


```
PPPPPPPP      AAAAAA      SSSSSSSS      SSSSSSSS      IIIIII      GGGGGGGG      NN      NN      AAAAAA      LL
PPPPPPPP      AAAAAA      SSSSSSSS      SSSSSSSS      IIIIII      GGGGGGGG      NN      NN      AAAAAA      LL
PP      PP      AA      AA      SS      SS      II      GG      NN      NN      AA      AA      LL
PP      PP      AA      AA      SS      SS      II      GG      NN      NN      AA      AA      LL
PP      PP      AA      AA      SS      SS      II      GG      NN      NN      AA      AA      LL
PP      PP      AA      AA      SS      SS      II      GG      NN      NN      AA      AA      LL
PPPPPPPP      AA      AA      SSSSSS      SSSSSS      II      GG      NN      NN      AA      AA      LL
PPPPPPPP      AA      AA      SSSSSS      SSSSSS      II      GG      NN      NN      AA      AA      LL
PP      AAAAAAAAAA      SS      SS      II      GG      GG      NN      NN      AAAAAAAAAA      LL
PP      AAAAAAAAAA      SS      SS      II      GG      GG      NN      NN      AAAAAAAAAA      LL
PP      AA      AA      SS      SS      II      GG      GG      NN      NN      AA      AA      LL
PP      AA      AA      SS      SS      II      GG      GG      NN      NN      AA      AA      LL
PP      AA      AA      SSSSSSSS      SSSSSSSS      IIIIII      GGGGGG      NN      NN      AA      AA      LL
PP      AA      AA      SSSSSSSS      SSSSSSSS      IIIIII      GGGGGG      NN      NN      AA      AA      LL
LL
```

```
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
LLLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
```



```
1 0001 0 MODULE PASS$$SIGNAL ( %TITLE 'Signal I/O errors'
2 0002 0 IDENT = '1-001' ! File: PASSIGNAL.B32 Edit: SBL1001
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: Pascal Language Support
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains procedures which are called by the
36 0036 1 VAX-11 Pascal Run-Time Library to signal error conditions.
37 0037 1
38 0038 1 ENVIRONMENT: User mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. SBL 1-April-1981
45 0045 1 --
46 0046 1
```



```
: 48      0047 1 %SBTTL 'Declarations'
: 49      0048 1
: 50      0049 1 : PROLOGUE DEFINITIONS:
: 51      0050 1 :
: 52      0051 1
: 53      0052 1 REQUIRE 'RTLIN:PASPROLOG';           ! Linkages, externals, PSECTs, structures
: 54      0116 1
: 55      0117 1 :
: 56      0118 1 : TABLE OF CONTENTS:
: 57      0119 1 :
: 58      0120 1
: 59      0121 1 FORWARD ROUTINE
: 60      0122 1     PAS$$SIGNAL: CALL SIGNAL NOVALUE,      ! Signals I/O error
: 61      0123 1     PAS$$BUGCHECK: NOVALUE;                ! Signals PAS$_BUGCHECK
: 62      0124 1
: 63      0125 1 :
: 64      0126 1 : MACROS:
: 65      0127 1 :
: 66      0128 1
: 67      0129 1 :+
: 68      0130 1 : Macros to read and write longwords that use autoincrementing.
: 69      0131 1 :-
: 70      0132 1
: 71      0133 1 MACRO
: 72      0134 1     RLONG_A (src) = (src=.src+4; .(.src-4)<0,32>) %;
: 73      0135 1     WLONG_A (src,dst) = (dst=.dst+4; (.dst-4)<0,32>=src) %;
: 74      0136 1
: 75      0137 1 :
: 76      0138 1 : EQUATED SYMBOLS:
: 77      0139 1 :
: 78      0140 1 :     NONE
: 79      0141 1 :
: 80      0142 1 : FIELDS:
: 81      0143 1 :
: 82      0144 1 :     NONE
: 83      0145 1 :
: 84      0146 1 : OWN STORAGE:
: 85      0147 1 :
: 86      0148 1 :     NONE
```



```
88 0149 1 %SBTTL 'PASS$SIGNAL - Signal PASS$_ERRACCFIL'
89 0150 1 GLOBAL ROUTINE PASS$SIGNAL ( ! Signal I/O error
90 0151 1 PFV: VOLATILE REF $PASS$PFV_FILE_VARIABLE, ! File variable
91 0152 1 MESSAGES: VECTOR [, LONG] ! Messages to signal
92 0153 1 ) : CALL_SIGNAL NOVALUE =
93 0154 1
94 0155 1 ++
95 0156 1 FUNCTIONAL DESCRIPTION:
96 0157 1
97 0158 1 This procedure is called by the Run-Time Library when it wishes
98 0159 1 to signal an error during a Pascal I/O operation. It adds
99 0160 1 to the message the file "variable" name, filename, and
100 0161 1 RMS statuses, if any.
101 0162 1
102 0163 1 CALLING SEQUENCE:
103 0164 1
104 0165 1 CALL PASS$SIGNAL (PFV.mr.r,
105 0166 1 MESSAGE1.rw.v [,ARGCNT1.rl.v [,FAOARGS.rz.z...]]
106 0167 1 [, MESSAGE2.rlc.v, FAOCNT2.rl.v [, FAOARGS.rz.z...]])
107 0168 1
108 0169 1 FORMAL PARAMETERS:
109 0170 1
110 0171 1 PFV - The Pascal File Variable (PFV) passed in R6.
111 0172 1 The structure of the PFV is defined in PASPFV.REQ.
112 0173 1
113 0174 1 MESSAGE1 - The first message which gives more detail about the
114 0175 1 error. This may be one of the PASS$K_xxx symbols
115 0176 1 which correspond to a PASS$xxx I/O error message, in
116 0177 1 which case it is expanded into the full 32-bit form.
117 0178 1
118 0179 1 FAOCNT1 - The count of FAO arguments for MESSAGE1, if not zero.
119 0180 1 If zero, the RMS STS and STV are taken from the RAB
120 0181 1 or FAB.
121 0182 1
122 0183 1 FAOARG1 - The first FAO argument for MESSAGE1, if any, etc.
123 0184 1
124 0185 1
125 0186 1 IMPLICIT INPUTS:
126 0187 1
127 0188 1 NONE
128 0189 1
129 0190 1 IMPLICIT OUTPUTS:
130 0191 1
131 0192 1 FCB$STATUS is set to the STATUS number of the error, unless
132 0193 1 there is no FCB, in which case the STATUS number is stored
133 0194 1 in PFV [PFV$STATUS];
134 0195 1
135 0196 1 ROUTINE VALUE:
136 0197 1
137 0198 1 NONE
138 0199 1
139 0200 1 SIDE EFFECTS:
140 0201 1
141 0202 1 NONE
142 0203 1
143 0204 1 SIGNALLED ERRORS:
144 0205 1
```



```
145 0206 1 | Message indicated by caller
146 0207 1 |
147 0208 1 |--
148 0209 1 |
149 0210 2 BEGIN
150 0211 2
151 0212 2 LOCAL
152 0213 2 PFD: REF $PASS$PFD FILE_DESCRIPTOR, ! Pascal File Descriptor
153 0214 2 MSG_PTR: REF VECTOR [, LONG], ! Pointer to MESSAGE_LIST
154 0215 2 ARG_PTR: REF VECTOR [, LONG], ! Pointer to argument list
155 0216 2 FILENAME_DSC: BLOCK [8, BYTE], ! Filename descriptor
156 0217 2 MESSAGE_LIST: VECTOR [20]; ! Argument list to LIB$SIGNAL
157 0218 2
158 0219 2 BIND
159 0220 2 PRIMARY_MESSAGE = MESSAGE_LIST [1]: BLOCK [4, BYTE];
160 0221 2
161 0222 2 BUILTIN
162 0223 2 ACTUALCOUNT,
163 0224 2 CALLG;
164 0225 2
165 0226 2 EXTERNAL ROUTINE
166 0227 2 LIB$SIGNAL; ! Can't use SIGNAL here
167 0228 2
168 0229 2 !+
169 0230 2 ! Expand the first message from the PASS$K value to the PASS$ value if
170 0231 2 ! necessary. Store the primary message in the message list.
171 0232 2 !-
172 0233 2
173 0234 2 IF .MESSAGES [0] LEQU 65535 ! Less than a word?
174 0235 2 THEN
175 0236 2 BEGIN
176 0237 2 PRIMARY_MESSAGE = PASS$ BASE; ! Get base message
177 0238 2 PRIMARY_MESSAGE = .PRIMARY_MESSAGE + (.MESSAGES [0]^3); ! Insert message number
178 0239 2 END
179 0240 2 ELSE
180 0241 2 PRIMARY_MESSAGE = .MESSAGES [0]; ! Get full message
181 0242 2
182 0243 2 !+
183 0244 2 ! Determine the number of signal arguments. If we have more than one
184 0245 2 ! argument, the second argument is the count of additional FAO arguments
185 0246 2 ! for the first message. Add two for possible RMS messages, plus the three
186 0247 2 ! implicit arguments for the first message.
187 0248 2 !-
188 0249 2
189 0250 2 IF ACTUALCOUNT () GTRU 1
190 0251 2 THEN
191 0252 2 BEGIN
192 0253 2 MESSAGE_LIST [0] = ACTUALCOUNT () + 3; ! Total signal args
193 0254 2 MESSAGE_LIST [2] = 3 + .MESSAGES [1]; ! FAO count for primary message
194 0255 2 MESSAGE_LIST [3] = PFV [PFV$R-PFV]; ! PFV address goes here
195 0256 2 ARG_PTR = MESSAGE_LIST [2]; ! Start of secondary arguments
196 0257 2 MSG_PTR = MESSAGE_LIST [4]; ! Next place in MESSAGE_LIST
197 0258 2
198 0259 2 !+
199 0260 2 ! Move extra FAO arguments for primary message, if any.
200 0261 2 !-
201 0262 2
```



```
202 0263 3      DECR I FROM (.MESSAGE_LIST [2] - 3) TO 1 DO
203 0264 3          WLONG_A (RLONG_A (ARG_PTR), MSG_PTR);      ! Copy an item
204 0265 3      END
205 0266 2      ELSE
206 0267 3          BEGIN
207 0268 3              MESSAGE_LIST [0] = 5;      ! Total signal args
208 0269 3              MESSAGE_LIST [2] = 3;      ! FAO count for primary message
209 0270 3              MESSAGE_LIST [3] = PFV [PFV$R_PFV];    ! PFV address goes here
210 0271 3              MSG_PTR = MESSAGE_LIST [4];    ! Next place in MESSAGE_LIST
211 0272 2          END;
212 0273 2
213 0274 2      !+
214 0275 2      ! Get PFD address. Take care that we have the "absolute" address and
215 0276 2      ! not the relative displacement!
216 0277 2      !-
217 0278 2
218 0279 2      IF .PFV [PFV$V_RELPFD]      ! Is it relative?
219 0280 2      THEN
220 0281 3          BEGIN
221 0282 3              !+
222 0283 3              ! The PFD pointer in the PFV was self-relative when we
223 0284 3              ! started - but it might not be now. Fetch it, then test
224 0285 3              ! again. If RELPFD is still clear, then what we have really
225 0286 3              ! is an offset. If it is now set, get the now-absolute address
226 0287 3              ! from the PFV.
227 0288 3              !-
228 0289 3
229 0290 3              PFD = .PFV [PFV$A_PFD] + .PFV;    ! Resolve address
230 0291 3              IF .PFV [PFV$V_RE[PFD]]    ! Was it set in the meantime?
231 0292 3              THEN
232 0293 3                  PFD = .PFV [PFV$A_PFD];
233 0294 3              END
234 0295 2      ELSE
235 0296 2          PFD = .PFV [PFV$A_PFD];    ! Get absolute address
236 0297 2
237 0298 2      !+
238 0299 2      ! File variable name is in PFD. If name has zero length, substitute a
239 0300 2      ! dummy name.
240 0301 2      !-
241 0302 2
242 0303 2      IF .PFD [PFD$T_NAME] GTRU 0      ! Name specified?
243 0304 2      THEN
244 0305 3          WLONG_A (PFD [PFD$T_NAME], MSG_PTR)    ! Counted string
245 0306 2      ELSE
246 0307 2          WLONG_A (UPLIT BYTE (%CHARCOUNT('-unnamed-'), '-unnamed-'), MSG_PTR);
247 0308 2
248 0309 2      !+
249 0310 2      ! See if we have a filename. If so, add it to the message.
250 0311 2      ! At the same time, store the error number in the FCB.
251 0312 2      !-
252 0313 2
253 0314 2      FILENAME_DSC [DSC$W_LENGTH] = 0;    ! Initially, no filename
254 0315 2      WLONG_A (FILENAME_DSC, MSG_PTR);    ! Address of descriptor
255 0316 2
256 0317 2      IF .PFV [PFV$V_FCB_VALID]
257 0318 2      THEN
258 0319 3          BEGIN
```



```
259 0320 3 LOCAL
260 0321 3 FCB: REF $PASS$FCB_CONTROL_BLOCK, ! File control block
261 0322 3 FAB: REF BLOCK [, BYTE], ! RMS FAB
262 0323 3 NAM: REF BLOCK [, BYTE]; ! RMS NAM
263 0324 3
264 0325 3 BIND
265 0326 3 RAB = FCB: REF BLOCK [, BYTE]; ! RMS RAB
266 0327 3
267 0328 3 FCB = .PFV [PFV$_FCB]; ! Get FCB address
268 0329 3
269 0330 3 IF NOT .FCB [FCB$_STRING]
270 0331 3 THEN
271 0332 4 BEGIN
272 0333 4 FAB = .RAB [RAB$_FAB]; ! Get FAB address
273 0334 4 NAM = .FAB [FAB$_NAM]; ! Get NAM address
274 0335 4 IF .NAM [NAM$_RS] GTRU 0 ! Do we have an RSN?
275 0336 4 THEN
276 0337 5 BEGIN
277 0338 5 FILENAME_DSC [DSC$_LENGTH] = .NAM [NAM$_RSL];
278 0339 5 FILENAME_DSC [DSC$_POINTER] = .NAM [NAM$_RSA];
279 0340 5 END
280 0341 4 ELSE IF .NAM [NAM$_ESL] GTRU 0 ! Do we have an ESN?
281 0342 4 THEN
282 0343 5 BEGIN
283 0344 5 FILENAME_DSC [DSC$_LENGTH] = .NAM [NAM$_ESL];
284 0345 5 FILENAME_DSC [DSC$_POINTER] = .NAM [NAM$_ESA];
285 0346 5 END
286 0347 4 ELSE IF .FAB [FAB$_FNS] GTRU 0 ! Do we have a filename?
287 0348 4 THEN
288 0349 5 BEGIN
289 0350 5 FILENAME_DSC [DSC$_LENGTH] = .FAB [FAB$_FNS];
290 0351 5 FILENAME_DSC [DSC$_POINTER] = .FAB [FAB$_FNA];
291 0352 4 END;
292 0353 3 END;
293 0354 3
294 0355 3 !+
295 0356 3 If we have neither an RSN or a filename, we'll use a null string
296 0357 3 by default.
297 0358 3 !-
298 0359 3
299 0360 3 !+
300 0361 3 Store error number in FCB so that STATUS can return it.
301 0362 3 !-
302 0363 3 FCB [FCB$_STATUS] = .PRIMARY_MESSAGE [STSS$_CODE] - PASS$_K_MSGV2LO;
303 0364 3 END
304 0365 3 ELSE
305 0366 2 !+
306 0367 2 Store error number in PFV.
307 0368 2 !-
308 0369 2 PFV [PFV$_STATUS] = .PRIMARY_MESSAGE [STSS$_CODE] - PASS$_K_MSGV2LO;
309 0370 2
310 0371 2 !+
311 0372 2 If any chained messages are to be signalled, copy them.
312 0373 2 !-
313 0374 2
314 0375 2 IF ACTUALCOUNT () GTRU 1 ! Any chained messages at all?
315 0376 2
```



```

: 316      0377      2      THEN
: 317      0378      3      BEGIN
: 318      0379      3      IF (.MESSAGE_LIST [0] - (.MESSAGE_LIST [2] + 2)) GTR 0
: 319      0380      3      THEN
: 320      0381      3      DECR I FROM (.MESSAGE_LIST [0] - (.MESSAGE_LIST [2] + 2)) TO 1 DO
: 321      0382      4      WLONG_A (RLONG_A (ARG_PTR), MSG_PTR)
: 322      0383      3      END
: 323      0384      2      ELSE
: 324      0385      2      !+
: 325      0386      2      ! If there's an error in the FAB or RAB, copy that error to the message.
: 326      0387      2      !-
: 327      0388      2      IF .PFV [PFV$V_FCB_VALID]
: 328      0389      2      THEN
: 329      0390      3      BEGIN
: 330      0391      3      LOCAL
: 331      0392      3      FCB: REF $PASSFCB_CONTROL_BLOCK,
: 332      0393      3      FAB: REF BLOCK [, BYTE];
: 333      0394      3      BIND
: 334      0395      3      RAB = FCB: REF BLOCK [, BYTE];
: 335      0396      3
: 336      0397      3      FCB = .PFV [PFV$A_FCB];
: 337      0398      3      IF NOT .FCB [FCB$V_STRING]
: 338      0399      3      THEN
: 339      0400      4      BEGIN
: 340      0401      4      FAB = .RAB [RAB$L_FAB];
: 341      0402      5      IF NOT .RAB [RAB$L_STS] AND (.RAB [RAB$L_STS] NEQ 0)
: 342      0403      4      THEN
: 343      0404      5      BEGIN
: 344      0405      5      WLONG_A (.RAB [RAB$L_STS], MSG_PTR);
: 345      0406      5      WLONG_A (.RAB [RAB$L_STV], MSG_PTR);
: 346      0407      5      MESSAGE_LIST [0] = .MESSAGE_LIST [0] + 2;
: 347      0408      5      END
: 348      0409      5      ELSE IF NOT .FAB [FAB$L_STS] AND (.FAB [FAB$L_STS] NEQ 0)
: 349      0410      4      THEN
: 350      0411      5      BEGIN
: 351      0412      5      WLONG_A (.FAB [FAB$L_STS], MSG_PTR);
: 352      0413      5      WLONG_A (.FAB [FAB$L_STV], MSG_PTR);
: 353      0414      5      MESSAGE_LIST [0] = .MESSAGE_LIST [0] + 2;
: 354      0415      4      END;
: 355      0416      3      END;
: 356      0417      2      END;
: 357      0418      2
: 358      0419      2      CALLG (MESSAGE_LIST, LIB$$SIGNAL);          ! Do the signal
: 359      0420      2
: 360      0421      2      !+
: 361      0422      2      ! We can only return here if a user handler has returned SS$ CONTINUE.
: 362      0423      2      ! Our callers assume that we never return, so signal_stop the special
: 363      0424      2      ! condition code PASS$ CONTINUE. This tells the appropriate incarnation
: 364      0425      2      ! of PASS$$IO_HANDLER to unwind to the caller of its establisher,
: 365      0426      2      ! effectively dismissing the error.
: 366      0427      2      !-
: 367      0428      2
: 368      0429      2      SIGNAL_STOP (PASS$ CONTINUE, 1, PFV [PFV$R_PFV]);
: 369      0430      2
: 370      0431      2      RETURN;          ! If handler continues
: 371      0432      2
: 372      0433      1      END;          ! End of routine PAS$$SIGNAL
```


										.TITLE PASS\$SIGNAL Signal I/O errors		
										.IDENT \1-001\		
										.PSECT _PASS\$CODE,NOWRT, SHR, PIC,2		
										.BYTE 9		:
										.ASCII \-unnamed-\		:
										.EXTRN PASS\$SIGNAL, PASS\$BUGCHECK		
										.EXTRN LIB\$SIGNAL, PASS\$ BASE		
										.EXTRN PASS\$K_MSGV2LO, PASS\$CONTINUE		
										.ENTRY PASS\$SIGNAL, Save R2,R3,R4,R5		: 0150
										MOVAB -88(SP), SP		:
										MOVL MESSAGES, R0		: 0234
										CMPL R0, #65535		:
										BGTRU 1\$		:
										MOVL #PASS\$ BASE, PRIMARY_MESSAGE		: 0237
										MOVAQ @PRIMARY_MESSAGE[R0], PRIMARY_MESSAGE		: 0238
										BRB 2\$		: 0234
										MOVL R0, PRIMARY_MESSAGE		: 0241
										MOVAB MESSAGE_LIST+16, MSG_PTR		: 0257
										CMPB (AP), #T		: 0250
										BLEQU 5\$		:
										MOVZBL (AP), MESSAGE_LIST		: 0253
										ADDL2 #3, MESSAGE_LIST		:
										ADDL3 #3, MESSAGES+4, MESSAGE_LIST+8		: 0254
										MOVL PFV, MESSAGE_LIST+12		: 0255
										MOVAB MESSAGES+8, ARG_PTR		: 0256
										SUBL3 #2, MESSAGE_LIST+8, I		: 0263
										BRB 4\$		:
										MOVL (ARG_PTR)+, (MSG_PTR)+		: 0264
										SOBGTR I, 3\$		:
										BRB 6\$		: 0250
										MOVL #5, MESSAGE_LIST		: 0268
										MOVL #3, MESSAGE_LIST+8		: 0269
										MOVL PFV, MESSAGE_LIST+12		: 0270
										BBC #4, 7(PFV), 7\$		: 0279
										ADDL3 8(PFV), PFV, PFD		: 0290
										BBC #4, 7(PFV), 8\$		: 0291
										MOVL 8(PFV), PFD		: 0296
										TSTB 12(PFD)		: 0303
										BEQL 9\$		:
										MOVAB 12(PFD), (MSG_PTR)+		: 0305
										BRB 10\$		:
										MOVAB P.AAA, (MSG_PTR)+		: 0307
										CLRW FILENAME_DSC		: 0314
										MOVAB FILENAME_DSC, (MSG_PTR)+		: 0315
										EXTZV #3, #12, PRIMARY_MESSAGE, R5		: 0364
										SUBL2 #PASS\$K_MSGV2LO, R5		:
										BBC #6, 7(PFV), 14\$		: 0317
										MOVL 12(PFV), FCB		: 0328
										BBS #4, -2(FCB), 13\$		: 0330
										MOVL 60(FCB), FAB		: 0333
										MOVL 40(FAB), NAM		: 0334

PASS\$SIGNAL
1-001

Signal I/O errors
PASS\$SIGNAL - Signal PASS_ERRACCFIL

K 2
16-Sep-1984 02:08:14
14-Sep-1984 12:51:56

VAX-11 Bliss-32 V4.0-7.2
[PASRTL.SRC]PASSIGNAL.832;1

Page 9
(3)

			03	A0	95	000AB	TSTB	3(NAM)	: 0335
				0C	13	000AE	BEQL	11\$	:
50	AE		03	A0	9B	000B0	MOVZBW	3(NAM), FILENAME_DSC	: 0338
54	AE		04	A0	D0	000B5	MOVL	4(NAM), FILENAME_DSC+4	: 0339
				20	11	000BA	BRB	13\$	: 0335
			0B	A0	95	000BC	11\$: TSTB	11(NAM)	: 0341
				0C	13	000BF	BEQL	12\$	:
50	AE		0B	A0	9B	000C1	MOVZBW	11(NAM), FILENAME_DSC	: 0344
54	AE		0C	A0	D0	000C6	MOVL	12(NAM), FILENAME_DSC+4	: 0345
				0F	11	000CB	BRB	13\$	: 0341
			34	A2	95	000CD	12\$: TSTB	52(FAB)	: 0347
				0A	13	000D0	BEQL	13\$	:
50	AE		34	A2	9B	000D2	MOVZBW	52(FAB), FILENAME_DSC	: 0350
54	AE		2C	A2	D0	000D7	MOVL	44(FAB), FILENAME_DSC+4	: 0351
D4	A3			55	D0	000DC	13\$: MOVL	R5, -44(FCB)	: 0364
				04	11	000E0	BRB	15\$	: 0317
				55	D0	000E2	14\$: MOVL	R5, 12(PFV)	: 0370
				6C	91	000E6	15\$: CMPB	(AP), #1	: 0376
				1B	1B	000E9	BLEQU	18\$	:
50	08	AE		02	C1	000EB	ADDL3	#2, MESSAGE_LIST+8, R0	: 0379
	50			6E	D1	000F0	CMPL	MESSAGE_LIST, R0	:
				42	15	000F3	BLEQ	21\$	:
50		6E	08	AE	C3	000F5	SUBL3	MESSAGE_LIST+8, MESSAGE_LIST, R0	: 0381
				50	D7	000FA	DECL	I	:
				03	11	000FC	BRB	17\$	:
		81		84	D0	000FE	16\$: MOVL	(ARG_PTR)+, (MSG_PTR)+	: 0382
		FA		50	F5	00101	17\$: SOBGTR	I, 16\$	:
				31	11	00104	BRB	21\$	: 0378
2C	07	A6		06	E1	00106	18\$: BBC	#6, 7(PFV), 21\$	: 0388
		50	0C	A6	D0	0010B	MOVL	12(PFV), FCB	: 0397
23	FE	A0		04	E0	0010F	BBS	#4, -2(FCB), 21\$	: 0398
		52	3C	A0	D0	00114	MOVL	60(FCB), FAB	: 0401
		0B	08	A0	E8	00118	BLBS	8(FCB), 19\$	: 0402
			08	A0	D5	0011C	TSTL	8(FCB)	:
				06	13	0011F	BEQL	19\$	:
		81	08	A0	7D	00121	MOVQ	8(FCB), (MSG_PTR)+	: 0405
				0D	11	00125	BRB	20\$	: 0407
		0C	08	A2	E8	00127	19\$: BLBS	8(FAB), 21\$	: 0409
			08	A2	D5	0012B	TSTL	8(FAB)	:
				07	13	0012E	BEQL	21\$	:
		81	08	A2	7D	00130	MOVQ	8(FAB), (MSG_PTR)+	: 0412
		6E		02	C0	00134	20\$: ADDL2	#2, MESSAGE_LIST	: 0414
00000000G	00			6E	FA	00137	21\$: CALLG	MESSAGE_LIST, LIB\$SIGNAL	: 0419
				56	DD	0013E	PUSHL	PFV	: 0429
				01	DD	00140	PUSHL	#1	:
				8F	DD	00142	PUSHL	#PASS CONTINUE	:
00000000G	00	00000000G		03	FB	00148	CALLS	#3, LIB\$STOP	:
				04	0014F		RET		: 0433

; Routine Size: 336 bytes, Routine Base: _PASSCODE + 000A

: 373 0434 1
: 374 0435 1 !<BLF/PAGE>


```
376 0436 1 %SBTTL 'PASS$BUGCHECK - Signal PASS$_BUGCHECK'
377 0437 1 GLOBAL ROUTINE PASS$BUGCHECK (
378 0438 1     CODE
379 0439 1     ) : NOVALUE =
380 0440 1
381 0441 1 ++
382 0442 1 FUNCTIONAL DESCRIPTION:
383 0443 1
384 0444 1     This procedure is called by the Run-Time Library when it wishes
385 0445 1     to indicate an internal consistency check failure. It signals
386 0446 1     PASS$_BUGCHECK, internal consistency error <nnn> in Pascal Run-Time Library.
387 0447 1
388 0448 1 CALLING SEQUENCE:
389 0449 1
390 0450 1     CALL PASS$BUGCHECK (CODE.rlu.v)
391 0451 1
392 0452 1 FORMAL PARAMETERS:
393 0453 1
394 0454 1     CODE          A longword integer denoting the particular error
395 0455 1                  being reported. This is to help locate the problem
396 0456 1                  from SPR reports. The code is generated by the
397 0457 1                  $PASS$BUGCHECK macro.
398 0458 1
399 0459 1 IMPLICIT INPUTS:
400 0460 1
401 0461 1     NONE
402 0462 1
403 0463 1 IMPLICIT OUTPUTS:
404 0464 1
405 0465 1     NONE
406 0466 1
407 0467 1 ROUTINE VALUE:
408 0468 1
409 0469 1     NONE
410 0470 1
411 0471 1 SIDE EFFECTS:
412 0472 1
413 0473 1     NONE
414 0474 1
415 0475 1 SIGNALLED ERRORS:
416 0476 1
417 0477 1     PASS$_BUGCHECK
418 0478 1
419 0479 1 --
420 0480 1
421 0481 2 BEGIN
422 0482 2
423 0483 2     SIGNAL_STOP (PASS$_BUGCHECK, 1, .CODE);      ! Signal fatal error
424 0484 2
425 0485 2     RETURN;                                          ! Should never return here
426 0486 2
427 0487 1 END;                                          ! End of routine PASS$BUGCHECK
```

.EXTRN PASS\$_BUGCHECK

PASS\$SIGNAL
1-001

Signal I/O errors
PASS\$BUGCHECK - Signal PASS\$_BUGCHECK

M 2
16-Sep-1984 02:08:14
14-Sep-1984 12:51:56

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASSIGNAL.B32;1

Page 11
(4)

0000 0000
04 AC DD 00002
01 DD 00005
00000000G 00 00000000G 8F DD 00007
03 FB 0000D
04 00014

.ENTRY PASS\$BUGCHECK, Save nothing
PUSHL CODE
PUSHL #1
PUSHL #PASS\$_BUGCHECK
CALLS #3, LIB\$STOP
RET

: 0437
: 0483
:
:
:
:
: 0487

: Routine Size: 21 bytes, Routine Base: _PASS\$CODE + 015A

: 428 0488 1
: 429 0489 1 !<BLF/PAGE>

PASS\$SIGNAL
1-001

Signal I/O errors
PASS\$BUGCHECK - Signal PASS\$_BUGCHECK

N 2
16-Sep-1984 02:08:14 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:51:56 [PASRTL.SRC]PASSIGNAL.B32;1

Page 12
(5)

: 431 0490 1 END
: 432 0491 1
: 433 0492 0 ELUDOM

! End of module PASS\$SIGNAL

.EXTRN LIB\$STOP

PSECT SUMMARY

: Name Bytes Attributes
: _PASS\$CODE 367 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	15	0	581	00:01.0
_\$255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1	427	107	25	33	00:00.4

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:PASSIGNAL/OBJ=OBJ\$:PASSIGNAL MSRC\$:PASSIGNAL/UPDATE=(ENH\$:PASSIGNAL
:)

: Size: 357 code + 10 data bytes
: Run Time: 00:11.8
: Elapsed Time: 00:36.9
: Lines/CPU Min: 2499
: Lexemes/CPU-Min: 19961
: Memory Used: 170 pages
: Compilation Complete

0297

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY