0777 1 |
101 0778 1 |
102 P 0779 1 | $SHR_MSGDEF (PAT, 109, GLOBAL,
103 P 0780 1 | (closein, severe),
104 P 0781 1 | (closeout, severe),
105 P 0782 1 | (openin, severe),
106 P 0783 1 | (openout, severe),
107 P 0784 1 | (readerr, severe),
108 P 0785 1 | (syserror, severe),
109 P 0786 1 | (writeerr, severe),
110 0787 1 | (overlay, info));
111 0788 1 |
112 0789 1 |
113 0790 1 | EQUATED SYMBOLS:
114 0791 1 |
115 0792 1 |
116 0793 1 | LITERAL
117 0794 1 | EXCEPTION_NAME = 1;
118 0795 1 |
119 0796 1 |
120 0797 1 | OWN STORAGE:
121 0798 1 |
122 0799 1 |
123 0800 1 | OWN
124 0801 1 | JNL_PUT_MSG : VECTOR [5, LONG] INITIAL (
125 0802 1 | 4,
126 0803 1 | PAT$_JNLPUT,
127 0804 1 | 0,
128 0805 1 | 0,

! Error handler
! Action routine for SYSSYS$PUTMSG call
! Error exit routine

! System structure definitions
! Define PSECTS
! Defines literals
! PATCH error message codes

! Offset to name of exception

! SYSS$PUTMSG argument list for journal error
! Count of arguments
! Error message code
! FAO argument count
! RMS error code
```

PATERR
V04-000

M 2
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1 (2) Page 4

```

: 129      0806 1
: 130      0807 1
: 131      0808 1
: 132      0809 1  ! EXTERNAL REFERENCES:
: 133      0810 1
: 134      0811 1
: 135      0812 1  EXTERNAL ROUTINE
: 136      0813 1      SYSSPUTMSG,
: 137      0814 1      PATSEND OF LINE,
: 138      0815 1      PAT$CLOSEFILES;
: 139      0816 1
: 140      0817 1  EXTERNAL
: 141      0818 1      PAT$GB_EXEC_CMD : BYTE,
: 142      0819 1      PAT$GL_FLAGS,
: 143      0820 1      PAT$GL_INPRAB : BLOCK[,BYTE],
: 144      0821 1      PAT$GL_OUTRAB : BLOCK[,BYTE],
: 145      0822 1      PAT$GL_ERRRAB : BLOCK[,BYTE],
: 146      0823 1      PAT$GL_JNLRAB : BLOCK[,BYTE],
: 147      0824 1      PAT$GL_ERRCODE,
: 148      0825 1      PAT$GL_ERRFAB : BLOCK[,BYTE],
: 149      0826 1      PAT$GL_INPFAB : BLOCK[,BYTE],
: 150      0827 1      PAT$GL_OUTFAB : BLOCK[,BYTE];

0);

! STV error code

! Outputs error messages
! Resets for next command
! Close all files

! Command execution flag
! CLI flags
! RAB for input commands (SYSS$INPUT)
! RAB for output information (SYSS$OUTPUT)
! RAB for error messages (SYSS$ERROR)
! RAB for journal file
! Error code
! FAB for SYSS$ERROR
! FAB for SYSS$INPUT
! FAB for SYSS$OUTPUT
```



```
152 0828 1 GLOBAL ROUTINE PAT$ERR_HANDLER (SIGNAL_ARG_PTR, MECH_ARG_PTR, ENABLE_ARG_PTR) =
153 0829 1
154 0830 1 ++
155 0831 1 FUNCTIONAL DESCRIPTION:
156 0832 1
157 0833 1 This routine is SIGNALLed whenever an error occurs. It calls
158 0834 1 the routine SYS$PUTMSG to output the error message to the journal
159 0835 1 file, SYS$ERROR, and SYS$OUTPUT (if different from SYS$OUTPUT).
160 0836 1 If the journal file output fails, then PATCH produces a corresponding
161 0837 1 message to the other channels and exits with an error status.
162 0838 1
163 0839 1 If the journal write was successful, then PATCH determines the
164 0840 1 error severity level and the type of patch session. The following
165 0841 1 table shows the actions taken:
166 0842 1
167 0843 1 SEVERITY LEVEL          INTERACTIVE SESSION    BATCH SESSION
168 0844 1 FATAL                  EXITS                EXITS
169 0845 1 SEVERE                 UNWINDS              EXITS
170 0846 1 WARNING                UNWINDS              EXITS
171 0847 1 INFORMATION            RETURNS              RETURNS
172 0848 1
173 0849 1 All exits cause the files to be closed and the status code to
174 0850 1 be passed to CLI. When PATCH unwinds, a new PATCH command is
175 0851 1 requested.
176 0852 1
177 0853 1 FORMAL PARAMETERS:
178 0854 1
179 0855 1 SIGNAL_ARG_PTR - Address of the signal argument block
180 0856 1 MECH_ARG_PTR - Address of the mechanism argument block
181 0857 1 ENABLE_ARG_PTR - Address of the enable argument block, from bliss enable directive
182 0858 1 (NONE)
183 0859 1
184 0860 1 IMPLICIT INPUTS:
185 0861 1
186 0862 1 The error messages must be already defined in the system and the
187 0863 1 output files open.
188 0864 1
189 0865 1 IMPLICIT OUTPUTS:
190 0866 1
191 0867 1 none
192 0868 1
193 0869 1 ROUTINE VALUE:
194 0870 1
195 0871 1 none
196 0872 1
197 0873 1 COMPLETION CODES:
198 0874 1
199 0875 1 none
200 0876 1
201 0877 1 SIDE EFFECTS:
202 0878 1
203 0879 1 The error message is written to the SYS$ERROR, journal file,
204 0880 1 and SYS$OUTPUT channels.
205 0881 1
206 0882 1 --
207 0883 1
208 0884 2 BEGIN
```



```

209 0885 2
210 0886 2 LITERAL
211 0887 2 ARG_COUNT = 0, ! Offset to count of arguments
212 0888 2 ERR_CODE = 1; ! Offset to error code
213 0889 2
214 0890 2 MAP
215 0891 2 SIGNAL_ARG_PTR : REF VECTOR, ! Signal argument block address
216 0892 2 MECH_ARG_PTR : REF VECTOR, ! Mechanism argument block address
217 0893 2 ENABLE_ARG_PTR : REF VECTOR; ! FAO argument block address
218 0894 2
219 0895 2 !++
220 0896 2 ! Check for UNWIND exception.
221 0897 2 !--
222 0898 2 IF (.SIGNAL_ARG_PTR[EXCEPTION_NAME] EQL SS$_UNWIND)
223 0899 2 THEN
224 0900 2 RETURN SS$_RESIGNAL;
225 0901 2
226 0902 2 !++
227 0903 2 ! Cancel Control-O on SYS$ERROR and SYS$OUTPUT channels.
228 0904 2 !--
229 0905 2 PAT$GL_OUTRAB[RAB$L_ROP] = .PAT$GL_OUTRAB[RAB$L_ROP] OR RAB$M_CCO;
230 0906 2 PAT$GL_ERRRAB[RAB$L_ROP] = .PAT$GL_ERRRAB[RAB$L_ROP] OR RAB$M_CCO;
231 0907 2
232 0908 2 !++
233 0909 2 ! Decrement the count of arguments to eliminate the PC and access-violation.
234 0910 2 !--
235 0911 2 SIGNAL_ARG_PTR[ARG_COUNT] = .SIGNAL_ARG_PTR[ARG_COUNT] - 2;
236 0912 2
237 0913 2 !++
238 0914 2 ! Format the message and output it to SYS$ERROR and SYS$OUTPUT and the journal file.
239 0915 2 !--
240 0916 2 SYS$PUTMSG(.SIGNAL_ARG_PTR, JNL_ERRMSG, 0);
241 0917 2
242 0918 2 !++
243 0919 2 ! Restore the argument count.
244 0920 2 !--
245 0921 2 SIGNAL_ARG_PTR[ARG_COUNT] = .SIGNAL_ARG_PTR[ARG_COUNT] + 2;
246 0922 2
247 0923 2
248 0924 2 !++
249 0925 2 ! Check if patch session is interactive or batch. Check severity level of error.
250 0926 2 ! Either exit, unwind, and/or return based on table described above.
251 0927 2 !--
252 0928 2
253 0929 2 !++
254 0930 2 ! First handle the interactive cases.
255 0931 2 !--
256 0932 2 IF (.PAT$GL_INPFAB[FAB$L_DEV] AND DEV$M_TRM) NEQ 0 ! If this is interactive
257 0933 2 THEN
258 0934 2 BEGIN
259 0935 2 IF (.SIGNAL_ARG_PTR[ERR_CODE] AND MSG$K_FATAL) EQL MSG$K_FATAL ! If error was fatal
260 0936 2 THEN
261 0937 2 PAT$ERROR EXIT(.SIGNAL_ARG_PTR[ERR_CODE]); ! Exit with error status
262 0938 2 IF (.SIGNAL_ARG_PTR[ERR_CODE] AND MSG$K_INFO) NEQ MSG$K_INFO ! If not information msg
263 0939 2 THEN
264 0940 2 BEGIN
265 0941 2 PAT$END_OF_LINE(0); ! Reset for next command
```


PATERR
V04-000

C 3
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1 Page 7 (3)

```

: 266      0942      4      SUNWIND();      ! Unwind to command processing frame
: 267      0943      M      END;      ! End of interactive handling
: 268      0944      M      END
: 269      0945      M
: 270      0946      M      !++
: 271      0947      M      Now for a batch session.
: 272      0948      M      --
: 273      0949      M      ELSE
: 274      0950      M      IF (.SIGNAL_ARG_PTR[ERR_CODE] AND MSG$K_INFO) NEQ MSG$K_INFO ! If not information msg
: 275      0951      M      THEN
: 276      0952      M      PAT$ERROR_EXIT(.SIGNAL_ARG_PTR[ERR_CODE]);      ! Exit with error status
: 277      0953      M
: 278      0954      M      !++
: 279      0955      M      Reset the RAB so Control-0 will work correctly.
: 280      0956      M      --
: 281      0957      M      PAT$GL_OUTRAB[RAB$V_CCO]=FALSE;
: 282      0958      M      PAT$GL_ERRRAB[RAB$V_CCO]=FALSE;
: 283      0959      M      RETURN SSS_CONTINUE;
: 284      0960      M      1 END;      ! End of PAT$ERR_HANDLER
```

.TITLE PATERR
.IDENT \V04-000\

.PSECT _PAT\$OWN,NOEXE,2

00000000 00000000 00000000 006D818C 00000004 00000 JNL_PUT_MSG:
.LONG 4, 7176588, 0, 0, 0 ;

ISE\$C_SIZE== 20
TXT\$C_SIZE== 4
PAL\$C_SIZE== 16
ASD\$C_SIZE== 9
FWR\$C_SIZE== 24
PAT\$_CLOSEIN== 7147604
PAT\$_CLOSEOUT== 7147612
PAT\$_OPENIN== 7147676
PAT\$_OPENOUT== 7147684
PAT\$_READERR== 7147700
PAT\$_SYSERROR== 7147956
PAT\$_WRITEERR== 7147732
PAT\$_OVERLAY== 7147691

.EXTRN SYSS\$PUTMSG, PAT\$END_OF_LINE
.EXTRN PAT\$CLOSEFILES, PAT\$GB-EXEC_CMD
.EXTRN PAT\$GL_FLAGS, PAT\$GL_INPRAB
.EXTRN PAT\$GL_OUTRAB, PAT\$GL_ERRRAB
.EXTRN PAT\$GL_JNLRAB, PAT\$GL_ERRCODE
.EXTRN PAT\$GL_ERRFAB, PAT\$GL_INPFAB
.EXTRN PAT\$GL_OUTFAB, SYSS\$UNWIND

.PSECT _PAT\$CODE,NOWRT,2

00000920 53 00000000V EF 9E 00002
52 04 AC D0 00009
8F 04 A2 D1 0000D
06 12 00015

.ENTRY PAT\$ERR_HANDLER, Save R2,R3
MOVAB PAT\$ERROR_EXIT, R3
MOVL SIGNAL_ARG_PTR, R2
CML 4(R2), #2336
BNEQ 1\$

: 0828
: 0898
:

PATERR
V04-000

D 3
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1

Page 8
(3)

		50	0918	8F	3C	00017		MOVZWL	#2328, R0	:	0900	
					04	0001C		RET		:		
		00000000G	EF	80	8F	88	0001D	1\$:	BISB2	#128, PAT\$GL_OUTRAB+7	:	0905
		00000000G	EF	80	8F	88	00025		BISB2	#128, PAT\$GL_ERRRAB+7	:	0906
			62		02	C2	0002D		SUBL2	#2, (R2)	:	0911
					7E	D4	00030		CLRL	-(SP)	:	0916
				00000000V	EF	9F	00032		PUSHAB	JNL_ERRMSG	:	
					52	DD	00038		PUSHL	R2	:	
		00000000G	EF		03	FB	0003A		CALLS	#3, SYSS\$PUTMSG	:	
			62		02	C0	00041		ADDL2	#2, (R2)	:	0921
	26	00000000G	EF		02	E1	00044		BBC	#2, PAT\$GL_INPFAB+64, 3\$	:	0932
	06		62		22	E1	0004C		BBC	#34, (R2), -2\$	:	0935
				04	A2	DD	00050		PUSHL	4(R2)	:	0937
03	04	A2	63		01	FB	00053		CALLS	#1, PAT\$ERROR_EXIT	:	
			02		00	ED	00056	2\$:	CMPZV	#0, #2, 4(R2), #3	:	0938
					22	13	0005C		BEQL	4\$	:	
					7E	D4	0005E		CLRL	-(SP)	:	0941
		00000000G	EF		01	FB	00060		CALLS	#1, PAT\$END_OF_LINE	:	
					7E	7C	00067		CLRQ	-(SP)	:	0942
		00000000G	00		02	FB	00069		CALLS	#2, SYSS\$UNWIND	:	
					0E	11	00070		BRB	4\$	:	0932
03	04	A2	02		00	ED	00072	3\$:	CMPZV	#0, #2, 4(R2), #3	:	0950
					06	13	00078		BEQL	4\$	:	
				04	A2	DD	0007A		PUSHL	4(R2)	:	0952
			63		01	FB	0007D		CALLS	#1, PAT\$ERROR_EXIT	:	
		00000000G	EF	80	8F	8A	00080	4\$:	BICB2	#128, PAT\$GL_OUTRAB+7	:	0957
		00000000G	EF	80	8F	8A	00088		BICB2	#128, PAT\$GL_ERRRAB+7	:	0958
			50		01	D0	00090		MOVL	#1, R0	:	0959
					04	00093		RET		:	0960	

; Routine Size: 148 bytes, Routine Base: _PAT\$CODE + 0000


```

286 0961 1 ROUTINE JNL_ERRMSG (MSG_DESC)=
287 0962 1
288 0963 1 ++
289 0964 1 FUNCTIONAL DESCRIPTION:
290 0965 1
291 0966 1 This routine is called by SYSS$PUTMSG just before the message is output
292 0967 1 to SYSS$ERROR and SYSS$OUTPUT. It writes the error message or informational
293 0968 1 message to the journal file.
294 0969 1
295 0970 1 FORMAL PARAMETERS:
296 0971 1
297 0972 1 MSG_DESC - String descriptor for message
298 0973 1
299 0974 1 IMPLICIT INPUTS:
300 0975 1
301 0976 1 The journal file is open for output.
302 0977 1
303 0978 1 IMPLICIT OUTPUTS:
304 0979 1
305 0980 1 none
306 0981 1
307 0982 1 ROUTINE VALUE:
308 0983 1
309 0984 1 none
310 0985 1
311 0986 1 COMPLETION CODES:
312 0987 1
313 0988 1 none
314 0989 1
315 0990 1 SIDE EFFECTS:
316 0991 1
317 0992 1 The message is written to the journal file. If this cannot be
318 0993 1 done, then the routine causes another error message to be output
319 0994 1 and causes PATCH to exit.
320 0995 1
321 0996 1 --
322 0997 1
323 0998 2 BEGIN
324 0999 2
325 1000 2
326 1001 2 LOCAL
327 1002 2 LOCAL_ERR; ! Local error code
328 1003 2
329 1004 2 MAP
330 1005 2 MSG_DESC : REF BLOCK[,BYTE]; ! String descriptor for message
331 1006 2
332 1007 2 ++
333 1008 2 Check that the journal file has been opened.
334 1009 2 --
335 1010 2 IF (.PAT$GL_FLAGS AND PAT$M_JOURNAL) EQL 0
336 1011 2 THEN
337 1012 2 RETURN TRUE; ! Return TRUE if it has NOT.
338 1013 2
339 1014 2 ++
340 1015 2 Output the error message to the journal file.
341 1016 2 --
342 1017 2 PAT$GL_JNL$RAB[RAB$W_RSZ]=.MSG_DESC[DSC$W_LENGTH]; ! Set error message length
```



```

343 1018 2 PAT$GL_JNL_RAB[RAB$RBF]=.MSG_DESC[DSC$A_POINTER];      ! Set error message address
344 1019 3 IF NOT (LOCAL_ERR=$PUT(RAB=PAT$GL_JNL_RAB))             ! Output error to journal file
345 1020 2 THEN
346 1021 1 BEGIN
347 1022 1 ++
348 1023 1 | Unable to write to the journal file. Report the error to the
349 1024 1 | error and output channels. Then output the journal write error.
350 1025 1 --
351 1026 3 PAT$GL_OUT_RAB[RAB$RSZ]=.MSG_DESC[DSC$W_LENGTH];          ! Set error message length
352 1027 3 PAT$GL_OUT_RAB[RAB$RBF]=.MSG_DESC[DSC$A_POINTER];      ! Set error message address
353 1028 3 $PUT(RAB=PAT$GL_OUT_RAB);                                ! Output error to SYSS$OUTPUT
354 1029 4 IF (.PAT$GL_ERR_FAB[FAB$W_IFI] NEQ .PAT$GL_OUT_FAB[FAB$W_IFI])
355 1030 3 THEN
356 1031 4 BEGIN
357 1032 4 PAT$GL_ERR_RAB[RAB$RSZ]=.MSG_DESC[DSC$W_LENGTH];      ! Set error message length
358 1033 4 PAT$GL_ERR_RAB[RAB$RBF]=.MSG_DESC[DSC$A_POINTER];    ! Set error message address
359 1034 4 $PUT(RAB=PAT$GL_ERR_RAB);                                ! Output error to SYSS$ERROR
360 1035 3 END;
361 1036 3 JNL_PUT_MSG[3] = .LOCAL_ERR;
362 1037 3 JNL_PUT_MSG[4] = .PAT$GL_JNL_RAB[RAB$STV];
363 1038 3 SYSS$PUTMSG(JNL_PUT_MSG, 0, 0);
364 1039 3 PAT$ERROR_EXIT(PAT$_JNL_PUT);                            ! Report journal PUT error
365 1040 2 END;                                                    ! Exit with error status
366 1041 2 RETURN TRUE;
367 1042 1 END;
```

.EXTRN SYSS\$PUT

01FC 00000 JNL_ERRMSG:

	58	00000000	'	EF	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8	: 0961
	57	00000000	G	EF	9E	00009	MOVAB	JNL_PUT_MSG+12, R8	
	56	00000000	G	EF	9E	00010	MOVAB	PAT\$GL_ERR_RAB+34, R7	
	55	00000000	G	00	9E	00017	MOVAB	PAT\$GL_OUT_RAB+34, R6	
	54	00000000	G	EF	9E	0001E	MOVAB	SYSS\$PUT, R5	
62	00000000	G	EF	01	E1	00025	MOVAB	PAT\$GL_JNL_RAB+34, R4	
	52		04	AC	D0	0002D	BBC	#1, PAT\$GL_FLAGS, 2\$	1010
	64			62	B0	00031	MOVL	MSG_DESC, R2	1017
06	A4		04	A2	D0	00034	MOVW	(R2), PAT\$GL_JNL_RAB+34	
			DE	A4	9F	00039	MOVL	4(R2), PAT\$GL_JNL_RAB+40	1018
	65			01	FB	0003C	PUSHAB	PAT\$GL_JNL_RAB	1019
	53			50	D0	0003F	CALLS	#1, SYSS\$PUT	
	4A			53	E8	00042	MOVL	R0, LOCAL_ERR	
	66			62	B0	00045	BLBS	LOCAL_ERR, 2\$	
06	A6		04	A2	D0	00048	MOVW	(R2), PAT\$GL_OUT_RAB+34	1026
			DE	A6	9F	0004D	MOVL	4(R2), PAT\$GL_OUT_RAB+40	1027
	65			01	FB	00050	PUSHAB	PAT\$GL_OUT_RAB	1028
00000000	G	EF	00000000	G	EF	B1	CALLS	#1, SYSS\$PUT	
	67			0E	13	0005E	CMPW	PAT\$GL_ERR_FAB+2, PAT\$GL_OUT_FAB+2	1029
				62	B0	00060	BEQL	1\$	
06	A7		04	A2	D0	00063	MOVW	(R2), PAT\$GL_ERR_RAB+34	1032
			DE	A7	9F	00068	MOVL	4(R2), PAT\$GL_ERR_RAB+40	1033
	65			01	FB	0006B	PUSHAB	PAT\$GL_ERR_RAB	1034
	68			53	D0	0006E	CALLS	#1, SYSS\$PUT	
04	A8		EA	A4	D0	00071	MOVL	LOCAL_ERR, JNL_PUT_MSG+12	1036
							MOVL	PAT\$GL_JNL_RAB+T2, JNL_PUT_MSG+16	1037

PATERR
V04-000

G 3
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1
Page 11
(4)

		F4	7E	7C	00076	CLRQ	-(SP)	:	1038
00000000G	EF		A8	9F	00078	PUSHAB	JNL_PUT MSG	:	
		006D818C	03	FB	0007B	CALLS	#3, -SYSSPUTMSG	:	
00000000V	EF		8F	DD	00082	PUSHL	#7176588	:	1039
	50		01	FB	00088	CALLS	#1, PAT\$ERROR_EXIT	:	
			01	D0	0008F	MOVL	#1, R0	:	1041
			04	00092	2\$:	RET		:	1042

; Routine Size: 147 bytes, Routine Base: _PAT\$CODE + 0094

```

: 369      1043 1 ROUTINE PAT$ERROR_EXIT(ERR_CODE) =
: 370      1044 1
: 371      1045 1 ++
: 372      1046 1 FUNCTIONAL DESCRIPTION:
: 373      1047 1
: 374      1048 1     This routine exits with an error status code.
: 375      1049 1
: 376      1050 1 FORMAL PARAMETERS:
: 377      1051 1
: 378      1052 1     ERR_CODE - Error status code
: 379      1053 1
: 380      1054 1 IMPLICIT INPUTS:
: 381      1055 1
: 382      1056 1     none
: 383      1057 1
: 384      1058 1 IMPLICIT OUTPUTS:
: 385      1059 1
: 386      1060 1     none
: 387      1061 1
: 388      1062 1 ROUTINE VALUE:
: 389      1063 1
: 390      1064 1     none
: 391      1065 1
: 392      1066 1 COMPLETION CODES:
: 393      1067 1
: 394      1068 1     All kinds of error status codes and success codes.
: 395      1069 1
: 396      1070 1 SIDE EFFECTS:
: 397      1071 1
: 398      1072 1     none
: 399      1073 1
: 400      1074 1 --
: 401      1075 1
: 402      1076 2 BEGIN
: 403      1077 2
: 404      1078 2 $EXIT(CODE=(.ERR_CODE OR 1^28));
: 405      1079 1 END;
: INFO#212      L1:1078
: Null expression appears in value-required context
```

! Error exit routine

! Exit with status
! End of PAT\$ERROR_EXIT

.EXTRN SYS\$EXIT

0000 00000 PAT\$ERROR_EXIT:

7E 04 AC 10000000

8F C9 00002
01 FB 0000B
50 D4 00012
04 00 00014.WORD Save nothing
BISL3 #268435456, ERR_CODE, -(SP)
CALLS #1, SYS\$EXIT
CLRL R0
RET: 1043
: 1078
: 1079
:

; Routine Size: 21 bytes, Routine Base: _PAT\$CODE + 0127


```

: 407      1080 1 GLOBAL ROUTINE getfiledesc (fab) =
: 408      1081 2 BEGIN
: 409      1082 2 ++
: 410      1083 2
: 411      1084 2 Functional Description:
: 412      1085 2
: 413      1086 2 This routine ferrets out a string descriptor for a file name to be
: 414      1087 2 returned to the caller.
: 415      1088 2
: 416      1089 2 Inputs:
: 417      1090 2
: 418      1091 2 fab Address of the FAB for the file.
: 419      1092 2
: 420      1093 2 Outputs:
: 421      1094 2
: 422      1095 2 Routine Value is the address of the string descriptor for the file name.
: 423      1096 2
: 424      1097 2 --
: 425      1098 2
: 426      1099 2 MAP
: 427      1100 2 fab : REF $BBLOCK;
: 428      1101 2
: 429      1102 2 BIND
: 430      1103 2 nam = .fab[fab$l_nam] : $BBLOCK;
: 431      1104 2
: 432      1105 2 OWN
: 433      1106 2 filedesc : $BBLOCK [dsc$c_s_bln];
: 434      1107 2
: 435      1108 2 IF (filedesc[dsc$w_length] = .nam[nam$b_rsl]) NEQ 0
: 436      1109 2 THEN
: 437      1110 2 filedesc[dsc$a_pointer] = .nam[nam$l_rsa]
: 438      1111 2 ELSE
: 439      1112 2 IF (filedesc[dsc$w_length] = .nam[nam$b_esl]) NEQ 0
: 440      1113 2 THEN
: 441      1114 2 filedesc[dsc$a_pointer] = .nam[nam$l_esa]
: 442      1115 2 ELSE
: 443      1116 2 BEGIN
: 444      1117 2 filedesc[dsc$w_length] = .fab[fab$b_fns];
: 445      1118 2 filedesc[dsc$a_pointer] = .fab[fab$_fna];
: 446      1119 2 END;
: 447      1120 2
: 448      1121 2 RETURN filedesc
: 449      1122 1 END;
```

! End of GETFILEDESC

.PSECT _PAT\$OWN,NOEXE,2

00014 FILEDESC:

.BLKB 8

.PSECT _PAT\$CODE,NOWRT,2

52 00000000' 0004 00000
EF 9E 00002.ENTRY GETFILEDESC, Save R2
MOVAB FILEDESC, R2: 1080
:

Page 14
32:1 (6)

```
; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LIS$:PATERR/OBJ=OBJ$:PATERR MSRC$:PATERR/UPDATE=(ENHS$:PATERR)
```


PATERR
V04-000

K 3
16-Sep-1984 00:11:18
14-Sep-1984 12:52:32

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[PATCH.SRC]PATERR.B32;1 Page 15
(6)

: Size: 372 code + 28 data bytes
: Run Time: 00:18.4
: Elapsed Time: 01:07.2
: Lines/CPU Min: 3666
: Lexemes/CPU-Min: 52386
: Memory Used: 123 pages
: Compilation Complete

0301 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY