

```

PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA  SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA  SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA  SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA      SSS      RRR      RRR      TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA      SSS      RRR      RRR      TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA      SSS      RRR      RRR      TTT              LLL
PPP              AAA          AAA      SSS      RRR              RRR      TTT              LLL
PPP              AAA          AAA      SSS      RRR              RRR      TTT              LLL
PPP              AAA          AAA      SSS      RRR              RRR      TTT              LLL
PPP              AAA          AAA      SSS      RRR              RRR      TTT              LLL
PPP              AAA          AAA      SSSSSSSSSSSS      RRR              RRR      TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA      SSSSSSSSSSSS      RRR              RRR      TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA      SSSSSSSSSSSS      RRR              RRR      TTT              LLLLLLLLLLLLLLLLLL

```

2

Sym

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS
PAS

PAS

100

PAS

PAS

143

PAS

PAS

PAS

PAS

PAS
PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS

PAS
PAS

PAS

```
PPPPPPPP      AAAAAA      SSSSSSSS      WW      WW      RRRRRRRR      IIIIII      EEEEEEEEE      NN      NN      UU      UU
PPPPPPPP      AAAAAA      SSSSSSSS      WW      WW      RRRRRRRR      IIIIII      EEEEEEEEE      NN      NN      UU      UU
PP      PP      AA      AA      SS      WW      WW      RR      RR      II      EE      NN      NN      UU      UU
PP      PP      AA      AA      SS      WW      WW      RR      RR      II      EE      NN      NN      UU      UU
PP      PP      AA      AA      SS      WW      WW      RR      RR      II      EE      NN      NN      UU      UU
PPPPPPPP      AA      AA      SSSSSS      WW      WW      RRRRRRRR      II      EE      NN      NN      UU      UU
PPPPPPPP      AA      AA      SSSSSS      WW      WW      RRRRRRRR      II      EE      NN      NN      UU      UU
PP      AAAAAAAAAA      SS      WW      WW      RR      RR      II      EE      NN      NN      UU      UU
PP      AAAAAAAAAA      SS      WW      WW      RR      RR      II      EE      NN      NN      UU      UU
PP      AA      AA      SS      WWW      WWW      RR      RR      II      EE      NN      NN      UU      UU
PP      AA      AA      SS      WWW      WWW      RR      RR      II      EE      NN      NN      UU      UU
PP      AA      AA      SSSSSSSS      WW      WW      RR      RR      IIIIII      EEEEEEEEE      NN      NN      UUUUUUUUU
PP      AA      AA      SSSSSSSS      WW      WW      RR      RR      IIIIII      EEEEEEEEE      NN      NN      UUUUUUUUU

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 PASSWRITE_ENUMERATED ( %TITLE 'Write an enumerated value'
2 0002 0 IDENT = '1-002' ! File: PASWRIENU.B32 Edit: SBL1002
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 a procedure which writes an enumerated value
36 0036 1 to a textfile.
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 1-002 - Make total-width a longword. SBL 29-Jun-1982
46 0046 1 --
47 0047 1
```

PASSWRITE_ENUM Write an enumerated value
1-002 Declarations

N 13
16-Sep-1984 02:16:50
14-Sep-1984 12:52:04

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

Page 2
(2)

```

: 49      0048 1 %SBTTL 'Declarations'
: 50      0049 1
: 51      0050 1 : PROLOGUE DEFINITIONS:
: 52      0051 1 :
: 53      0052 1
: 54      0053 1 REQUIRE 'RTLIN:PASPROLOG';           ! Externals, linkages, PSECTs, structures
: 55      0117 1
: 56      0118 1 :
: 57      0119 1 : TABLE OF CONTENTS:
: 58      0120 1 :
: 59      0121 1
: 60      0122 1 FORWARD ROUTINE
: 61      0123 1     PASSWRITE_ENUMERATED: NOVALUE,
: 62      0124 1     PASSWRITEV_ENUMERATED: NOVALUE;    ! Write to textfile
: 63      0125 1                                           ! Write to string
: 64      0126 1 :
: 65      0127 1 : MACROS:
: 66      0128 1 :
: 67      0129 1 :     NONE
: 68      0130 1 :
: 69      0131 1 : EQUATED SYMBOLS:
: 70      0132 1 :
: 71      0133 1 :     NONE
: 72      0134 1 :
: 73      0135 1 : FIELDS:
: 74      0136 1 :
: 75      0137 1 :     NONE
: 76      0138 1 :
: 77      0139 1 : OWN STORAGE:
: 78      0140 1 :
: 79      0141 1 :     NONE
: 80      0142 1 :
```



```

: 82      0143 1 %SBTTL 'PASSWRITE_ENUMERATED - Write an enumerated value to textfile'
: 83      0144 1 GLOBAL ROUTINE PASSWRITE_ENUMERATED (
: 84      0145 1     PFV: REF $PASSPFV FILE VARIABLE,           | File variable
: 85      0146 1     PETD: REF VECTOR [, LONG],                 | Enumerated type descriptor
: 86      0147 1     VALUE,                                       | Value to write
: 87      0148 1     TOTAL_WIDTH: SIGNED,                         | Total field width
: 88      0149 1     ERROR                                         | Error unwind address
: 89      0150 1 ): NOVALUE =
: 90      0151 1
: 91      0152 1 ++
: 92      0153 1 FUNCTIONAL DESCRIPTION:
: 93      0154 1     This procedure writes an enumerated value to the specified textfile.
: 94      0155 1
: 95      0156 1 CALLING SEQUENCE:
: 96      0157 1
: 97      0158 1     CALL PASSWRITE_ENUMERATED (PFV.mr.r, PETD.rr.r, VALUE.rlu.v,
: 98      0159 1     TOTAL_WIDTH.rl.v [ERROR.j.r])
: 99      0160 1
: 100     0161 1 FORMAL PARAMETERS:
: 101     0162 1
: 102     0163 1     PFV      - The Pascal File Variable (PFV) passed by reference.
: 103     0164 1     The structure of the PFV is defined in PASPFV.REQ.
: 104     0165 1
: 105     0166 1     PETD    - Pascal Enumerated Type Descriptor, passed by reference.
: 106     0167 1     The structure of a PETD is as follows:
: 107     0168 1
: 108     0169 1
: 109     0170 1
: 110     0171 1     +-----+
: 111     0172 1     | offset of ASCII type name | <-- PETD
: 112     0173 1     +-----+
: 113     0174 1     | count of possible values (n) |
: 114     0175 1     +-----+
: 115     0176 1     | offset of ASCII value name 0 |
: 116     0177 1     +-----+
: 117     0178 1     | offset of ASCII value name 1 |
: 118     0179 1     +-----+
: 119     0180 1     |
: 120     0181 1     | offset of ASCII value name n-1 |
: 121     0182 1     +-----+
: 122     0183 1
: 123     0184 1     The offsets are relative to the address
: 124     0185 1     of the descriptor (PETD). The names are
: 125     0186 1     counted strings with 1-byte counts. It
: 126     0187 1     is assumed that the compiler has upcased
: 127     0188 1     all of the strings.
: 128     0189 1
: 129     0190 1     VALUE    - The enumerated value to write.
: 130     0191 1
: 131     0192 1     TOTAL_WIDTH - Total field width.
: 132     0193 1
: 133     0194 1     ERROR    - Optional. Address to unwind to if an error occurs.
: 134     0195 1
: 135     0196 1 IMPLICIT INPUTS:
: 136     0197 1
: 137     0198 1     NONE
: 138     0199 1

```



```
139      0200 1 | IMPLICIT OUTPUTS:
140      0201 1 |
141      0202 1 |     NONE
142      0203 1 |
143      0204 1 | ROUTINE VALUE:
144      0205 1 |
145      0206 1 |     NONE
146      0207 1 |
147      0208 1 | SIDE EFFECTS:
148      0209 1 |
149      0210 1 |     If the file is the standard file INPUT or OUTPUT, it is implicitly opened.
150      0211 1 |
151      0212 1 | SIGNALLED ERRORS:
152      0213 1 |
153      0214 1 |     LINTOOLON - line too long
154      0215 1 |     WRIINVENU - WRITE of invalid enumerated value
155      0216 1 |     NEGWIDDIG - negative Width or Digits specification is not allowed
156      0217 1 |
157      0218 1 | --
158      0219 1 |
159      0220 2 | BEGIN
160      0221 2 |
161      0222 2 | LOCAL
162      0223 2 |     FCB: REF $PASSFCB_CONTROL_BLOCK,           ! File control block
163      0224 2 |     FIELD_WIDTH,                               ! Total width
164      0225 2 |     VALUE_STRING: REF VECTOR [, BYTE],         ! Enumerated value string
165      0226 2 |     VALUE_WIDTH,                               ! Width of enumerated value
166      0227 2 |     PFV_ADDR: VOLATILE,                        ! Enable argument
167      0228 2 |     UNWIND_ACT: VOLATILE,                      ! Enable argument
168      0229 2 |     ERROR_ADDR: VOLATILE;                      ! Enable argument
169      0230 2 |
170      0231 2 | BUILTIN
171      0232 2 |     ACTUALCOUNT;
172      0233 2 |
173      0234 2 | ENABLE
174      0235 2 |     PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); ! Enable error handler
175      0236 2 |
176      0237 2 | !+
177      0238 2 | ! Get ERROR parameter, if present.
178      0239 2 | !-
179      0240 2 |
180      0241 2 | IF ACTUALCOUNT () GEQU 5
181      0242 2 | THEN
182      0243 2 |     ERROR_ADDR = .ERROR;           ! Set unwind address
183      0244 2 |
184      0245 2 | PFV_ADDR = PFV [PFV$R_PFV];       ! Set PFV address
185      0246 2 |
186      0247 2 | !+
187      0248 2 | ! Validate PFV and get PFV.
188      0249 2 | !-
189      0250 2 |
190      0251 2 | PASS$VALIDATE_PFV (PFV [PFV$R_PFV]; FCB);
191      0252 2 |
192      0253 2 | !+
193      0254 2 | ! Set unwind action to unlock file.
194      0255 2 | !-
195      0256 2 |
```



```

: 196      0257 2      UNWIND_ACT = PASSK_UNWIND_UNLOCK;
: 197      0258 2
: 198      0259 2      !+
: 199      0260 2      ! Do common initialization.
: 200      0261 2      !-
: 201      0262 2
: 202      0263 2      PASS$INIT_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]; FCB);
: 203      0264 2
: 204      0265 2      !+
: 205      0266 2      ! Check for invalid width.
: 206      0267 2      !-
: 207      0268 2
: 208      0269 2      IF .TOTAL_WIDTH LSS 0
: 209      0270 2      THEN
: 210      0271 2          $PASSIO_ERROR (PASS$_NEGWIDDIG,0);
: 211      0272 2
: 212      0273 2      !+
: 213      0274 2      ! Get field width
: 214      0275 2      !-
: 215      0276 2
: 216      0277 2      FIELD_WIDTH = .TOTAL_WIDTH;
: 217      0278 2
: 218      0279 2      !+
: 219      0280 2      ! Determine if value is out-of range and then locate value.
: 220      0281 2      !-
: 221      0282 2
: 222      0283 2      IF .VALUE GEQU .PETD [1]
: 223      0284 2      THEN
: 224      0285 2          $PASSIO_ERROR (PASS$_WRIINVENU,0);
: 225      0286 2      VALUE_STRING = .PETD + .PETD [.VALUE+2];      ! Relative address
: 226      0287 2      VALUE_WIDTH = MIN (.VALUE_STRING [0], .FIELD_WIDTH);
: 227      0288 2
: 228      0289 2      !+
: 229      0290 2      ! See if field will fit in record.
: 230      0291 2      !-
: 231      0292 2
: 232      0293 2      BEGIN
: 233      0294 2      LOCAL
: 234      0295 2          EXTRA;      ! Extra characters past end of line
: 235      0296 2      EXTRA = (.FCB [FCB$A_RECORD_CUR] + .FIELD_WIDTH) - .FCB [FCB$A_RECORD_END];
: 236      0297 2      IF .EXTRA GTR 0
: 237      0298 2      THEN
: 238      0299 2          $PASSIO_ERROR (PASS$_LINTOOLON,1,.EXTRA);
: 239      0300 2      END;
: 240      0301 2
: 241      0302 2      !+
: 242      0303 2      ! Move leading blanks, if any.
: 243      0304 2      !-
: 244      0305 2
: 245      0306 2      IF .FIELD_WIDTH - .VALUE_WIDTH GTR 0
: 246      0307 2      THEN
: 247      0308 2          FCB [FCB$A_RECORD_CUR] = CH$FILL (%C' ', .FIELD_WIDTH - .VALUE_WIDTH,
: 248      0309 2          .FCB [FCB$A_RECORD_CUR]);
: 249      0310 2
: 250      0311 2      !+
: 251      0312 2      ! Now move value
: 252      0313 2

```



```

: 253      0314 2
: 254      0315 2
: 255      0316 2
: 256      0317 2
: 257      0318 2
: 258      0319 2
: 259      0320 2
: 260      0321 2
: 261      0322 2
: 262      0323 2
: 263      0324 2
: 264      0325 2
: 265      0326 2
: 266      0327 1

FCB [FCB$A_RECORD_CUR] = CH$MOVE (.VALUE_WIDTH, VALUE_STRING [1],
    .FCB [FCB$A_RECORD_CUR]);

!+
! Call WRITE epilogue routine to move the last character written to the
! user's buffer and to unlock the file variable.
!-

PASS$END_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]);

RETURN;

END;

```

! End of routine PASSWRITE_ENUMERATED

				.TITLE	PASSWRITE_ENUMERATED Write an enumerated value	
				.IDENT	\1-002\	
				.EXTRN	PASSWRITE_ENUMERATED	
				.EXTRN	PASSWRITE_ENUMERATED	
				.EXTRN	PASS\$IO_HANDLER	
				.EXTRN	PASS\$VALIDATE_PFV	
				.EXTRN	PASS\$INIT_WRITE	
				.EXTRN	PASS\$SIGNAL, PASS\$K_NEGWIDDIG	
				.EXTRN	PASS\$K_WRIINVENU	
				.EXTRN	PASS\$K_LINTOOLON	
				.EXTRN	PASS\$END_WRITE	
				.PSECT	_PASS\$CODE, NOWRT, SHR, PIC, 2	
				.ENTRY	PASSWRITE_ENUMERATED, Save R2,R3,R4,R5,R6,-	0144
					R7,R8,R9,R10	
				MOVAB	PASS\$SIGNAL, R10	
				SUBL2	#8, SP	
				CLRL	ERROR_ADDR	0220
				CLRL	UNWIND_ACT	
				MOVAL	8\$, (FP)	
				CMPB	(AP), #5	0241
				BLSSU	1\$	
				MOVL	ERROR, ERROR_ADDR	0243
				MOVL	PFV, R6	0245
				MOVL	R6, PFV_ADDR	
				JSB	PASS\$VALIDATE_PFV	0251
				MOVL	#1, UNWIND_ACT	0257
				JSB	PASS\$INIT_WRITE	0263
				TSTL	TOTAL_WIDTH	0269
				BGEQ	2\$	
				CLRL	-(SP)	0271
				MOVZBL	#PASS\$K_NEGWIDDIG, -(SP)	
				BRB	3\$	
				MOVL	TOTAL_WIDTH, FIELD_WIDTH	0277
				MOVL	PETD, R3	0283
				CMPL	VALUE, 4(R3)	
				BLSSU	4\$	
				CLRL	-(SP)	0285
				MOVZBL	#PASS\$K_WRIINVENU, -(SP)	

	6A		02	FB	00059	3\$:	CALLS	#2, PASS\$SIGNAL	
					04		RET		
	50	0C	AC	D0	0005D	4\$:	MOVL	VALUE, R0	0286
58	53	08	A3	C1	00061		ADDL3	8(R3)[R0], R3, VALUE_STRING	
	50		68	9A	00067		MOVZBL	(VALUE_STRING), R0	0287
	52		50	D1	0006A		CMPL	R0, FIELD_WIDTH	
			03	15	0006D		BLEQ	5\$	
	50		52	D0	0006F		MOVL	FIELD_WIDTH, R0	
	59		50	D0	00072	5\$:	MOVL	R0, VALUE_WIDTH	
50	52	EC	A7	C1	00075		ADDL3	-20(FCB), FIELD_WIDTH, R0	0296
	50	FO	A7	C2	0007A		SUBL2	-16(FCB), EXTRA	
			0C	15	0007E		BLEQ	6\$	0297
			50	DD	00080		PUSHL	EXTRA	0299
			01	DD	00082		PUSHL	#1	
	7E	00G	8F	9A	00084		MOVZBL	#PASS\$ LINTOOLON, -(SP)	
	6A		03	FB	00088		CALLS	#3, PASS\$SIGNAL	
					04		RET		
	59		52	D1	0008C	6\$:	CMPL	FIELD_WIDTH, VALUE_WIDTH	0306
			0E	15	0008F		BLEQ	7\$	
	52		59	C2	00091		SUBL2	VALUE_WIDTH, R2	0308
52	6E		00	2C	00094		MOVCS	#0, (SP), #32, R2, @-20(FCB)	0309
		EC	B7		00099				
			53	D0	0009B		MOVL	R3, -20(FCB)	
	EC	B7	59	28	0009F	7\$:	MOVCS	VALUE_WIDTH, 1(VALUE_STRING), @-20(FCB)	0316
		EC	53	D0	000A5		MOVL	R3, -20(FCB)	
			00	16	000A9		JSB	PASS\$END_WRITE	0323
					04		RET		0327
					0000	8\$:	.WORD	Save nothing	0220
	50	08	AC	D0	000B2		MOVL	8(AP), R0	
	50	04	A0	D0	000B6		MOVL	4(R0), R0	
		F4	A0	9F	000BA		PUSHAB	ERROR_ADDR	
		F8	A0	9F	000BD		PUSHAB	UNWIND_ACT	
		FC	A0	9F	000C0		PUSHAB	PFV_ADDR	
			03	DD	000C3		PUSHL	#3	
			5E	DD	000C5		PUSHL	SP	
	7E	04	AC	7D	000C7		MOVQ	4(AP), -(SP)	
	00000000G	00	03	FB	000CB		CALLS	#3, PASS\$IO_HANDLER	
			04		000D2		RET		

; Routine Size: 211 bytes, Routine Base: _PASS\$CODE + 0000

; 267 0328 1
 ; 268 0329 1 !<BLF/PAGE>


```
270 0330 1 %SBTTL 'PASSWRITEV_ENUMERATED - Write ENUMERATED to string'
271 0331 1 GLOBAL ROUTINE PASSWRITEV_ENUMERATED (
272 0332 1     MAX_LENGTH: WORD,           ! Maximum length of string
273 0333 1     STRING_LINE: REF VECTOR [, WORD], ! String to write to
274 0334 1     PETD: REF VECTOR [, LONG],      ! Enumerated type descriptor
275 0335 1     VALUE,                     ! Value to write
276 0336 1     TOTAL_WIDTH: SIGNED,        ! Total field width
277 0337 1     ERROR                     ! Error unwind address
278 0338 1 ) : NOVALUE =
279 0339 1
280 0340 1 ++
281 0341 1 FUNCTIONAL DESCRIPTION:
282 0342 1
283 0343 1     This procedure writes an enumerated value to the specified string.
284 0344 1
285 0345 1 CALLING SEQUENCE:
286 0346 1
287 0347 1     CALL PASSWRITEV_ENUMERATED (MAX_LENGTH.rw.v, STRING_LINE.wvt.r,
288 0348 1     VALUE.rlu.v, TOTAL_WIDTH.rl.v [, ERROR.j.r])
289 0349 1
290 0350 1 FORMAL PARAMETERS:
291 0351 1
292 0352 1     MAX_LENGTH      - The maximum length of STRING_LINE.
293 0353 1
294 0354 1     STRING_LINE    - A varying string to which the output will be appended.
295 0355 1
296 0356 1     VALUE          - The value to write.
297 0357 1
298 0358 1     TOTAL_WIDTH    - The width of the field to write.
299 0359 1
300 0360 1     ERROR        - Optional. If specified, the address to unwind to
301 0361 1                   in case of an error.
302 0362 1
303 0363 1 IMPLICIT INPUTS:
304 0364 1
305 0365 1     NONE
306 0366 1
307 0367 1 IMPLICIT OUTPUTS:
308 0368 1
309 0369 1     NONE
310 0370 1
311 0371 1 ROUTINE VALUE:
312 0372 1
313 0373 1     NONE
314 0374 1
315 0375 1 SIDE EFFECTS:
316 0376 1
317 0377 1     NONE
318 0378 1
319 0379 1 SIGNALLED ERRORS:
320 0380 1
321 0381 1     See PASSWRITE_ENUMERATED
322 0382 1
323 0383 1 --
324 0384 1
325 0385 2 BEGIN
326 0386 2
```



```

327 0387 2 LOCAL
328 0388 2 PFV: $PASS$PFV FILE VARIABLE, ! Pascal File Variable
329 0389 2 ARG_LIST: VECTOR [5, LONG], ! Argument list
330 0390 2 PFV_ADDR: VOLATILE, ! Enable argument
331 0391 2 UNWIND_ACT: VOLATILE, ! Enable argument
332 0392 2 ERROR_ADDR: VOLATILE; ! Enable argument
333 0393 2
334 0394 2 BUILTIN
335 0395 2 ACTUALCOUNT; ! Count of arguments
336 0396 2
337 0397 2 ENABLE
338 0398 2 PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); ! Enable error handler
339 0399 2
340 0400 2 !+
341 0401 2 ! Get ERROR parameter, if present.
342 0402 2 !-
343 0403 2
344 0404 2 IF ACTUALCOUNT () GEQU 6
345 0405 2 THEN
346 0406 2 ERROR_ADDR = .ERROR; ! Set unwind address
347 0407 2
348 0408 2 PFV_ADDR = PFV [PFV$R_PFV]; ! Set PFV address
349 0409 2
350 0410 2 !+
351 0411 2 ! Set up ARG_LIST.
352 0412 2 !-
353 0413 2
354 0414 2 ARG_LIST [0] = 4; ! Four arguments
355 0415 2 ARG_LIST [1] = PFV [PFV$R_PFV]; ! PFV address
356 0416 2 ARG_LIST [2] = .PETD; ! Enumerated type descriptor
357 0417 2 ARG_LIST [3] = .VALUE; ! Value to write
358 0418 2 ARG_LIST [4] = .TOTAL_WIDTH; ! Field width
359 0419 2
360 0420 2 !+
361 0421 2 ! Call PASS$DO_WRITEV to do the work, giving it the address of
362 0422 2 ! PASS$WRITE_ENUMERATED to call.
363 0423 2 !-
364 0424 2
365 0425 2 PASS$DO_WRITEV (PFV [PFV$R_PFV], .MAX_LENGTH, STRING_LINE [0], ARG_LIST,
366 0426 2 PASS$WRITE_ENUMERATED);
367 0427 2
368 0428 2 RETURN;
369 0429 2
370 0430 1 END; ! End of routine PASS$WRITEV_ENUMERATED

```

```
.EXTRN  PASS$DO_WRITEV
```

			007C	00000
5E		2C	C2	00002
		7E	D4	00005
	04	AE	7C	00007
6D	003E	CF	DE	0000A
06		6C	91	0000F
		04	1F	00012
6E	18	AC	D0	00014

```

.ENTRY PASSWRITEV_ENUMERATED, Save R2,R3,R4,R5,R6 ; 0331
SUBL? #44, SP ;
CLRL ERROR_ADDR ; 0385
CLRQ UNWIND_ACT ;
MOVAL 2$, (FP) ;
CMPB (AP), #6 ; 0404
BLSSU 1$ ;
MOVL ERROR, ERROR_ADDR ; 0406

```

08	AE	20	AE	9E	00018	1\$:	MOVAB	PFV, PFV_ADDR	:	0408
0C	AE		04	D0	0001D		MOVL	#4, ARG_LIST	:	0414
10	AE	20	AE	9E	00021		MOVAB	PFV, ARG_LIST+4	:	0415
14	AE	0C	AC	7D	00026		MOVQ	PETD, ARG_LIST+8	:	0416
1C	AE	14	AC	D0	0002B		MOVL	TOTAL_WIDTH, ARG_LIST+16	:	0418
	55	FEF9	CF	9E	00030		MOVAB	PASSWRITE_ENUMERATED, R5	:	0425
	54	0C	AE	9E	00035		MOVAB	ARG_LIST, R4	:	
	56	20	AE	9E	00039		MOVAB	PFV, R6	:	
	53	08	AC	D0	0003D		MOVL	STRING_LINE, R3	:	
	52	04	AC	3C	00041		MOVZWL	MAX_LENGTH, R2	:	
		00000000G	00	16	00045		JSB	PASS\$DO_WRITEV	:	
				04	0004B		RET		:	0430
				0000	0004C	2\$:	.WORD	Save nothing	:	0385
50		08	AC	D0	0004E		MOVL	8(AP), R0	:	
50		04	A0	D0	00052		MOVL	4(R0), R0	:	
		D0	A0	9F	00056		PUSHAB	ERROR_ADDR	:	
		D4	A0	9F	00059		PUSHAB	UNWIND_ACT	:	
		D8	A0	9F	0005C		PUSHAB	PFV_ADDR	:	
			03	DD	0005F		PUSHL	#3	:	
			5E	DD	00061		PUSHL	SP	:	
	7E	04	AC	7D	00063		MOVQ	4(AP), -(SP)	:	
00000000G	00		03	FB	00067		CALLS	#3, PASS\$IO_HANDLER	:	
				04	0006E		RET		:	

; Routine Size: 111 bytes, Routine Base: _PASSCODE + 00D3

; 371 0431 1
 ; 372 0432 1 !<BLF/PAGE>

:

374

0433

1

END

:

375

0434

1

:

376

0435

0

ELUDOM

! End of module PASSWRITE_ENUMERATED

PSECT SUMMARY

:

:

:

:

:

Name

Bytes

Attributes

_PASS\$CODE

322

NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

:

:

:

:

:

File

Symbols

Pages

Processing

Total

Loaded

Percent

Mapped

Time

\$255\$DUA28:[SYSLIB]STARLET.L32;1

9776

0

0

581

00:01.0

_\$255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1

427

97

22

33

00:00.4

COMMAND QUALIFIERS

:

:

:

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

)

:

:

:

:

:

:

:

:

:

:

377

0436

0

Size:

322

code + 0 data bytes

Run Time:

00:08.3

Elapsed Time:

00:36.2

Lines/CPU Min:

3170

Lexemes/CPU-Min:

13876

Memory Used:

96 pages

Compilation Complete

0297

AH-BT13A-SE
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY